Data Visualization

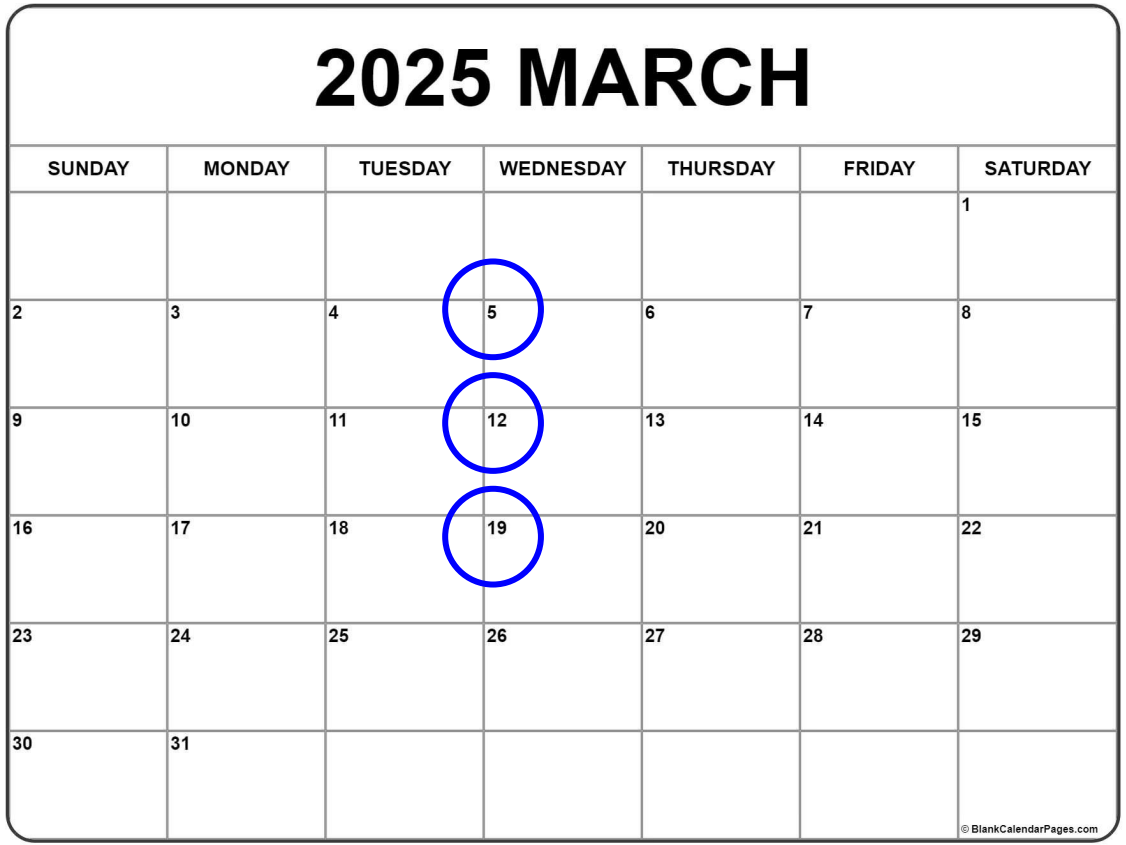
CPE 232: Data Models

Dr. Sansiri Tarnpradab

Department of Computer Engineering, KMUTT

Announcement
(INTL)

- Quiz 2 *(Today, for 10 mins)*
- Midterm *(5/3)*
- Visiting Professor *(12/3)*
- Proposal *(19/3)*

