3123588

slido

Join at slido.com #3123588

Click **Present with Slido** or install our Chrome extension to display joining instructions for participants while presenting.

3123588

LECTURE 7

Visualization I

Visualizing distributions and KDEs

Data 100, Summer 2025 @ UC Berkeley

Josh Grossman and Michael Xiao

📣 Announcements

3123588

Homework 2A due tonight!

Lab 2B due tomorrow!

Homework 2B due Monday, July 7th

Reminder to make sure your **DSP accommodations are submitted ASAP**

- **By Sunday, July 6th** at the latest
- Very important if you have exam accommodations

Goals for this Lecture

Lecture 7, Data 100 Summer 2025

Understand the theories behind effective visualizations and start to generate plots of our own

- The necessary "pre-thinking" before creating a plot
- Python libraries for visualizing data

Where Are We?

3123588

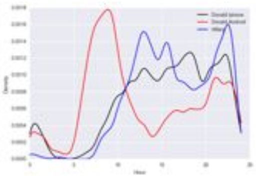
?

Question & Problem Formulation

Data Acquisition

Prediction and Inference

Exploratory Data Analysis



Reports, Decisions, and Solutions

(today)

Data Wrangling Intro to EDA	Working with Text Data Regular Expressions	Plots and variables `seaborn` KDE	Viz principles Transformations

(Part I: Processing Data)

(Part II: Visualizing and Reporting Data)

- Visualization
  - Goals of visualization
  - Visualizing distributions
  - Kernel density estimation (KDE)

Agenda

Lecture 7, Data 100 Summer 2025

Goals of Visualization

Lecture 7, Data 100 Summer 2025

- **Visualization**
  - **Goals of visualization**
  - Visualizing distributions
  - Kernel density estimation (KDE)

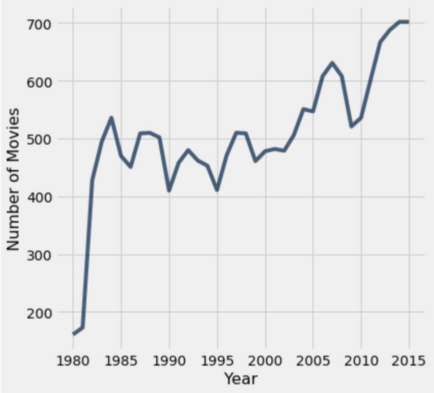
3123588

Visualizations in Data 8 (and Data 100, so far)

3123588

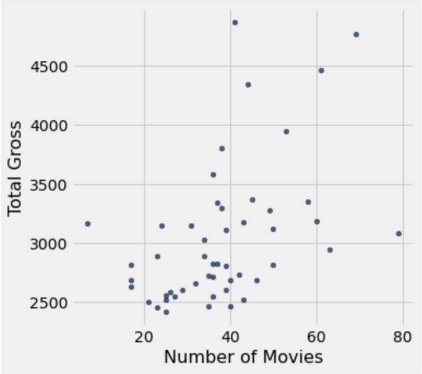
You have worked with several types of visualizations so far.

```
# The call is
# t.plot(x_label, y_label)

movies.plot('Year', 'Number of Movies')
```

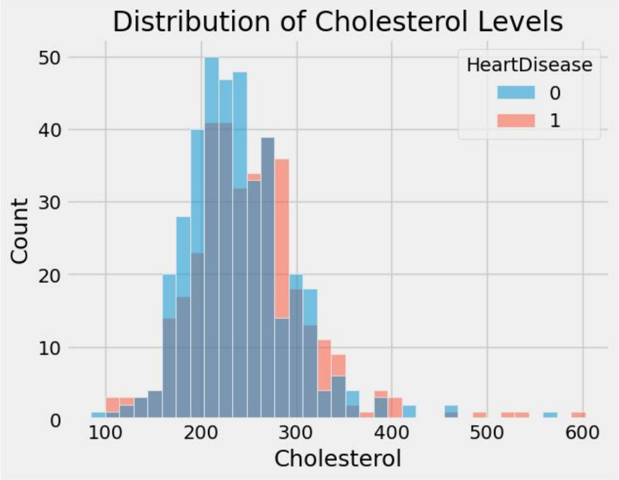


Line plot

```
# The call is
# t.scatter(x_label, y_label)

actors.scatter('Number of Movies', 'Total Gross')
```



Scatterplot



Histogram from Homework #1

What did these plots achieve?

- High-level **summary** of a complex dataset.
- **Communicate** trends to viewers.

Goals of Data Visualization

Goal 1: To **help your own understanding** of your data/results.

- Essential part of EDA.
- Summarize trends visually.
- Lightweight, iterative, and flexible.

Goal 2: To **communicate results/conclusions to others**.

- Highly editorial and selective.
- Be thoughtful and careful!
- Fine-tuned to achieve a communications goal.
- Considerations: clarity, accessibility, and context.

What do these goals imply?

Visualizations aren't a matter of making "pretty" pictures.

We need to do a lot of thinking about what stylistic choices communicate ideas most effectively.

Goals of Data Visualization ("Data Viz")

3123588

Visualizations aren't a matter of making "pretty" pictures.

We need to **think hard** about what stylistic choices communicate ideas most effectively.

1st half of Data 100 viz: **Choosing the "right" plot**

- Different plots for different variable types
- Frameworks+code required to generate

2nd half of Data 100 viz: **Stylizing plots**

- Transformations of visual data
- Context through labels and color

Big mindset change: Minimize your audience's **cognitive burden** by shouldering that burden yourself. You want to make your plot so smooth+intuitive that the reader spends little to no time thinking about how hard you worked. Don't make lazy plots.

Visualizing Distributions

Lecture 7, Data 100 Summer 2025

3123588

- **Visualization**
  - Goals of visualization
  - **Visualizing distributions**
  - Kernel density estimation (KDE)

Distributions

3123588

A **univariate** distribution describes:

- The set of possible values of **one** variable.
- The frequency of each value.

Counts should **sum to the total # of datapoints**.

Percentages should **sum to 100%**.

Cal Undergrad Major	# of Degrees (2024)
Computer Science	891
Data Science	846
Economics	678
. . .	. . .

Source

Example distribution

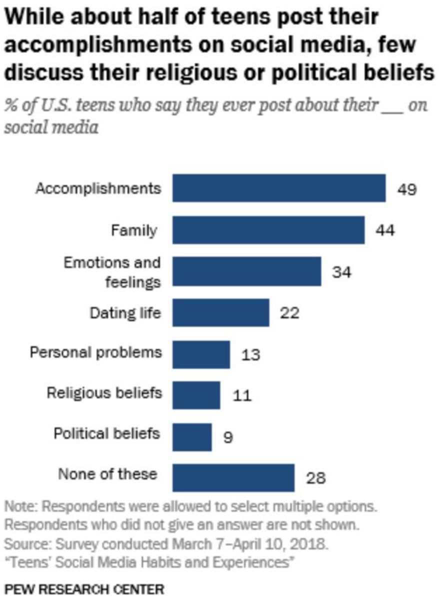
Technical note: This is the description of a discrete univariate distribution. We will not work much with continuous distributions in Data 100.

3123588

slido

Does this chart show a distribution?

Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.



3123588

Does this chart show a distribution?

No.

The chart shows percentages of individuals in different categories.

But, this is <u>not</u> a distribution because categories overlap. See fine print!

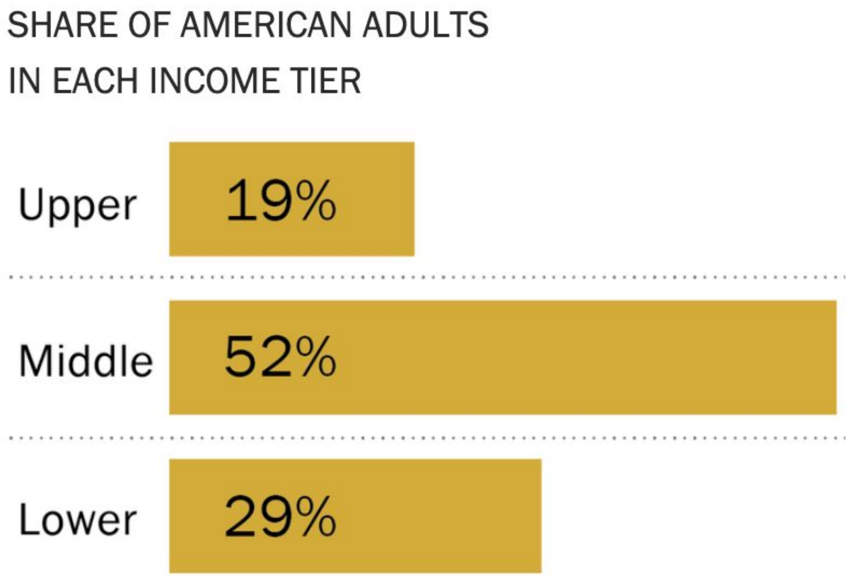
The percentages do not sum to 100%.

slido

3123588

Does this chart show a distribution?

Click **Present with Slido** or install our Chrome extension to activate this poll while presenting.



Does this chart show a distribution?

3123588

Yes!

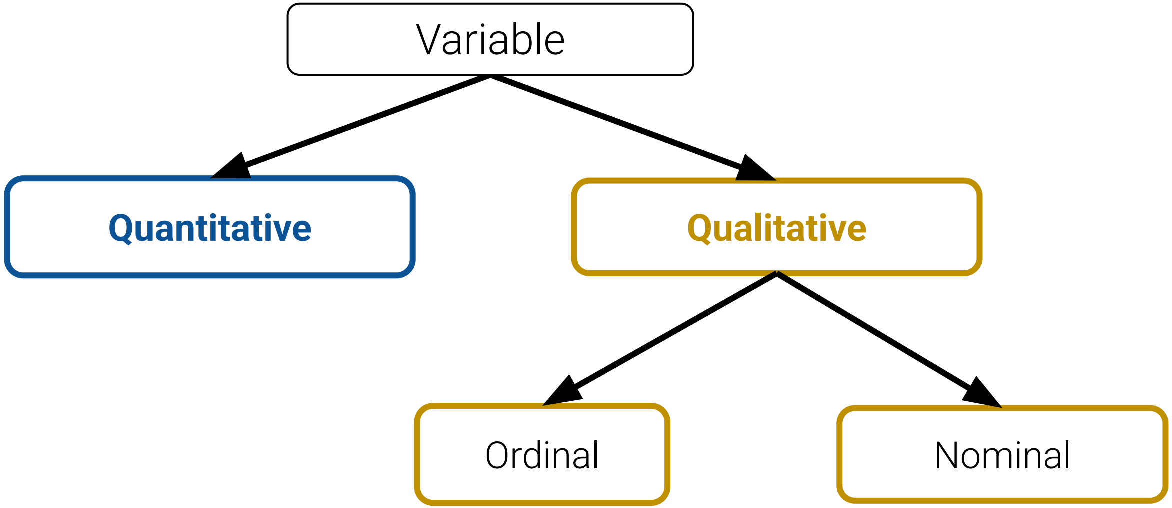
The values are the proportions of individuals in each category.

Each individual is in exactly one category.

The total percentage is 100%.

Variable Types Should Inform Plot Choice

Some plots are better suited for particular types of variables. Our friends from Lecture 5!

