anscombe_eda

October 9, 2025

1 Anscombe's Quartet – Exploratory Data Analysis (EDA)

Author: Ivan Johnson
Date: 2025-10-08

This notebook explores **Anscombe's Quartet**, a famous dataset created by statistician Francis Anscombe in 1973 to show the importance of **data visualization**.

Even though all four datasets share nearly identical statistical properties (mean, variance, correlation, etc.), their scatter plots reveal *strikingly different patterns*.

[ ]:
```
!pip install seaborn matplotlib plotly altair pandas numpy
```

[18]:
```
import seaborn as sns
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import os
import plotly.express as px
import altair as alt

# Create output folder
os.makedirs("output", exist_ok=True)
```

1.1 Loading Anscombe's Quartet

Seaborn includes the Anscombe dataset built-in.
Let's load and preview it.

[19]:
```
df = sns.load_dataset("anscombe")
print("Dataset preview:")
df.head()
```

```
Dataset preview:
```

[19]:

	dataset	x	y
0	I	10.0	8.04
1	I	8.0	6.95
2	I	13.0	7.58
3	I	9.0	8.81

```
4       I  11.0  8.33
```

Each dataset (I, II, III, IV) has the same basic statistics, but different patterns.
Let's group them for analysis.

[20]:
```
datasets = {name: group for name, group in df.groupby("dataset")}
for name, data in datasets.items():
    print(f"\n{name} dataset:")
    print(data.describe())
```

```
I dataset:
                x          y
count  11.000000  11.000000
mean    9.000000   7.500909
std     3.316625   2.031568
min     4.000000   4.260000
25%     6.500000   6.315000
50%     9.000000   7.580000
75%    11.500000   8.570000
max    14.000000  10.840000

II dataset:
                x          y
count  11.000000  11.000000
mean    9.000000   7.500909
std     3.316625   2.031657
min     4.000000   3.100000
25%     6.500000   6.695000
50%     9.000000   8.140000
75%    11.500000   8.950000
max    14.000000   9.260000

III dataset:
                x          y
count  11.000000  11.000000
mean    9.000000   7.500000
std     3.316625   2.030424
min     4.000000   5.390000
25%     6.500000   6.250000
50%     9.000000   7.110000
75%    11.500000   7.980000
max    14.000000  12.740000

IV dataset:
                x          y
count  11.000000  11.000000
mean    9.000000   7.500909
```

```
std      3.316625   2.030579
min      8.000000   5.250000
25%      8.000000   6.170000
50%      8.000000   7.040000
75%      8.000000   8.190000
max     19.000000  12.500000
```

1.2 Summary Statistics

Let's compute summary statistics for all four datasets together to compare.

[21]:
```
summary = df.groupby("dataset").agg({
    "x": ["mean", "std", "var", "min", "max"],
    "y": ["mean", "std", "var", "min", "max"]
})
print(summary)
```