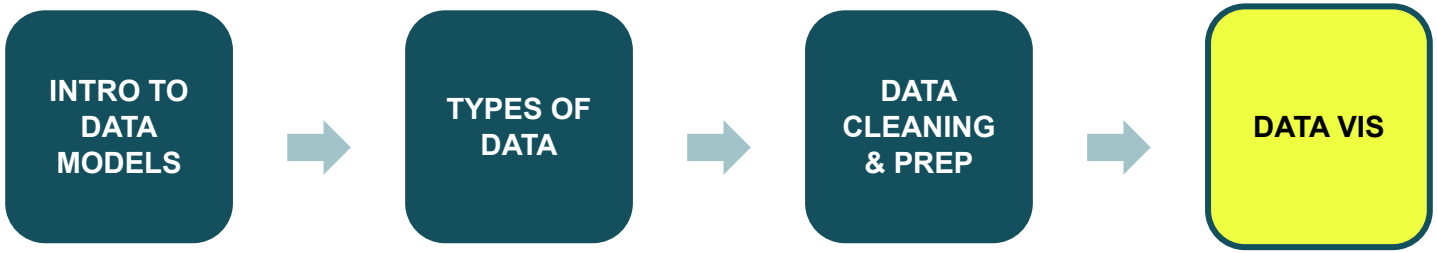


Proposal
(INTL)

# Members Per Group		Availability (# groups)
KM (KMUTT)	**EX** (EXCHANGE)	
6	1	9
6	2	5

- As a group
  - 7-8 members per group
  - ∴ Total groups = 14
- **No more than 3 pages** submitted on LEB2
  - Introduction (What)
  - Objectives (Why)
- Be able to demonstrate how you apply Data Models concepts
- Due on 19/3 23.59 PM (Two weeks after our class midterm)