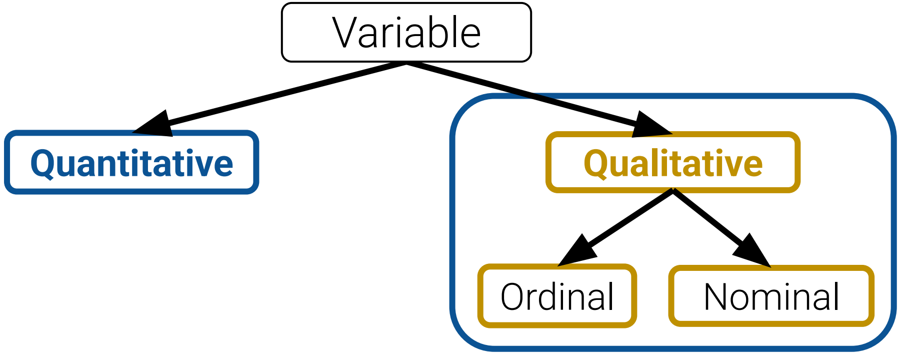


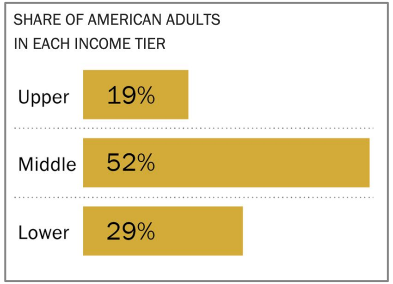
Step 1 of visualization: Pick the variables to visualize. Then, select an appropriate plot type.

Bar Plots: Distributions of Qualitative Variables

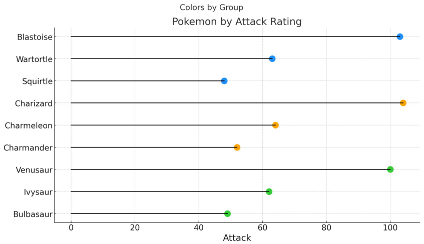
3123588

Bar plots are the most common way of displaying the **distribution** of a **qualitative** variable.

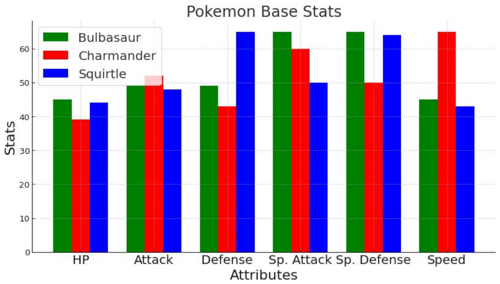




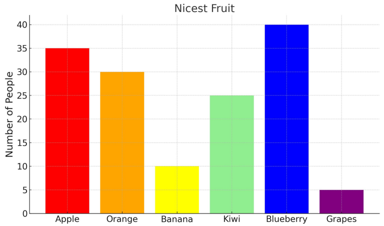
[Horizontal bars are often preferable to vertical bars]!



[Only bar length matters. Width does not!]



Color can be used for groups.



Sometimes colors are just pretty! But, avoid [chart junk].

World Bank Dataset

We will be using the `wb` dataset about world countries for most of our work today.

	Continent	Country	Primary completion rate: Male: % of relevant age group: 2015	Primary completion rate: Female: % of relevant age group: 2015	Lower secondary completion rate: Male: % of relevant age group: 2015	Lower secondary completion rate: Female: % of relevant age group: 2015	Youth literacy rate: Male: % of ages 15-24: 2005-14	Youth literacy rate: Female: % of ages 15-24: 2005-14	Adult literacy rate: Male: % ages 15 and older: 2005-14	Adult literacy rate: Female: % ages 15 and older: 2005-14
0	Africa	Algeria	106.0	105.0	68.0	85.0	96.0	92.0	83.0	68.0
1	Africa	Angola	NaN	NaN	NaN	NaN	79.0	67.0	82.0	60.0
2	Africa	Benin	83.0	73.0	50.0	37.0	55.0	31.0	41.0	18.0
3	Africa	Botswana	98.0	101.0	86.0	87.0	96.0	99.0	87.0	89.0
5	Africa	Burundi	58.0	66.0	35.0	30.0	90.0	88.0	89.0	85.0

Generating Bar Plots: Matplotlib

3123588

In Data 100, we will use three libraries to make plots: [matplotlib](), [seaborn](), and [plotly]().

Most `matplotlib` plotting functions follow the same structure: Pass in a sequence (`list`, `array`, or `series`) of **x-axis** values, and a sequence of **y-axis** values.

`matplotlib` alias is `plt`

```
import matplotlib.pyplot as plt
plt.example_plotting_function(x_values, y_values)
```

To add labels and a title:

```
plt.xlabel("x axis label")
plt.ylabel("y axis label")
plt.title("Title of the plot");
```

Generating Bar Plots: `matplotlib`

3123588

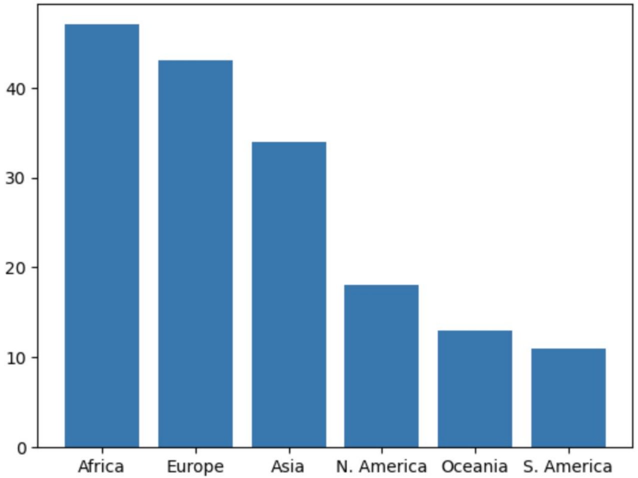
To create a bar plot in `matplotlib`: `plt.bar(___)`
[Documentation]

```
continents = wb["Continent"].value_counts()
```

```
Africa        47
Europe        43
Asia          34
N. America    18
Oceania       13
S. America    11
Name: Continent, dtype: int64
```

```
plt.bar(continents.index, continents.values);
```

$\underbrace{\texttt{continents.index}}_{\text{x values}}$, $\underbrace{\texttt{continents.values}}_{\text{y values}}$



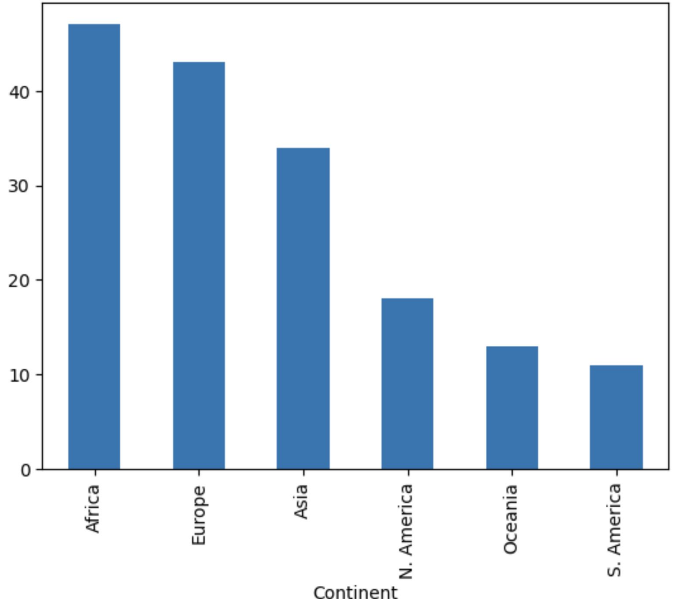
Generating Bar Plots: pandas Native Plotting

3123588

To create a bar plot in native **pandas**: `.plot(kind='bar')`

```
Africa        47
Europe        43
Asia          34
N. America    18
Oceania       13
S. America    11
Name: Continent, dtype: int64
```

```
wb["Continent"].value_counts().plot(kind='bar')
```



In general, don't use **pandas** for anything but basic+default plots in EDA.

3123588

`seaborn` has different syntax: Pass in a `DataFrame` and specify which column(s) to plot.

Seaborn alias is `sns` (This is why)

```
import seaborn as sns
sns.example_plotting_function(data=df, x="x_col", y="y_col")
```

To add labels and a title, use the same syntax as before:

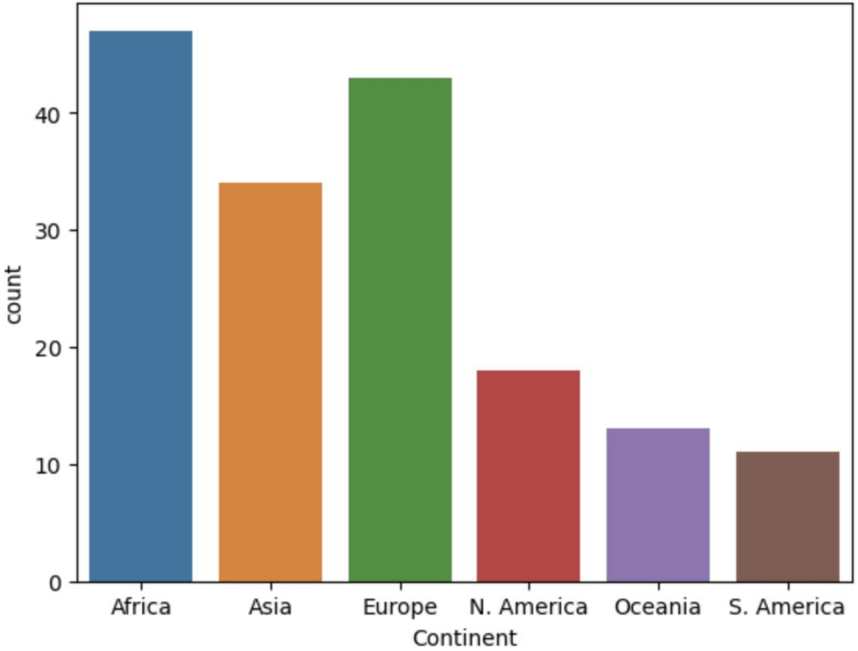
```
plt.xlabel("x axis label")
plt.ylabel("y axis label")
plt.title("Title of the plot");
```

`seaborn` is actually just `matplotlib` under the hood, but with an easier-to-use interface for working with `DataFrames` and creating certain types of plots.

Generating Bar Plots: `seaborn`

To create a bar plot in **`seaborn`**: **`sns.countplot(___)`**

[Documentation]



`countplot` operates at a higher level of abstraction!

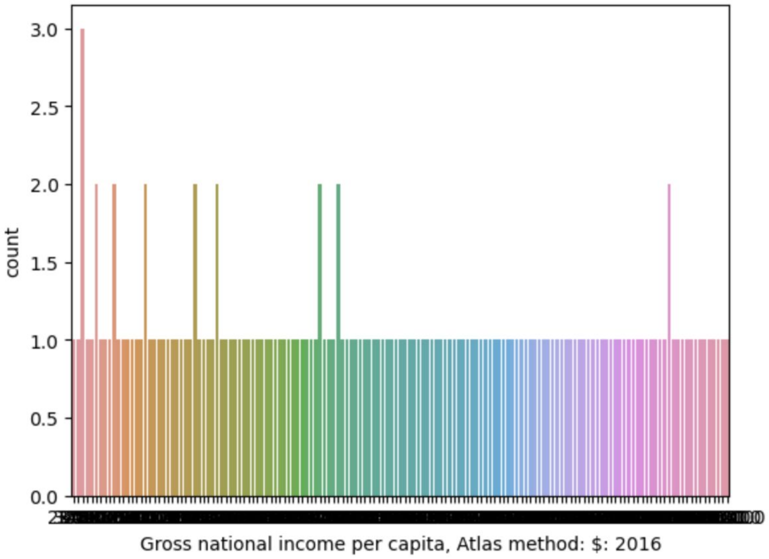
You give it the entire **`DataFrame`** and it does the counting for you.

```python
import seaborn as sns

sns.countplot(data=wb, x="Continent");
```

3123588

Why are bar plots only appropriate for **qualitative** variables, and not **quantitative**?