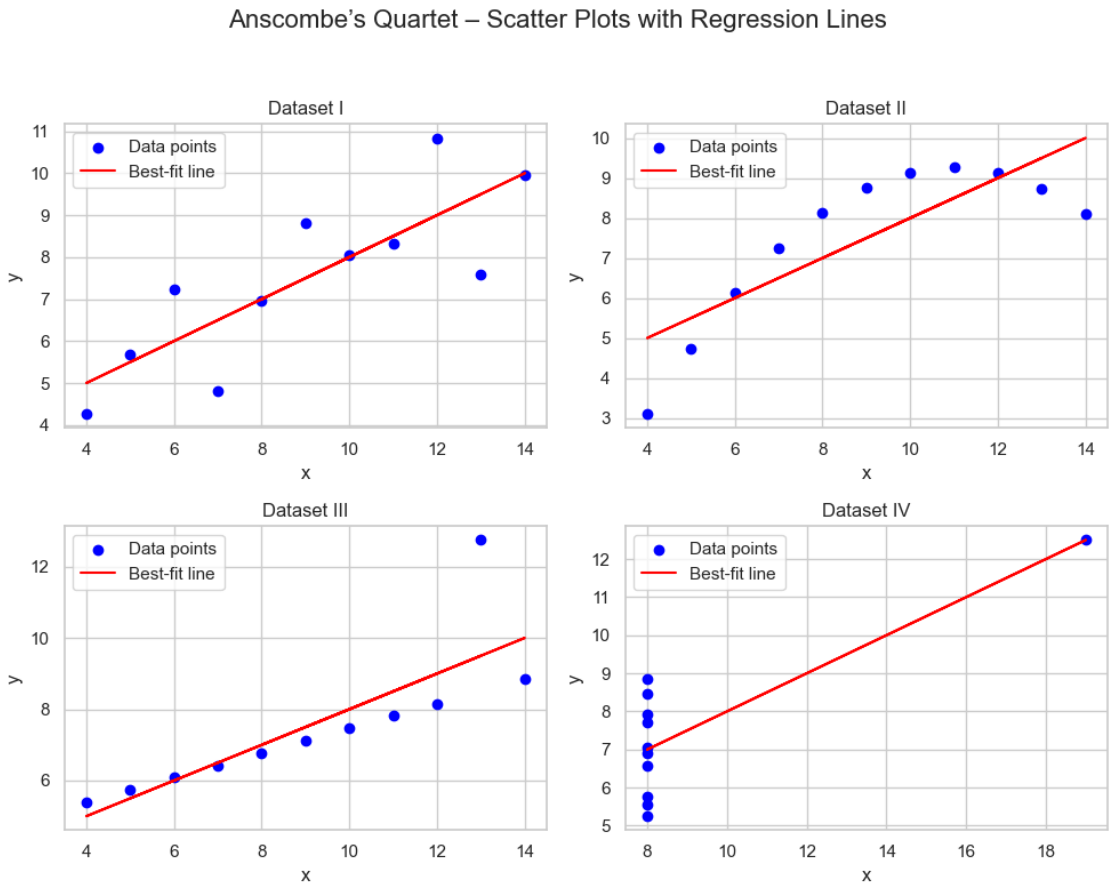
```
           x                                        y                          \
        mean       std   var  min   max      mean       std       var   min
dataset
I        9.0  3.316625  11.0  4.0  14.0  7.500909  2.031568  4.127269  4.26
II       9.0  3.316625  11.0  4.0  14.0  7.500909  2.031657  4.127629  3.10
III      9.0  3.316625  11.0  4.0  14.0  7.500000  2.030424  4.122620  5.39
IV       9.0  3.316625  11.0  8.0  19.0  7.500909  2.030579  4.123249  5.25


           max
dataset
I        10.84
II        9.26
III      12.74
IV       12.50
```

1.3 Scatter Plots (Matplotlib)

We'll visualize each dataset separately to show how patterns differ.

[26]:
```
sns.set(style="whitegrid")

fig, axes = plt.subplots(2, 2, figsize=(10, 8))
axes = axes.flatten()

for i, (name, data) in enumerate(datasets.items()):
    ax = axes[i]
    ax.scatter(data["x"], data["y"], color="blue", label="Data points")

    # Add regression line (best-fit)
    m, b = np.polyfit(data["x"], data["y"], 1)
    ax.plot(data["x"], m * data["x"] + b, color="red", label="Best-fit line")
```

```
    ax.set_title(f"Dataset {name}")
    ax.set_xlabel("x")
    ax.set_ylabel("y")
    ax.legend()

plt.suptitle("Anscombe's Quartet - Scatter Plots with Regression Lines",␣
 ↪fontsize=16)
plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.savefig("output/anscombe_scatter.png")
plt.show()
```