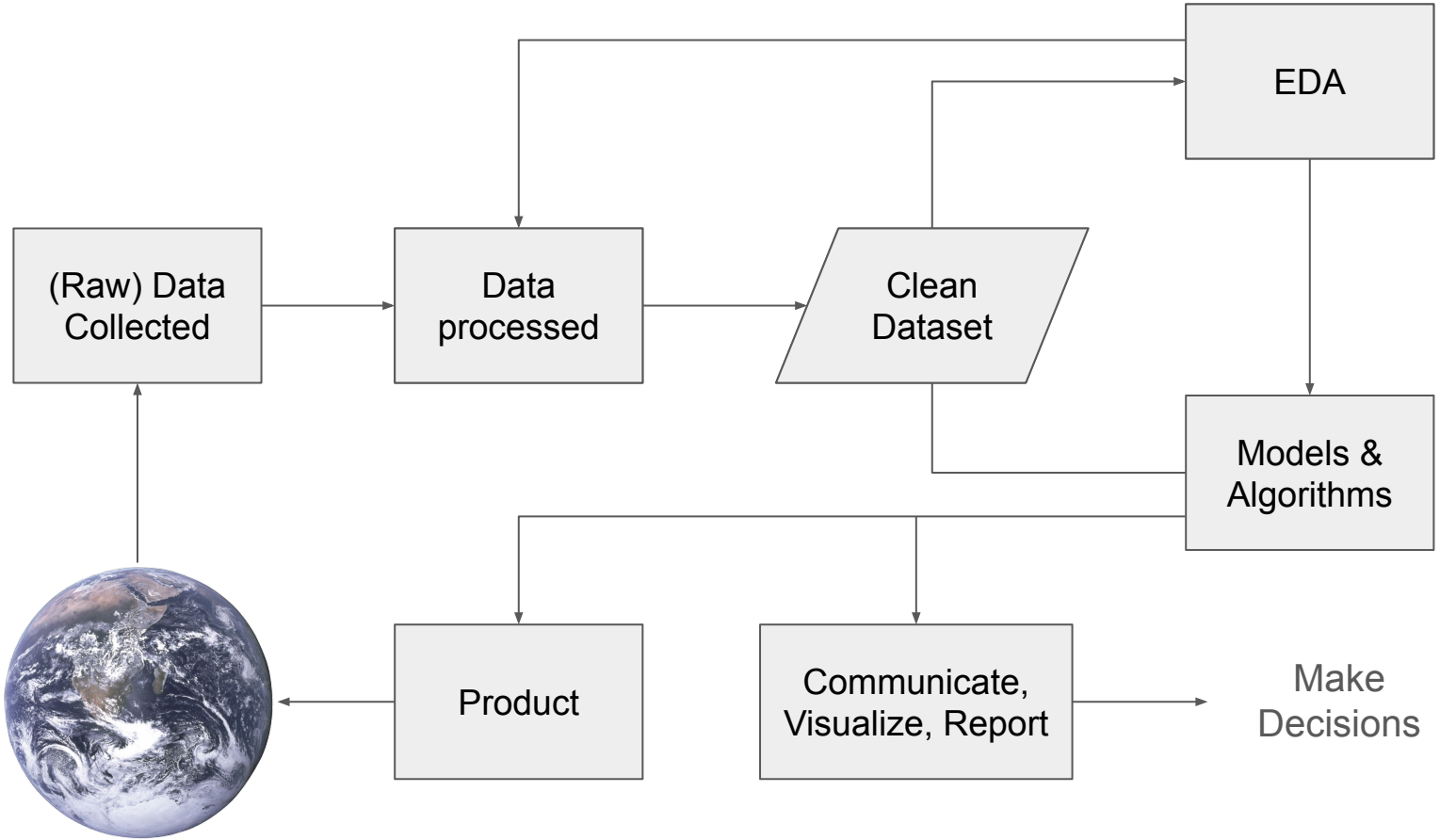
Review


INTRO TO DATA MODELS
TYPES OF DATA
DATA CLEANING & PREP
DATA VIS

Outline

- Intro to Data Visualization
- Effective Visuals
- Types of Charts
- Dashboard
- Intro to Python libraries for Data Visualization
  - Static
  - Interactive