Consider the distribution of gross national income per capita, as a bar chart:

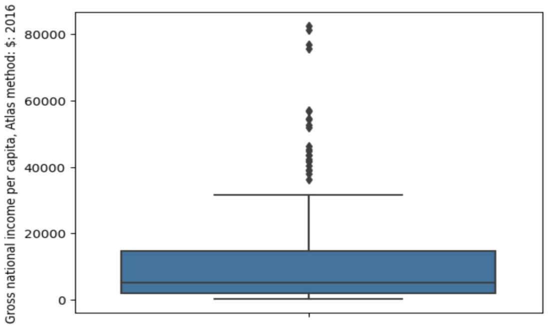


A bar plot has a separate bar for each **unique** x-axis value.

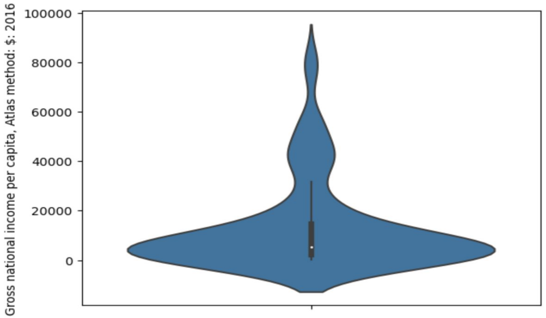
There are almost as many bars as data points! Not helpful.

3123588

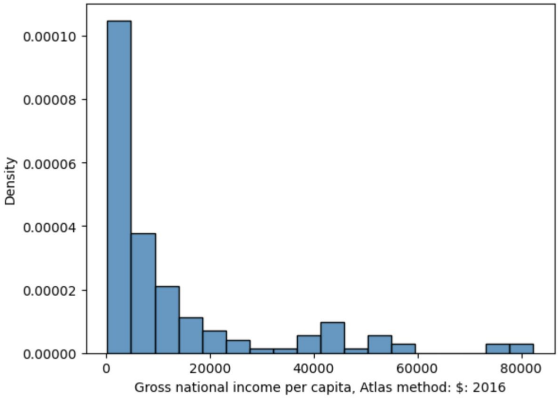
Appropriate plots for continuous **quantitative** variables:



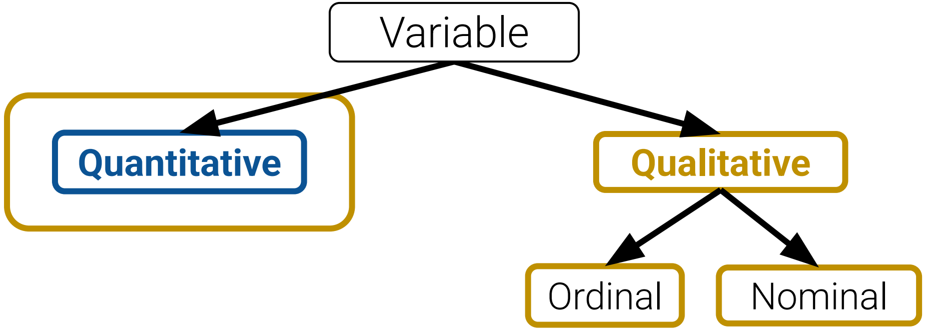
Box plot



Violin plot



Histogram

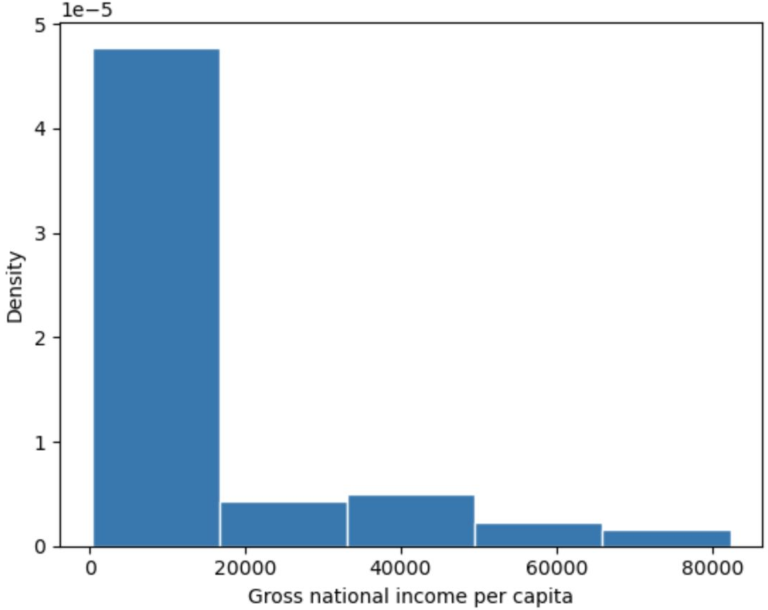


Histograms

3123588

A histogram:

- Groups datapoints with similar values into shared **bins**.
- Each bin's **<u>area</u>** (not height!) is the **percentage** of all datapoints it contains (as in [Data 8]).



The first bin has a width of \$16410 and a height (**density**) of 4.77×10^{-5}

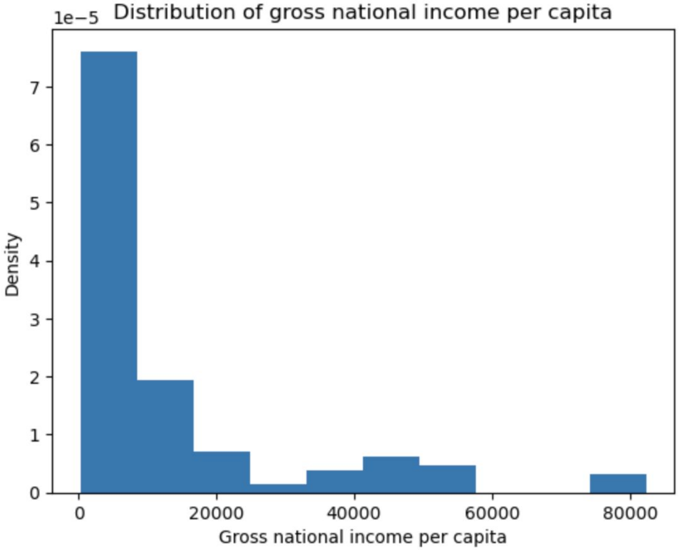
This means that it contains $16410 \times (4.77 \times 10^{-5})$ = **78.3%** of all datapoints in the dataset.

Density is not probability. Density is important for continuous distributions, which we don't work with much in Data 100. See Data 140!

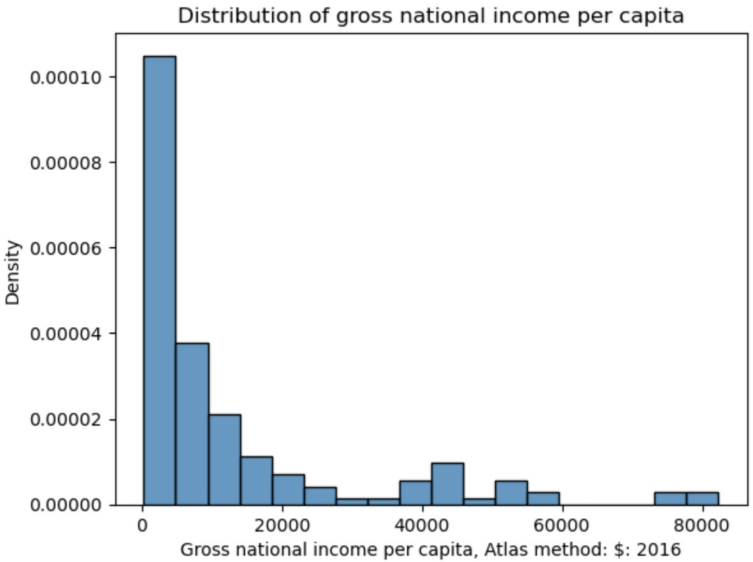
Histograms in Code

In `matplotlib` [Documentation]: `plt.hist(x_values, density=True)`

In `seaborn` [Documentation]: `sns.histplot(data=df, x="x_column", stat="density")`



`matplotlib`



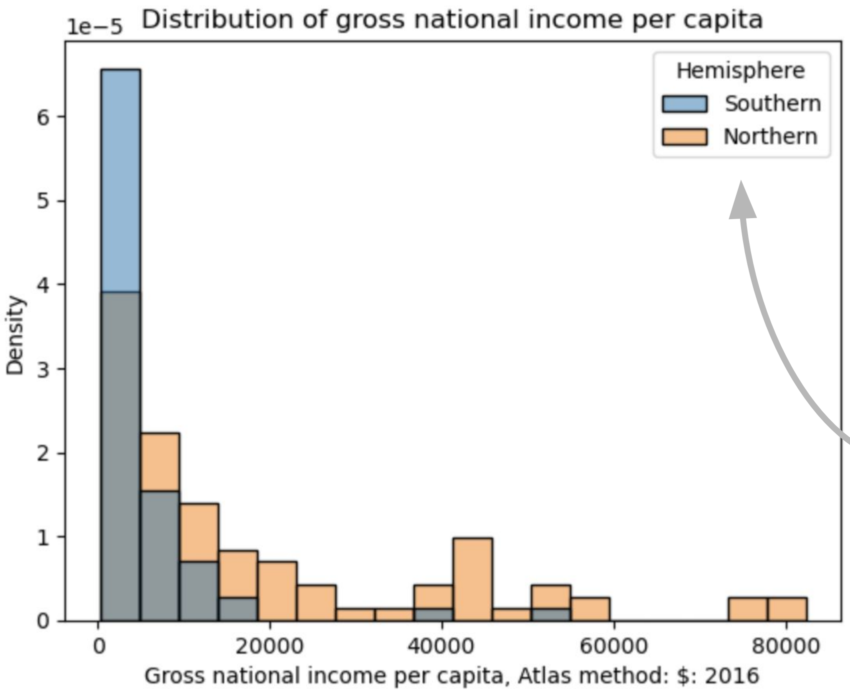
`seaborn`

3123588

Overlaid Histograms

3123588

Overlay histograms to compare **quantitative** distributions across **qualitative** categories.