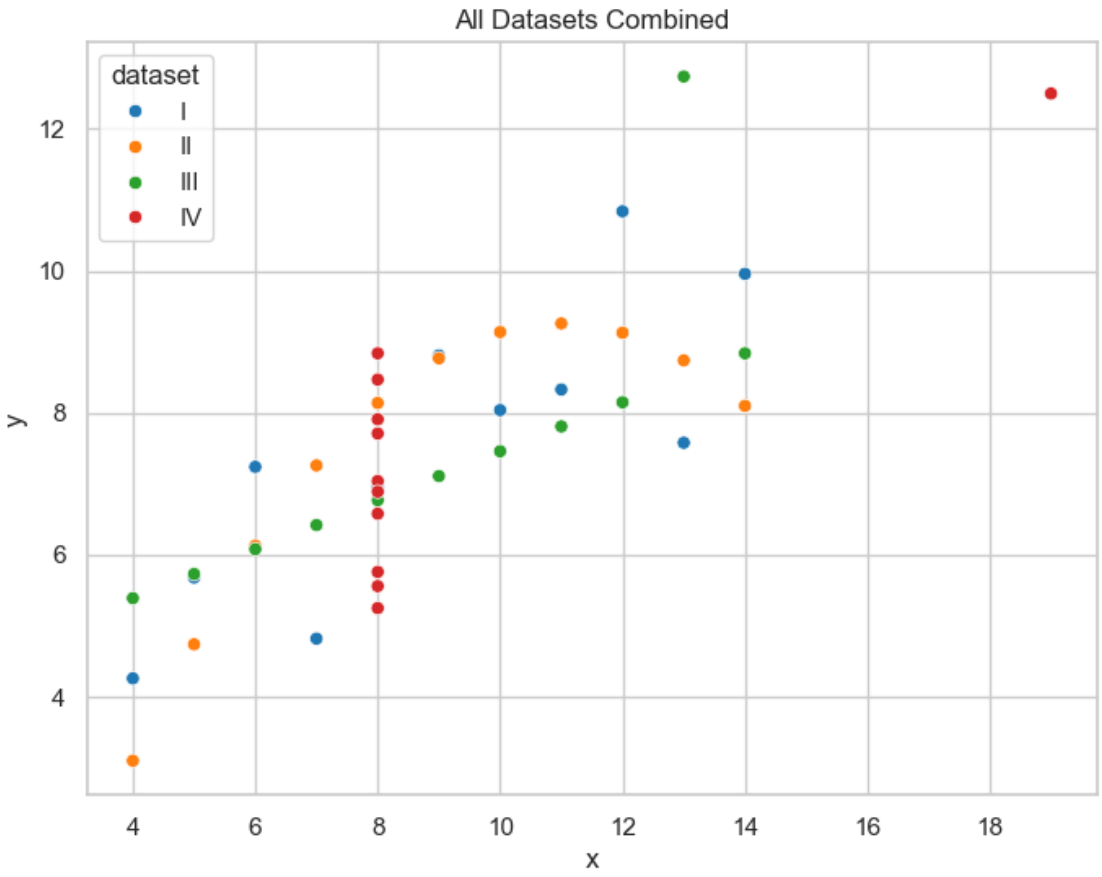


1.4 Combined Graph

Now let's see all datasets on one plot.

[23]:
```
plt.figure(figsize=(8, 6))
sns.scatterplot(data=df, x="x", y="y", hue="dataset", palette="tab10")
plt.title("All Datasets Combined")
```

```
plt.savefig("output/anscombe_combined.png")
plt.show()
```

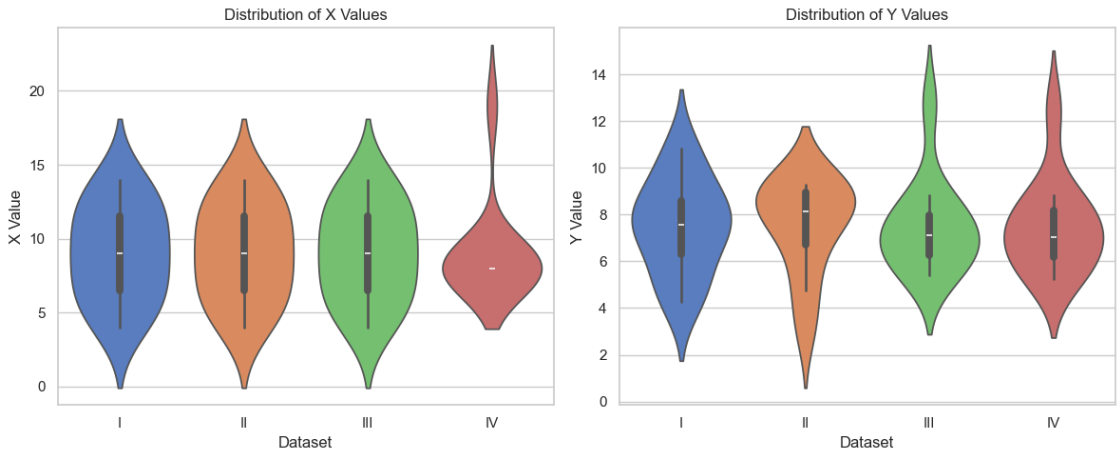


1.5 Violin Plots for Distribution Comparison

Violin plots show the distribution of x and y values for each dataset.