Data Science Workflow



1 Data Collection

2 Data Preparation

3 Data Visualization

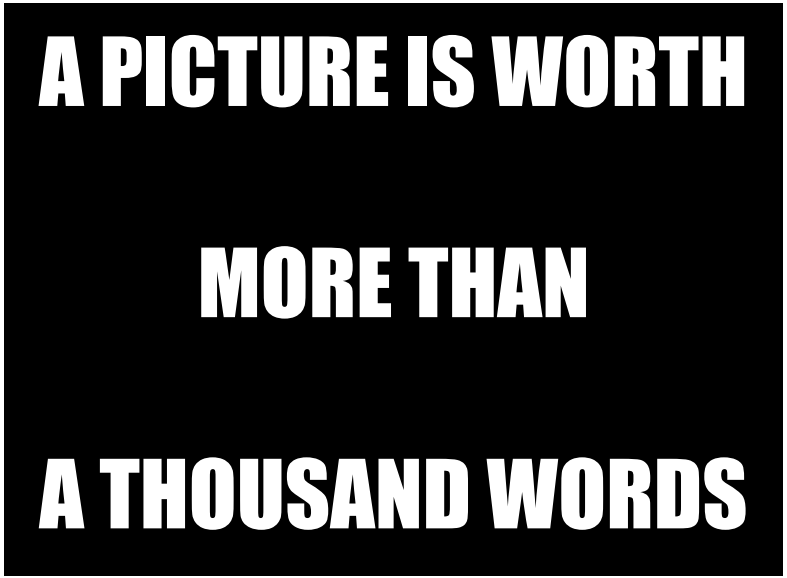
4 Data Analysis

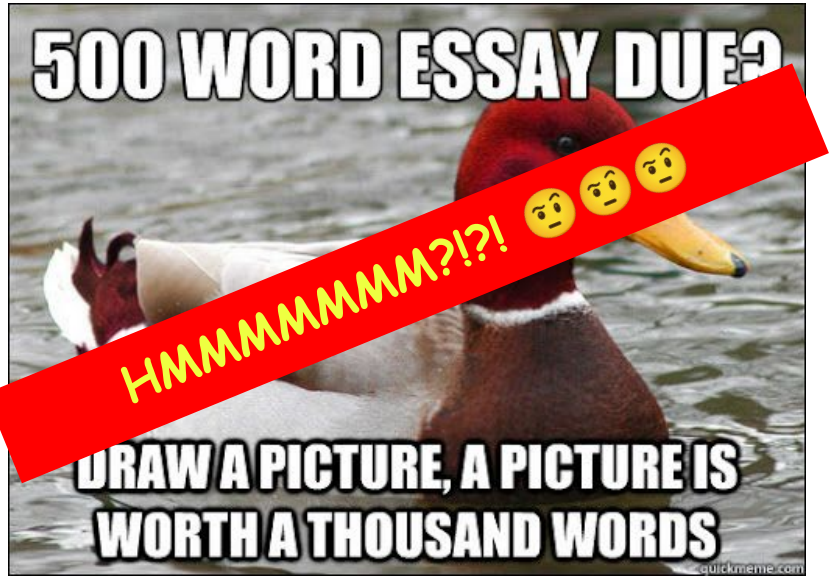
5 Data Storytelling

Ref: https://www.eddeb.com/blog/10-lessons-i-learned-being-a-data-analyst-at-quintoandar/

Data Visualization

Technique to present data in a pictorial/graphical format.





Ref: http://www.quickmeme.com/meme/3u8hy9

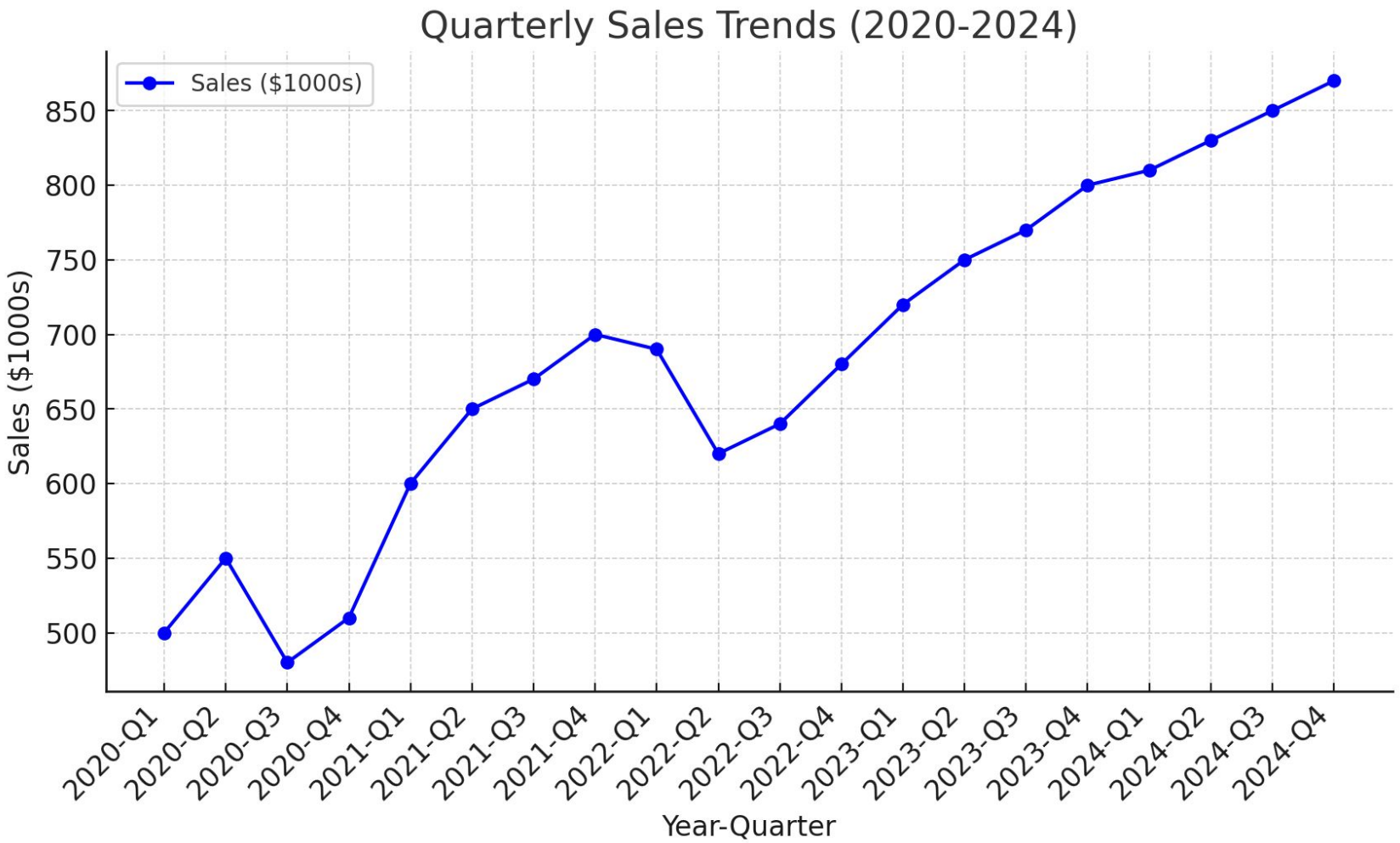
Significance & Benefits of Data Visualization