The **`hue`** parameter of **`seaborn`** plotting functions sets the column that should be used to determine color.

```
sns.histplot(data=wb, hue="Hemisphere",
x="Gross national income…")
```

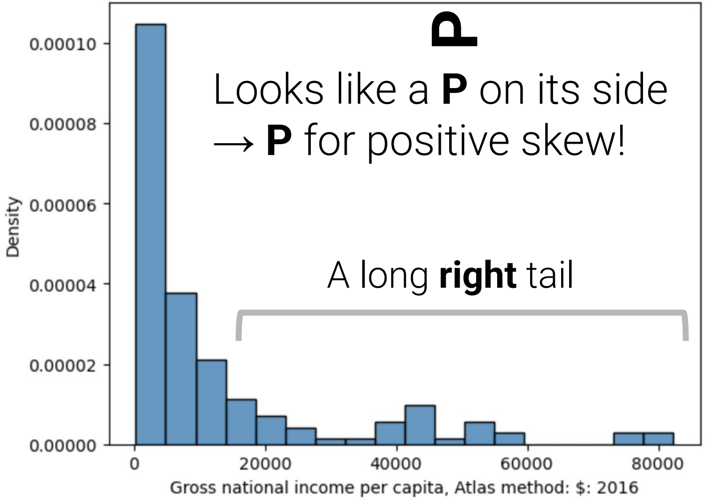
Always include a legend when color is used to encode information!

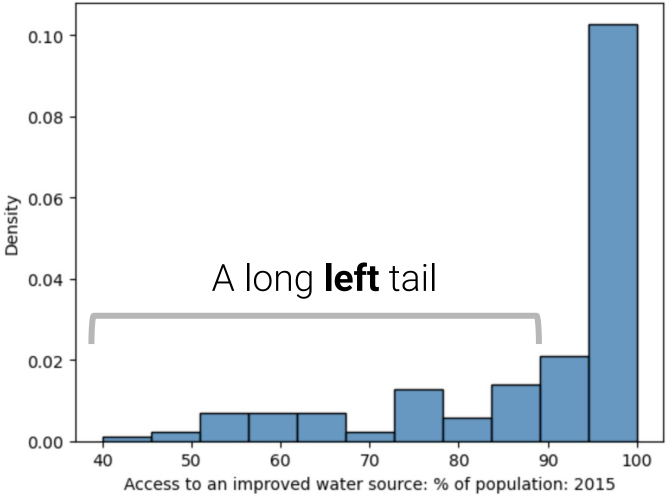
Interpreting Histograms

The **skew** of a histogram describes the direction in which its "tail" extends.

- A distribution with a long right tail is **right/positive** skewed → Mean > Median
- A distribution with a long left tail is **left/negative** skewed → Mean < Median

A histogram with no clear skew is called symmetric.

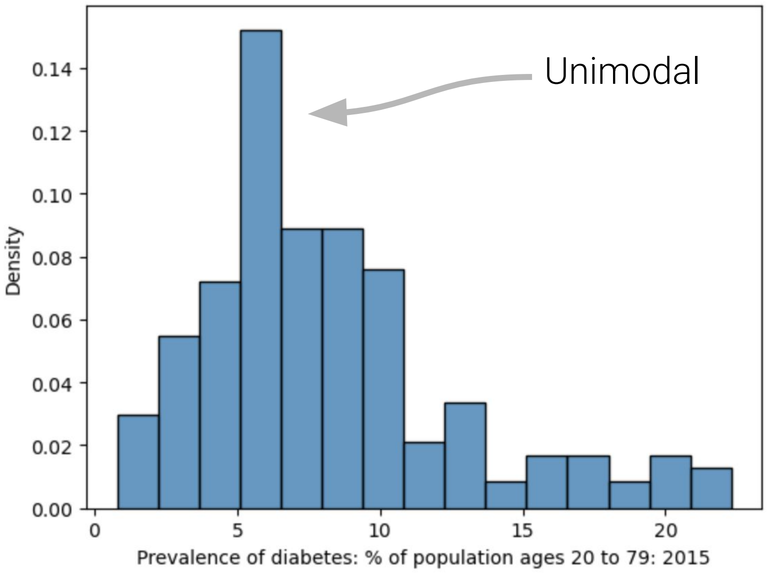




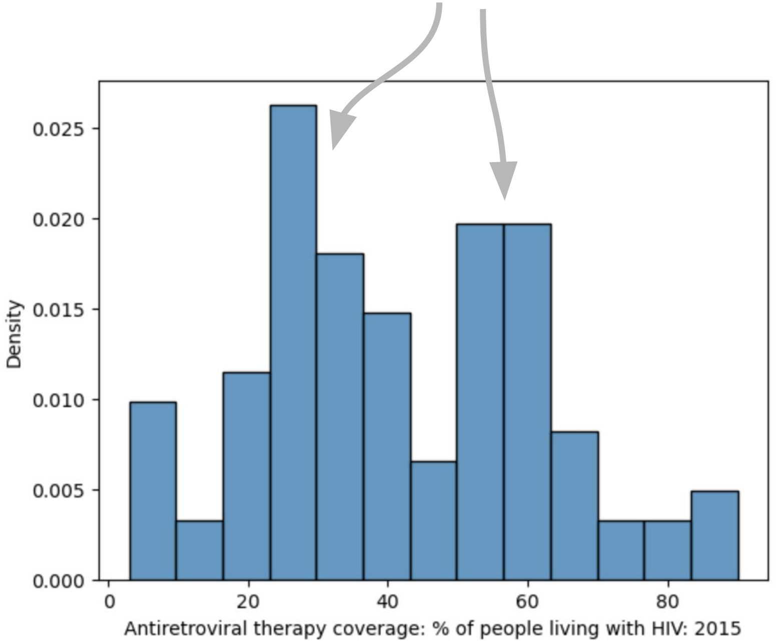
3123588

The **mode** is the **mo**st frequent value of a distribution.

- A distribution with one clear peak is called **unimodal**.
- Two peaks: bimodal.
- More peaks: multimodal.



Bimodal?



Quartiles

3123588

For a quantitative variable:

- **1st quartile** (**Q1**): 25th percentile.
- 2nd quartile: 50th percentile (**median**).
- **3rd quartile** (**Q3**): 75th percentile.

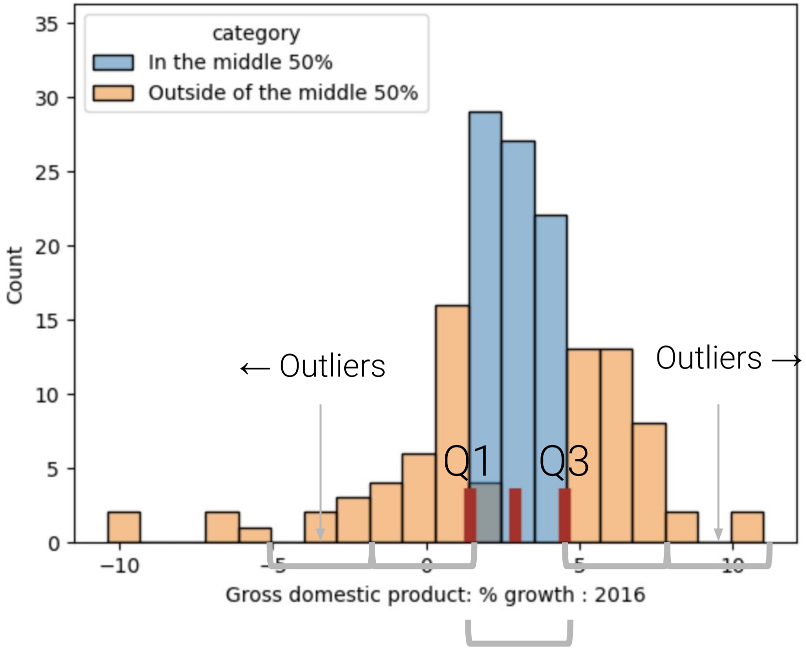
The interval **[Q1, Q3]** contains "middle 50%" of data.

Interquartile range (IQR) measures spread.

- IQR = Q3 - Q1.

Definition of an **outlier**:

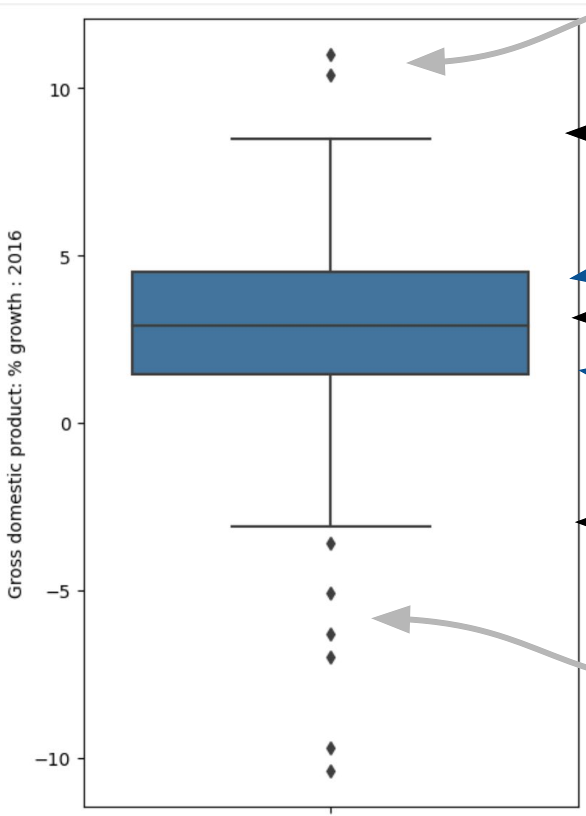
- Values > Q3 + 1.5 * IQR
- Values < Q1 - 1.5 * IQR



The length of this region is the IQR

Why 1.5? "I was present when Dr. John Tukey gave a talk, and he was asked... why 1.5 was chosen [for the IQR multiplier].... what he [Tukey] said was [that] 2 was too big... and 1 was too small... and 1.5 was just right."

Box Plots



Outliers (> Q3 + 1.5*IQR)

Whisker: Largest actual value that is not an outlier.

3rd quartile (75th percentile)

2nd quartile (median)

1st quartile (25th percentile)

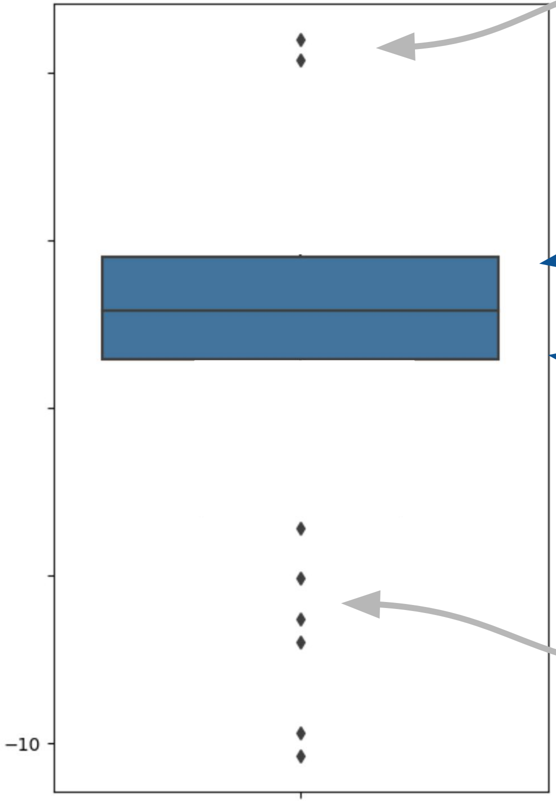
Whisker: Smallest actual value that is not an outlier.