[31]:
```
# === Violin Plots for X and Y (Clean Version) ===
sns.set(style="whitegrid")
plt.figure(figsize=(12, 5))

# Violin plot for X
plt.subplot(1, 2, 1)
sns.violinplot(
    data=df,
    x="dataset",
    y="x",
    hue="dataset",
    palette="muted",
```

```
    legend=False
)
plt.title("Distribution of X Values")
plt.xlabel("Dataset")
plt.ylabel("X Value")

# Violin plot for Y
plt.subplot(1, 2, 2)
sns.violinplot(
    data=df,
    x="dataset",
    y="y",
    hue="dataset",
    palette="muted",
    legend=False
)
plt.title("Distribution of Y Values")
plt.xlabel("Dataset")
plt.ylabel("Y Value")

plt.tight_layout()
plt.savefig("output/anscombe_violin_xy.png")
plt.show()
```

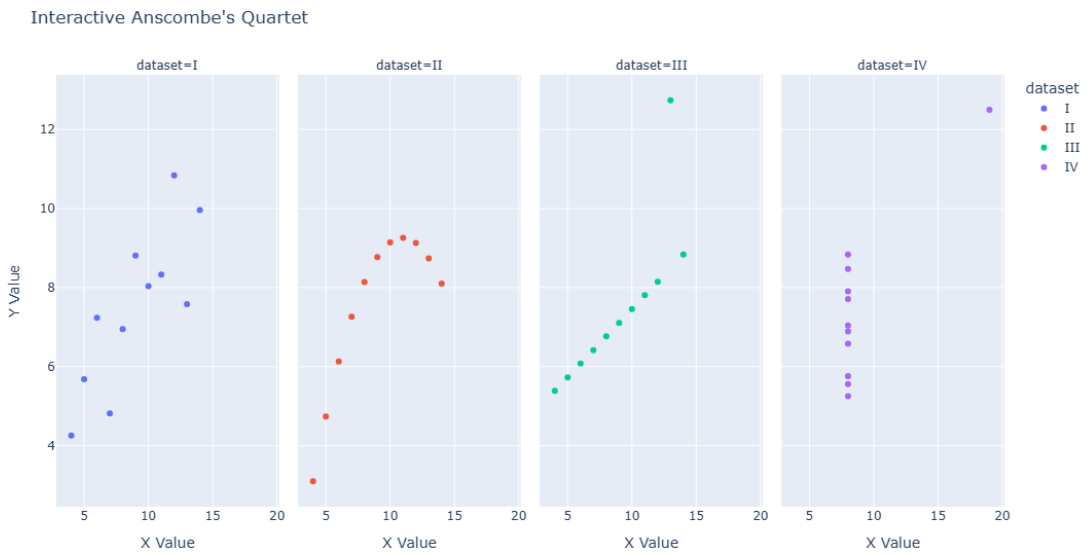


1.6 Interactive Scatter Plot (Plotly)

Let's build an interactive version of the scatter plots using **Plotly**.

[25]:
```
fig = px.scatter(
    df,
    x='x',
```

```
    y='y',
    color='dataset',
    facet_col='dataset',
    title="Interactive Anscombe's Quartet",
    labels={'x':'X Value', 'y':'Y Value'}
)

fig.update_layout(height=600, width=900)
fig.write_html("output/anscombe_plotly.html")
fig.show()
```



1.7 Reflection

While all datasets have **similar means, variances, and correlation coefficients**, their **visual patterns** are dramatically different.

This perfectly demonstrates Anscombe's message:
> *"Statistics are not enough — always visualize your data."*