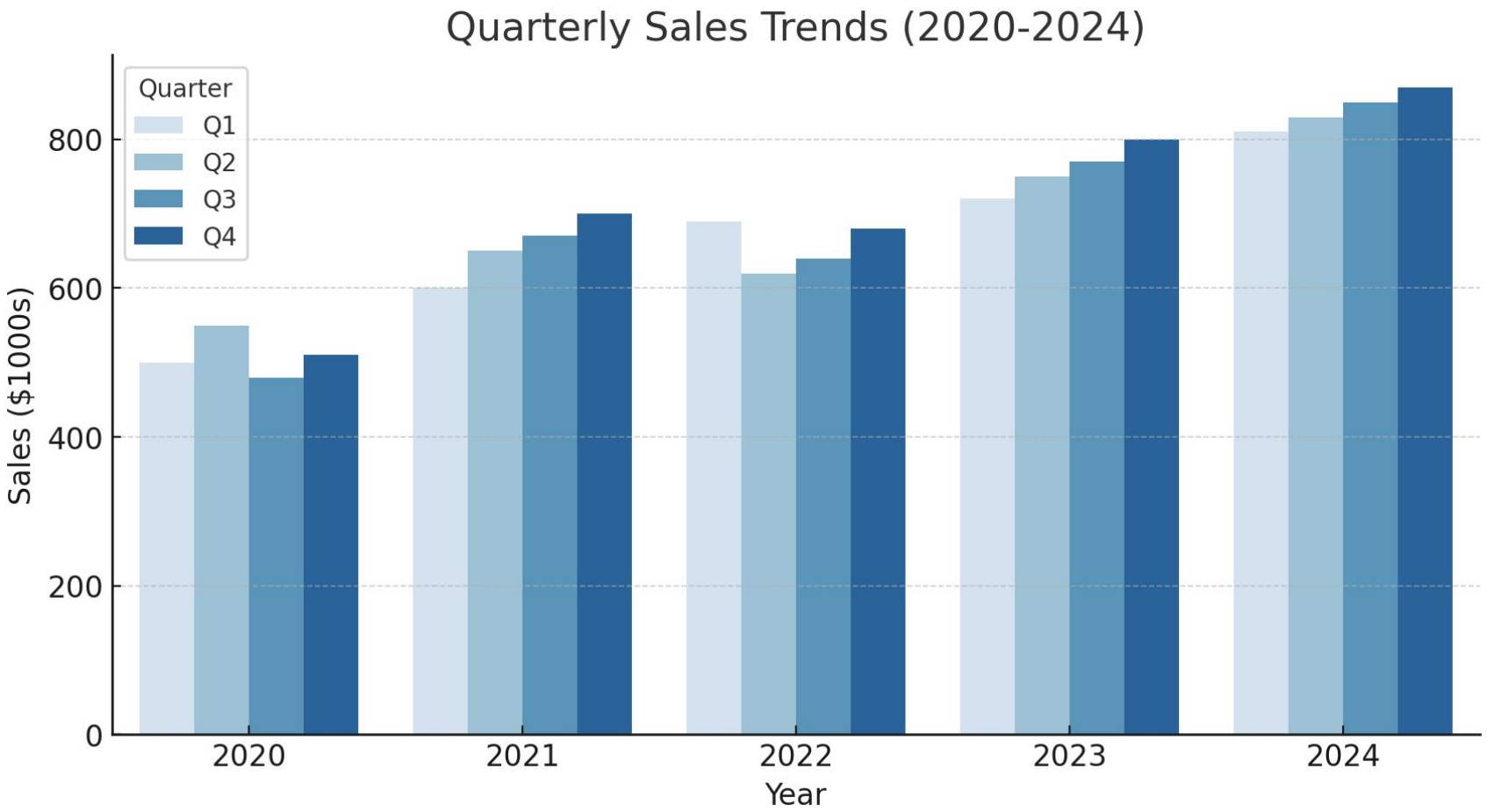
- Simplify complex data
- Identify patterns, trends, structure, irregularities, relationships among data
- Help find interesting regions and suitable parameters for quantitative analysis
- Gain insights into an information space by mapping data onto graphical primitives
- Provide qualitative overview of large data sets
- Comparative analysis
- Support storytelling
- Enhance engagement
- Faster decision making