Outliers (< Q1 - 1.5*IQR)

```
sns.boxplot(data=wb, y="Gross domestic product: % growth : 2016")
```

What's the minimum possible length of a boxplot whisker?

Hypothetical Slido boxplot



Outliers (> Q3 + 1.5*IQR)

No Upper Whisker! All values above Q3 are outliers.

Q3

Q1

No Lower Whisker! All values below Q1 are outliers.

Outliers (< Q1 - 1.5*IQR)

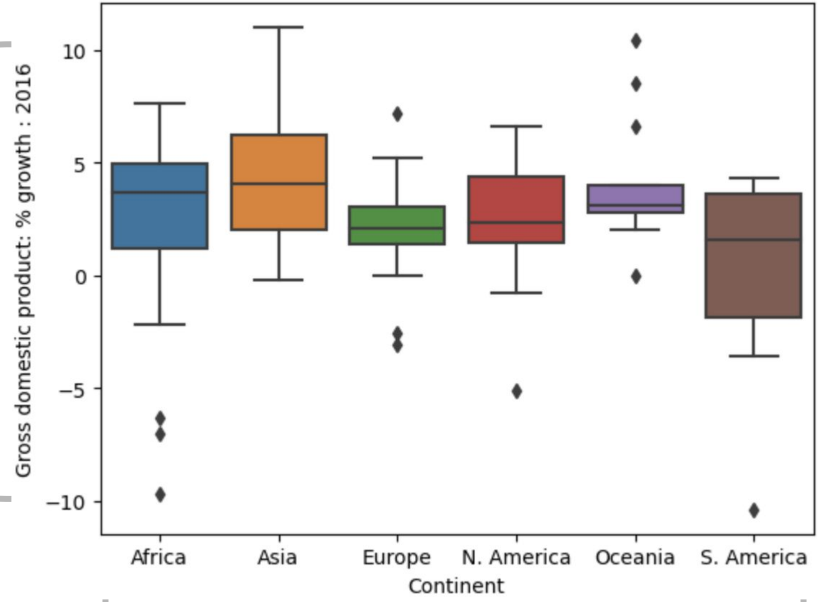
Side-by-side Box Plots

3123588

What if we wanted to incorporate a ***qualitative*** variable as well? For example, compare the distribution of a quantitative continuous variable *across* different qualitative **categories**.

```
sns.boxplot(data=wb, x="Continent", y="Gross domestic product: % growth : 2016");
```

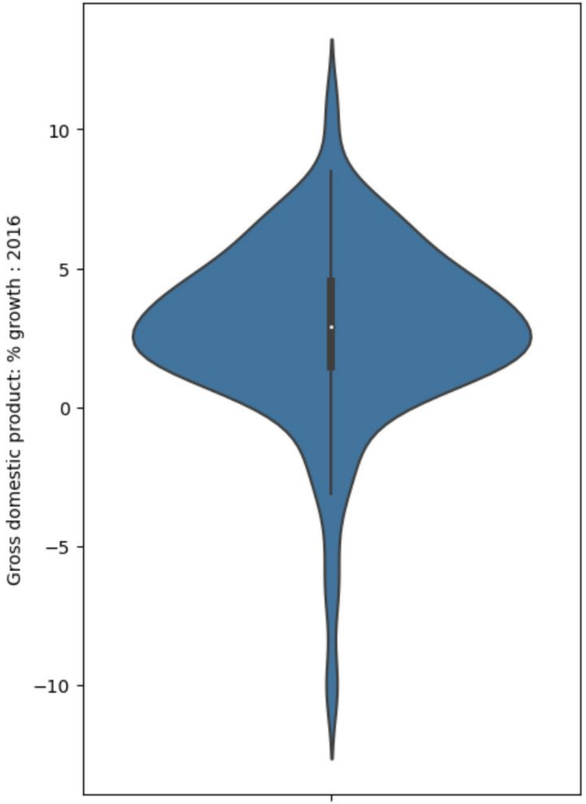
GDP growth:
quantitative continuous



Continent: qualitative nominal

Note: Color has no meaning here! `seaborn` defaults to different colors.

Violin Plots



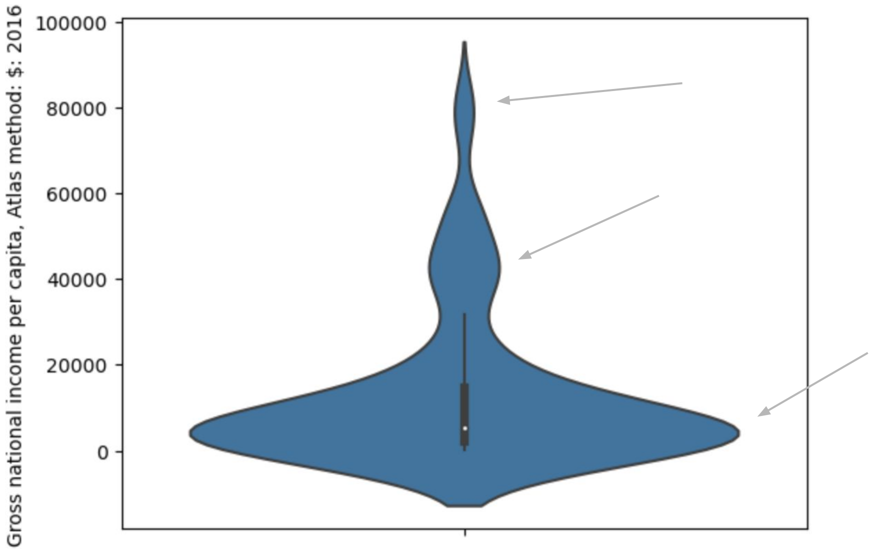
Violin plots are just box plots with smoothed density curves.

- The width indicates the density of points.
- Q1, median, Q3, and "whiskers" are still present – look closely!

```
sns.violinplot(data=wb, y="Gross domestic product: % growth : 2016")
```

Which of the following can show the modality of a distribution?

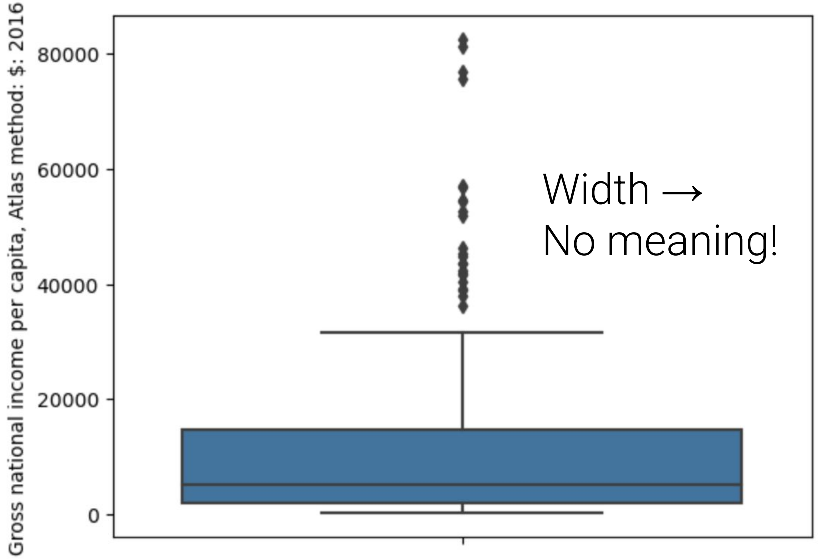
Box plots cannot show modality

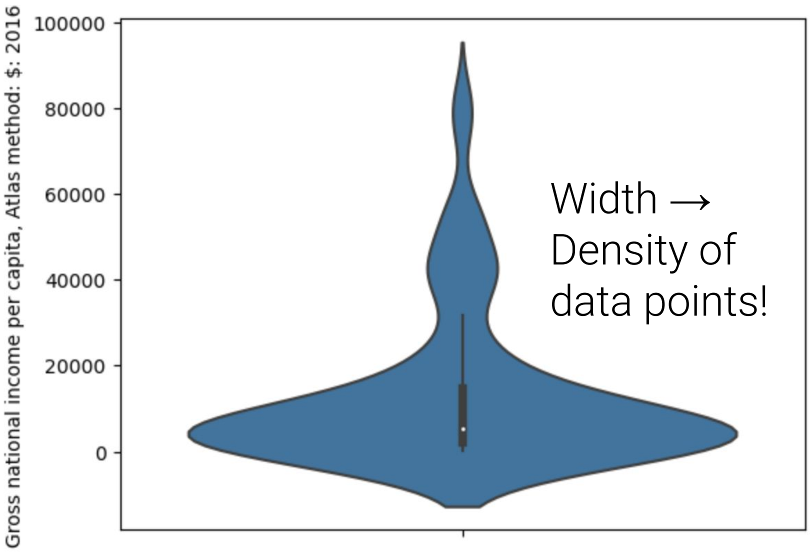


This distribution is (arguably) multimodal.

You cannot know this based on just the boxplot.

Comparing box plots and Violin Plots






```
sns.boxplot(data=df, y="y_variable");
```

[Documentation]

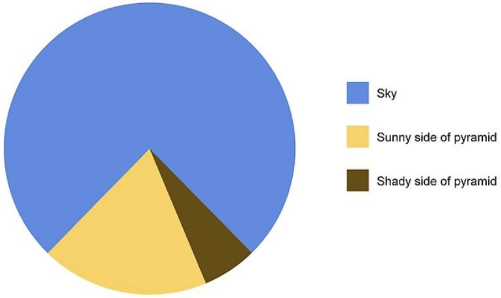
```
sns.violinplot(data=df, y ="y_variable");
```

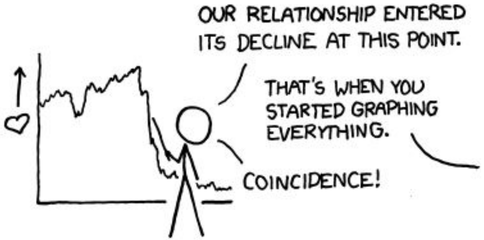
[Documentation]

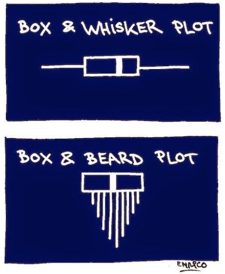
2-min stretch!

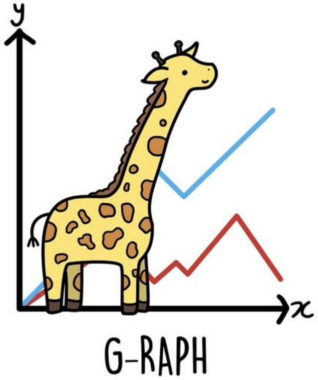
data-to-viz.com → A beautiful resource for exploring plots!