Example #1

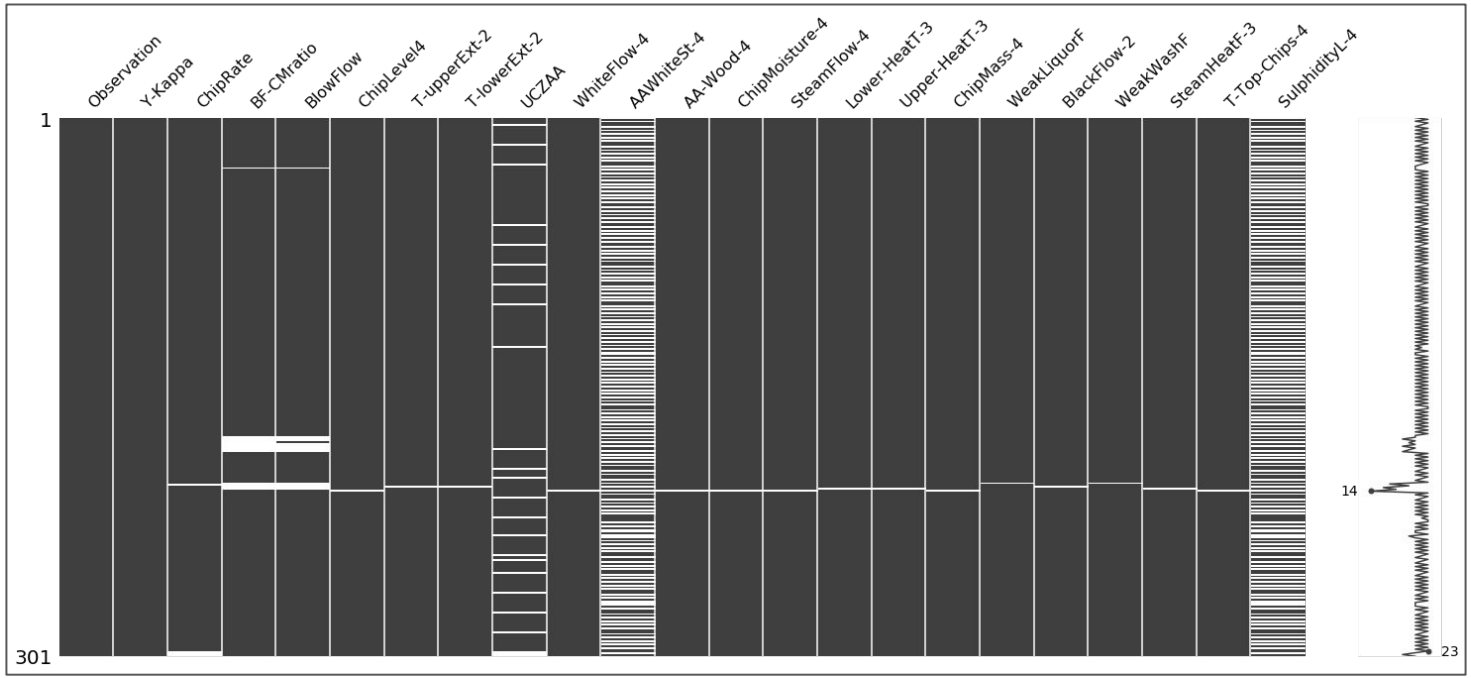
Imagine you are presenting quarterly sales data for a company over the past 5 years.

“In 2020, sales were around $500,000 in Q1, increased to $550,000 in Q2, dropped to $480,000 in Q3, and slightly recovered to $510,000 in Q4. In 2021, sales grew steadily, reaching $700,000 by Q4. However, in 2022, there was a sharp decline in Q2 due to market conditions, dropping sales back to $620,000 before they rose again…”


Quarterly Sales Trends (2020-2024)
Sales ($1000s)
Sales ($1000s)
850
800
750
700
650
600
550
500
2020-Q1
2020-Q2
2020-Q3
2020-Q4
2021-Q1
2021-Q2
2021-Q3
2021-Q4
2022-Q1
2022-Q2
2022-Q3
2022-Q4
2023-Q1
2023-Q2
2023-Q3
2023-Q4
2024-Q1
2024-Q2
2024-Q3
2024-Q4
Year-Quarter


Quarterly Sales Trends (2020-2024)
Quarter
Q1
Q2
Q3
Q4
Sales ($1000s)
0
200
400
600
800
2020
2021
2022
2023
2024
Year

Example #2

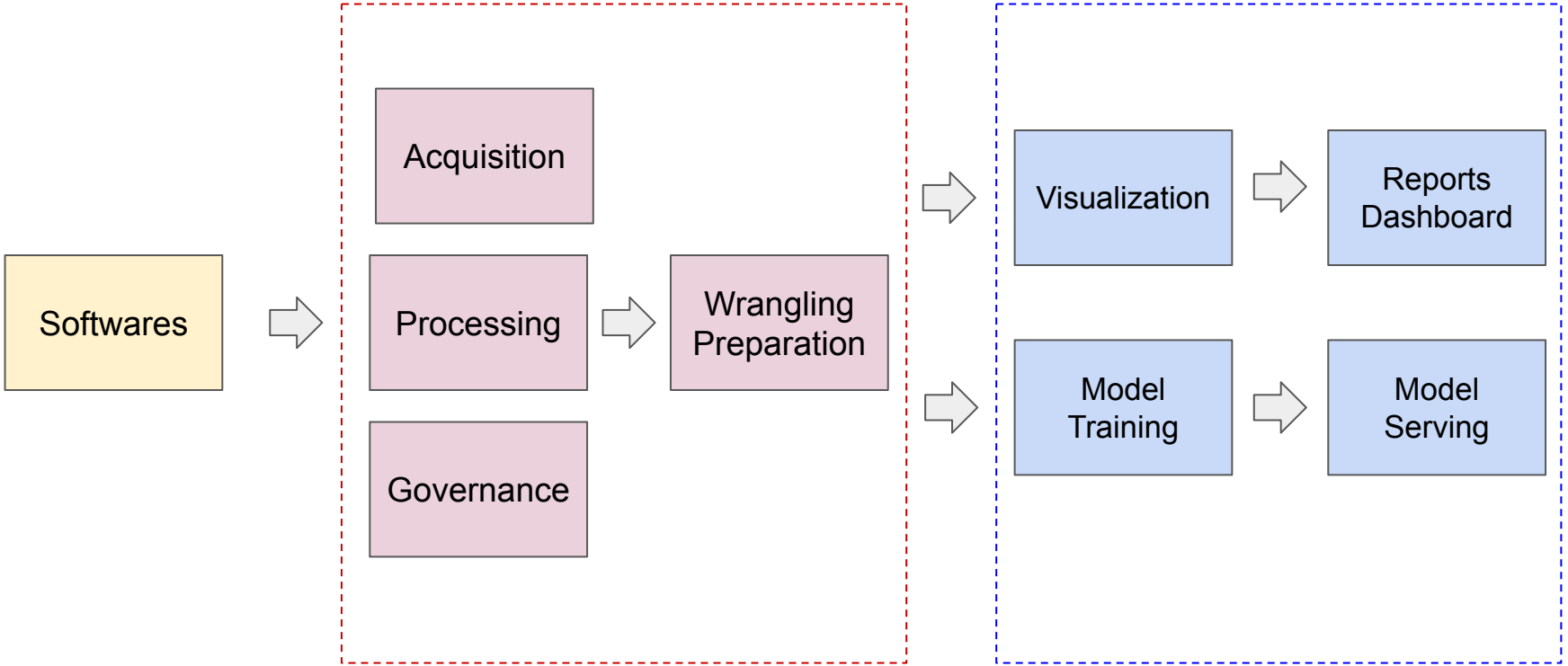


Ref: https://www.geeksforgeeks.org/python-visualize-missing-values-nan-values-using-missingno-library/