3123588









Kernel Density Estimation (KDE)

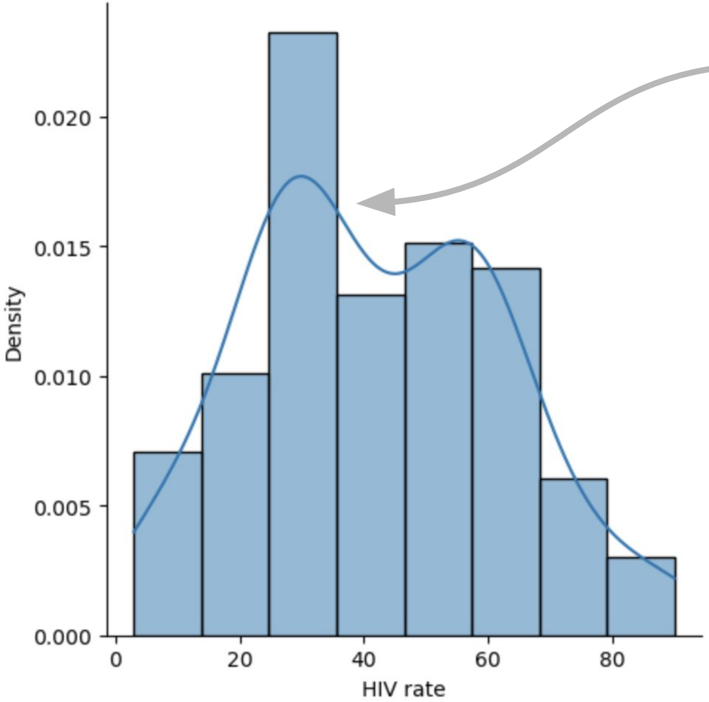
Lecture 7, Data 100 Summer 2025

- **Visualization**
  - Goals of visualization
  - Visualizing distributions
  - **Kernel density estimation (KDE)**

Kernel Density Estimation (KDE): Intuition

3123588

Sometimes, we want to identify *general* trends of a distribution, rather than focus on details. Smoothing a distribution helps **generalize** the structure of the data and reduce **noise**.



A KDE curve and histogram for the **same** data

Idea: Approximate the **data-generating distribution**.

1. Assign an "error range" (**kernel**) to each data point, to account for randomness in sampling.
2. **Sum** up the kernels across all data points.
3. **Scale** the resulting distribution to have area=1.

Kernel Density Estimation (KDE): Process

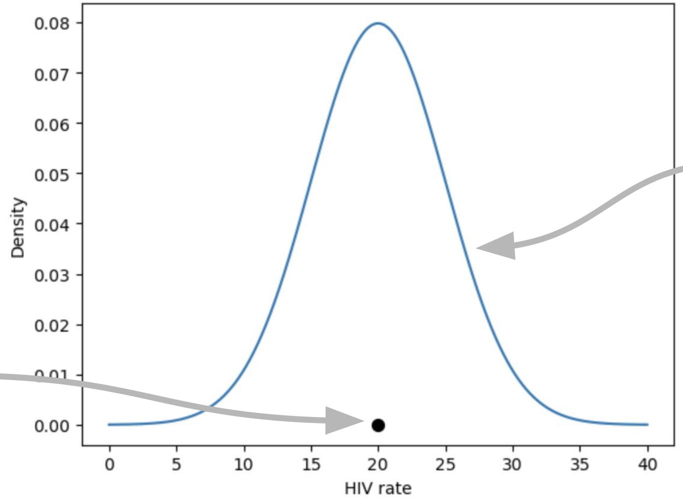
3123588

Idea: Approximate the data-generating distribution.

1. Place a **kernel** at each data point.
2. **Normalize** kernels so that total area = 1.
3. **Sum** all normalized kernels.

A **kernel** is a function that tries to capture the randomness of our sampled data.

A **datapoint** in our dataset



The **kernel** models the probability of us sampling that datapoint.

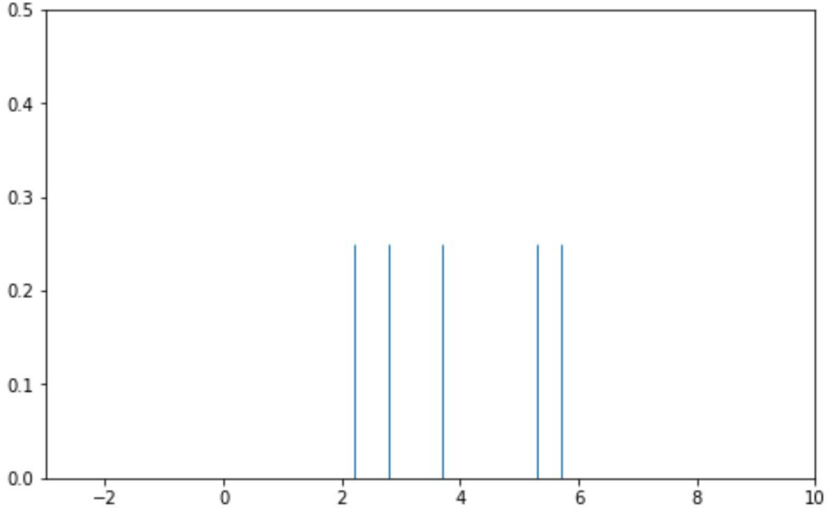
Area below integrates to 1.

Step 1 – Place a Kernel at Each Data Point

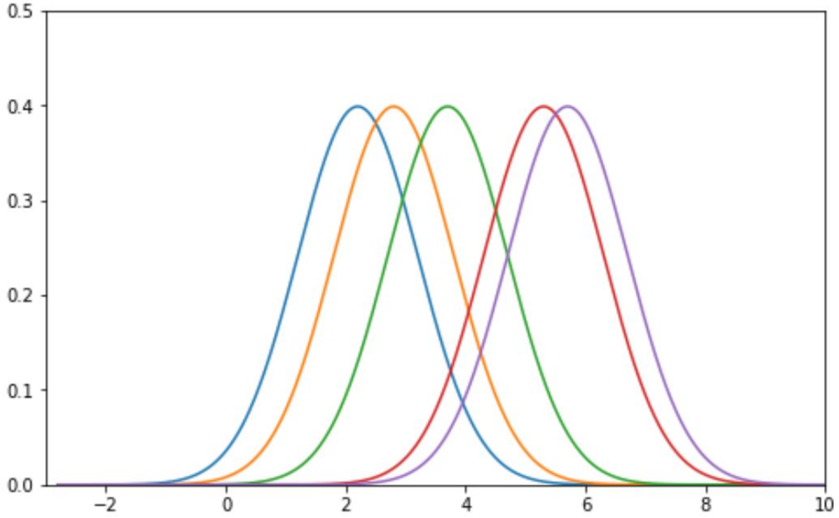
3123588

Consider a fake dataset with just five datapoints.

- Place a **Gaussian (i.e., normal) kernel** with **bandwidth** of **α = 1 ("alpha=1")**.
- We will precisely define both the **Gaussian kernel** and **bandwidth** in a few slides.



Each line represents a datapoint in the dataset (e.g., one country's HIV rate). This is a **rug plot**.



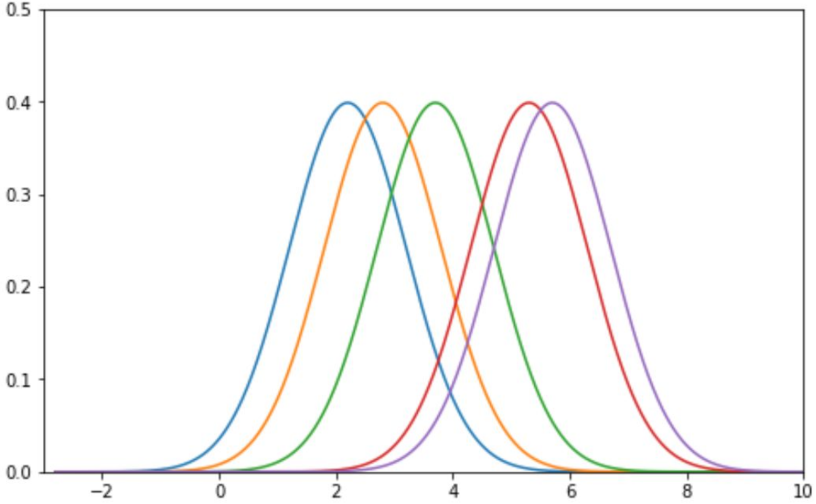
Place a kernel on top of each datapoint.

Step 2 – Normalize Kernels

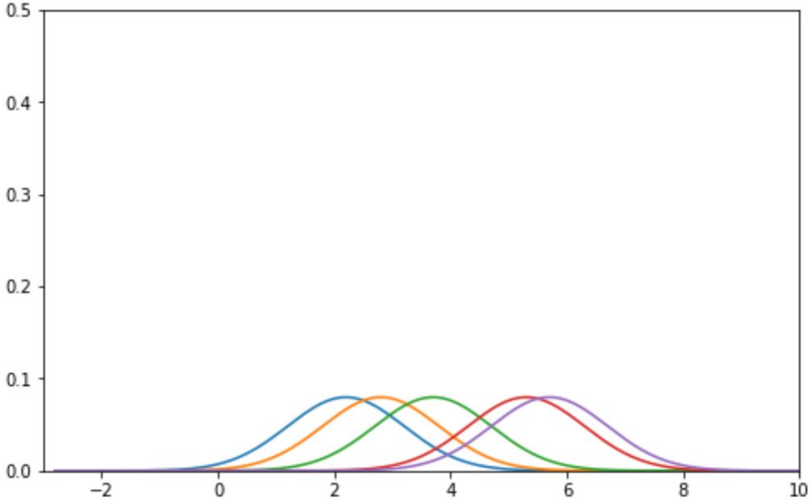
3123588

In Step 3, we will sum the kernels to produce a distribution.

- We want the result to be a valid **probability** distribution that has **area=1**.
- We have **n=5** different kernels, each with an area 1.
- So, in Step 2 we **normalize** by multiplying each kernel by **⅕**.



Each kernel has area 1.



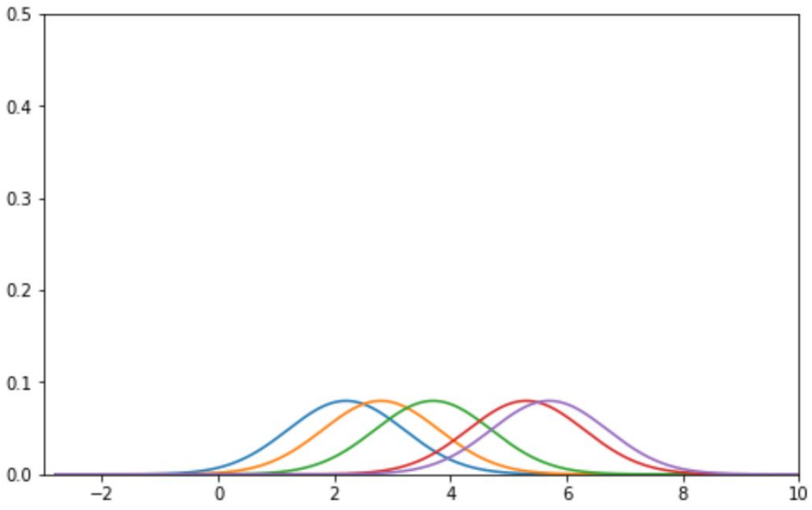
Each normalized kernel has density ⅕.

Note: It would be equivalent to first sum the kernels and then normalize the result.

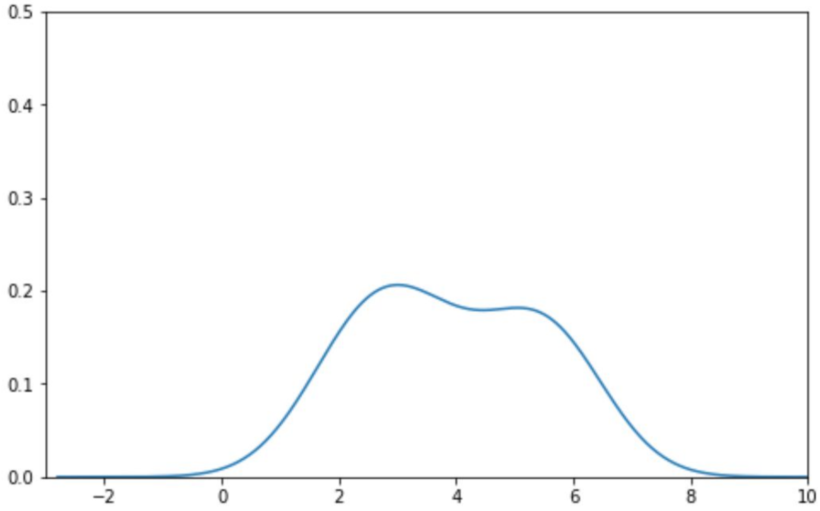
Step 3 – Sum the Normalized Kernels

3123588

At each point in the distribution, add up the values of all kernels. This gives us a smooth curve with area=1 → an approximation of a probability distribution!



Sum the five normalized curves together.



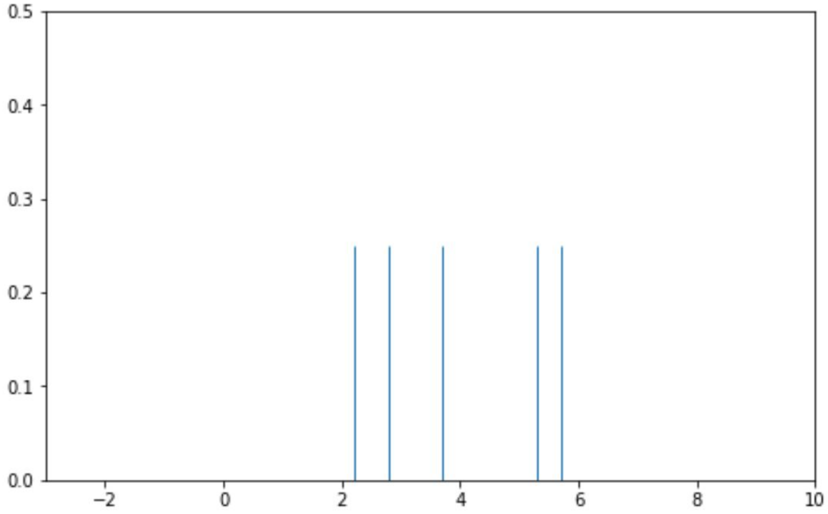
The final KDE curve.

Producing a KDE plot with `seaborn`

3123588

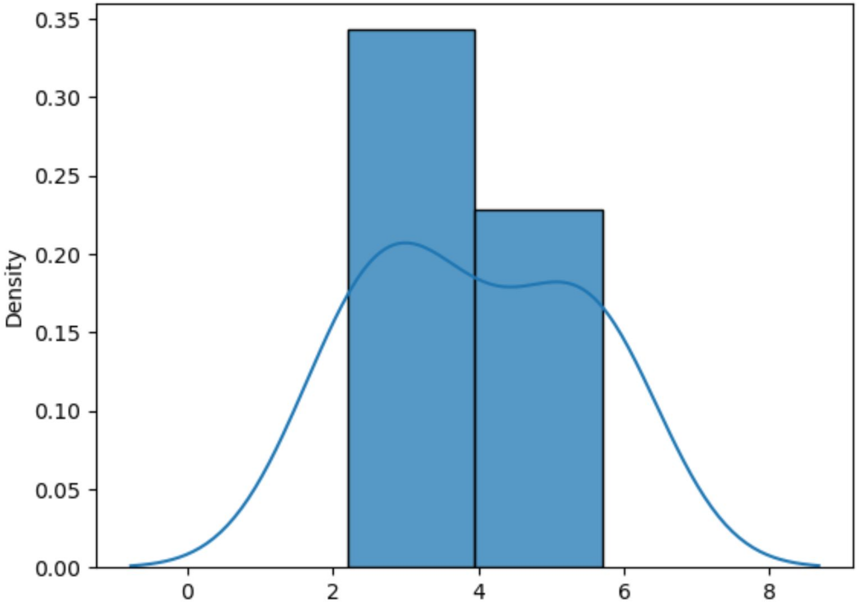
Rug plot of the original data.

```
sns.rugplot(points)
```



Each line represents a datapoint in the dataset (e.g., one country's HIV rate).

```
sns.kdeplot(points, bw_method=0.65)
sns.histplot(points,
             stat="density",
             bins=2);
```

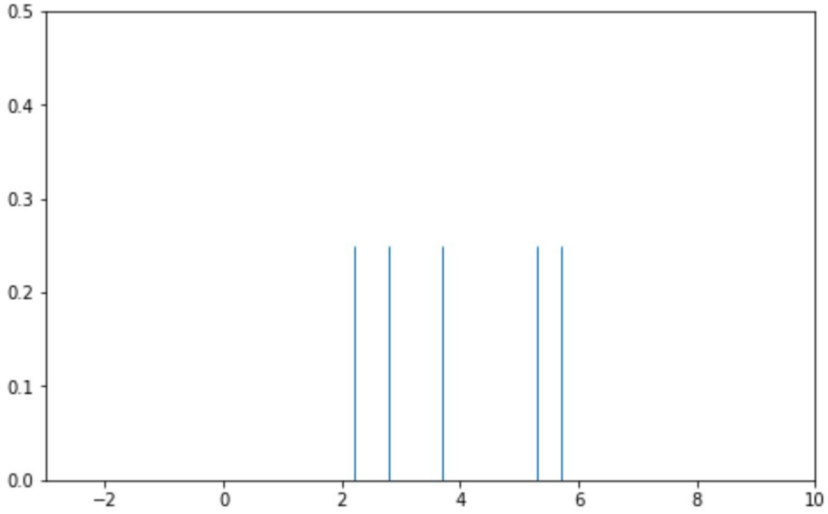


The density at each x is the sum of the height of all 5 normalized kernels at that x.

Producing a KDE plot with `seaborn` - Alternate Method

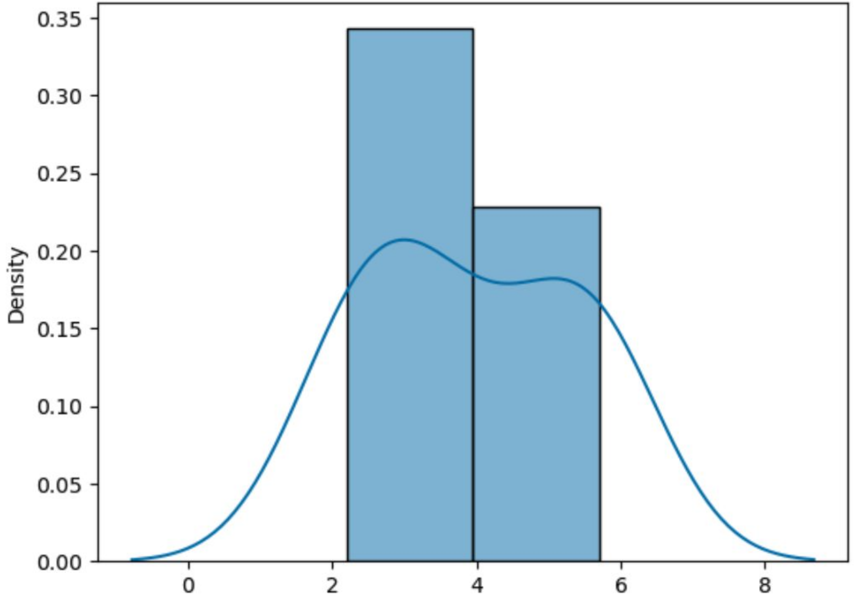
Rug plot of the original data.

```
sns.rugplot(points)
```



Each line represents a datapoint in the dataset (e.g., one country's HIV rate).

```
sns.histplot(points, bins=2, kde=True,
             stat="density",
             kde_kws=dict(cut=3,
                          bw_method=0.65))
```



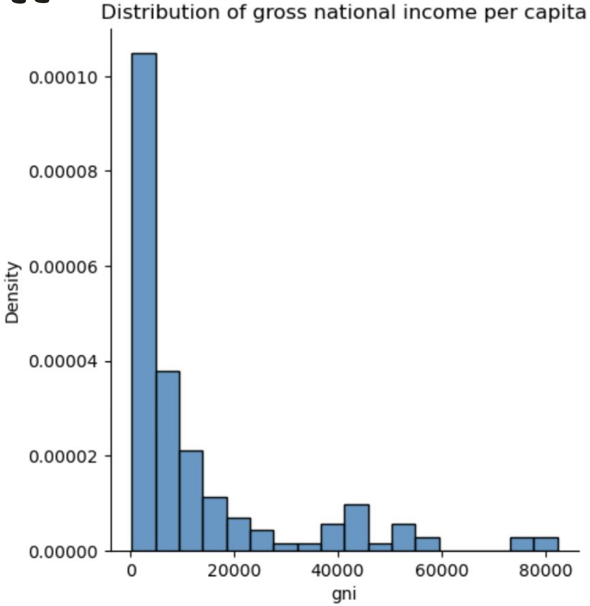
The density at each x is the sum of the height of all 5 normalized kernels at that x.

3123588

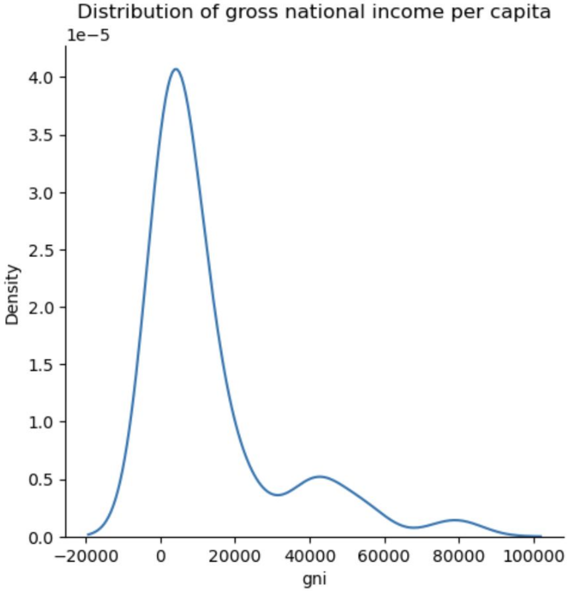
displot

`displot` is a wrapper for **`histplot`**, **`kdeplot`**, and **`ecdfplot`** to plot distributions.

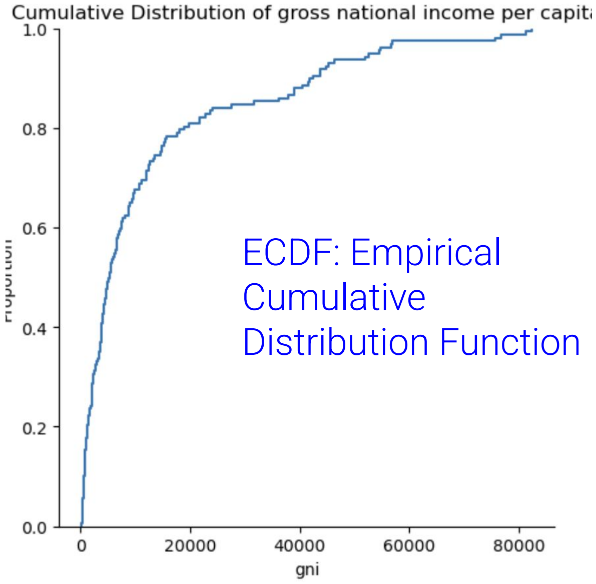
```python
sns.displot(data=wb,
            x="gni",
            kind="hist",
            stat="density")
```




```python
sns.displot(data=wb,
            x="gni",
            kind="kde")
```

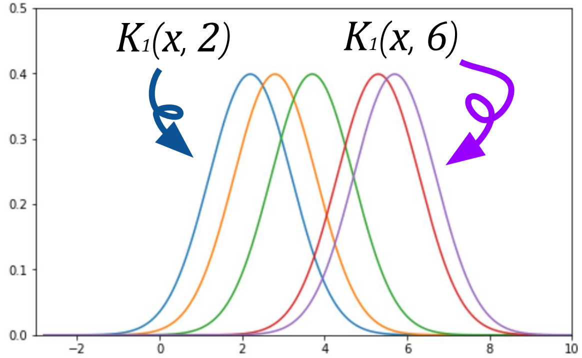



```python
sns.displot(data=wb,
            x="gni",
            kind="ecdf")
```

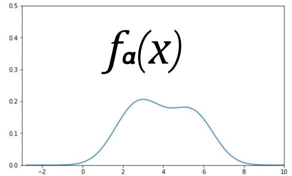


Summary of KDE

$$\overset{\boxed{1}}{f_\alpha(x)} = \overset{\boxed{3}}{\frac{1}{n}} \overset{\boxed{4}}{\sum_{i=1}^{n}} \overset{\boxed{2}}{K_\alpha(x, x_i)}$$



A general "KDE formula" function is given above.



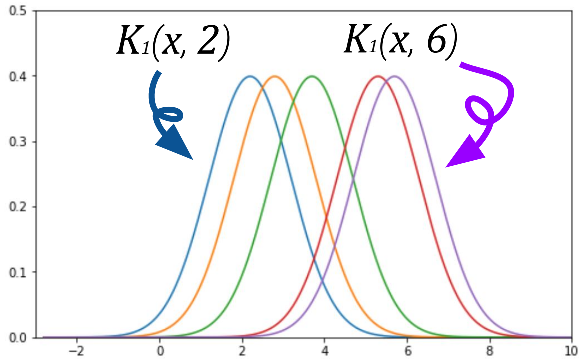
$\boxed{1}$ $f_\alpha(x)$ is the end result → **Height** of the final KDE curve at any x-value.

$\boxed{2}$ $K_\alpha(x, x_i)$ is the height of the chosen **kernel** function for observation ***i*,** at any x-value. The **center** of the kernel is at the specific x-value of observation ***i*.**

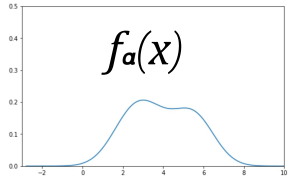
Summary of KDE

3123588

$$\overset{\boxed{1}}{f_\alpha(x)} = \overset{\boxed{3}}{\frac{1}{n}} \overset{\boxed{4}}{\sum_{i=1}^{n}} \overset{\boxed{2}}{K_\alpha(x, x_i)}$$



A general “KDE formula” function is given above.



$\boxed{1}$ $f_\alpha(x)$ is the end result → **Height** of the final KDE curve at any x-value.

$\boxed{2}$ $K_\alpha(x, x_i)$ is the height of the chosen **kernel** function for observation ***i***, at any x-value. The **center** of the kernel is at the specific x-value of observation ***i***.

$\boxed{3}$ n is the # of data points.

- We multiply by $1/n$ to normalize the kernels so that total area = 1.

$\boxed{4}$ Each x_i ($x_1, x_2, \ldots, x_n$) represents an observed data point. We sum the n kernels (one for each datapoint) to create the final KDE curve.

α is the **bandwidth** or **smoothing parameter**.

Kernels

3123588

A **kernel** is a valid density function, meaning:

- It must be non-negative for all inputs.
- It must integrate to 1 (area under curve = 1).

The most common is the **Gaussian kernel**.

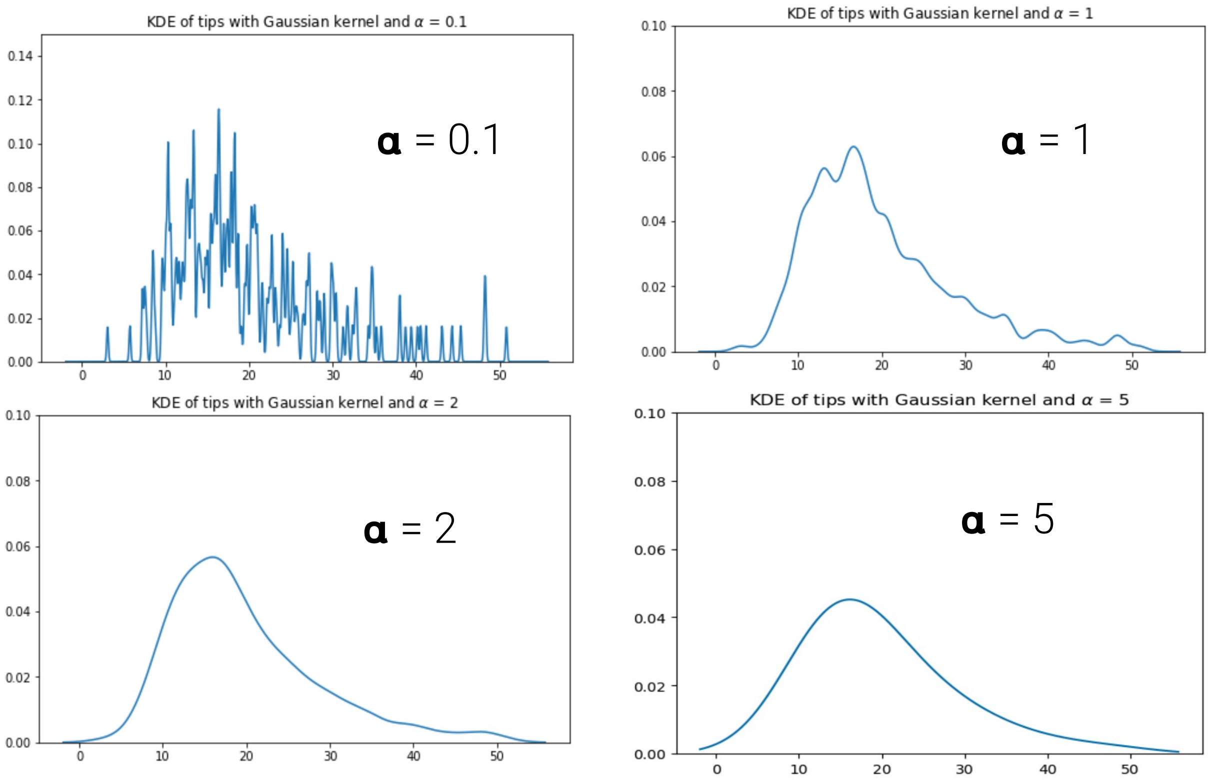
- Gaussian = Normal distribution = bell curve.
- Here, x represents any input, and x_i represents the ith observed datapoint.
- Each kernel is **centered** on an observed x_i value.
- **α** is the **bandwidth parameter**. It controls the smoothness of our KDE. Here, it is also the **standard deviation** of the Gaussian.

$$K_\alpha(x, x_i) = \frac{1}{\sqrt{2\pi\alpha^2}} e^{-\frac{(x-x_i)^2}{2\alpha^2}}$$

Don't memorize this formula! Understand its shape and how the bandwidth parameter **α** smoothes the KDE.

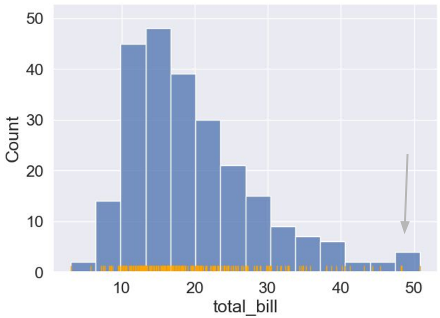
Effect of Bandwidth on KDEs

3123588



Bandwidth is analogous to the width of each **bin** in a histogram.

- As **α** increases, the KDE becomes smoother.
- **α** too small → Noisy
- **α** too big → Hides important distributional info (e.g., multimodality).



Other Kernels: Boxcar

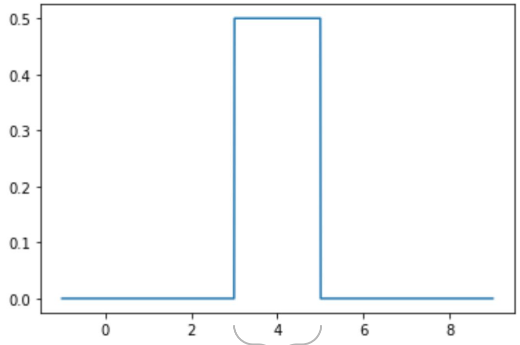
3123588

Another example kernel: the **boxcar kernel**.

- **Uniform** (i.e., constant) density to points within a "window" of the observation, and 0 elsewhere.
- Resembles a histogram… *sort of*.

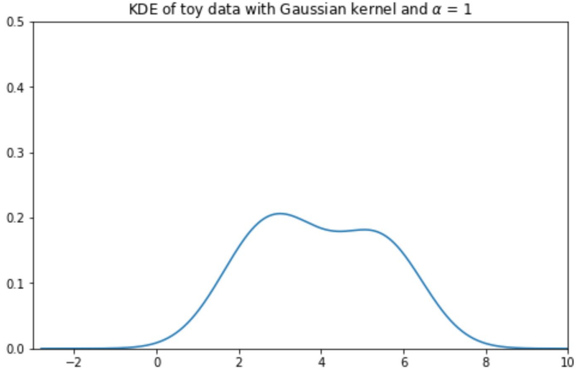
$$K_\alpha(x, x_i) = \begin{cases} \frac{1}{\alpha}, & |x - x_i| \le \frac{\alpha}{2} \\ 0, & \text{else} \end{cases}$$

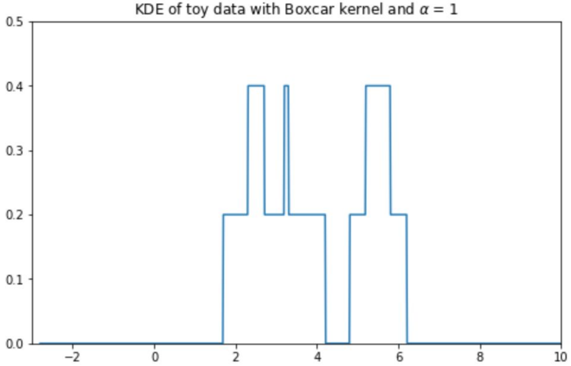
- Not of any practical use! Presented as a simple theoretical alternative to test KDE knowledge.



α = 2

A boxcar kernel centered on x_i = 4 with α = 2. Note: Area=1





Which of the following are valid kernel density plots?

Have a Normal Day!

3123588

3123588

LECTURE 7

Visualization I

Content credit: Acknowledgments