Example #3

Ref: https://www.reddit.com/r/mathmemes/comments/18vlkbn/this_graph_made_my_eyes_melt_from_data/?rdt=60716

Revisiting Different Roles



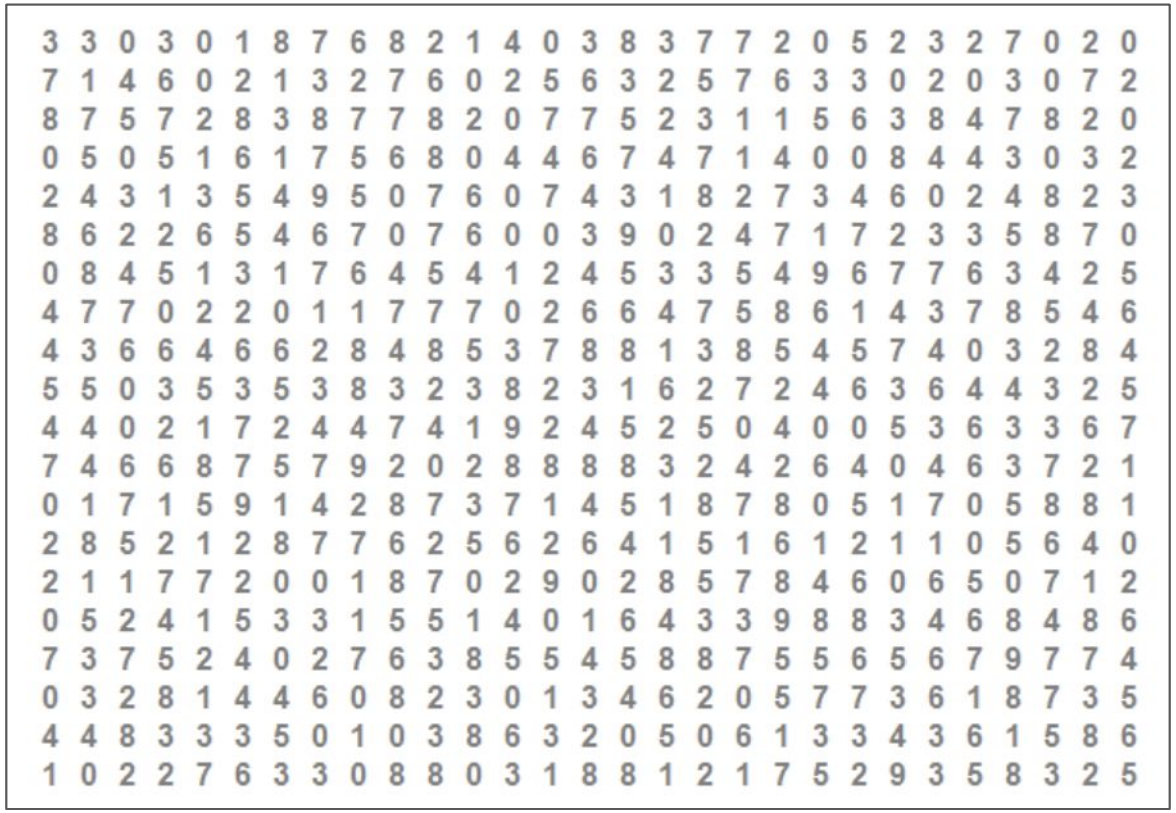
Roles of Data Analyst

- Identify data sources that are suitable to address business questions
- Identify methods to address business questions
- Explore data and identify useful insights
- Create meaningful reports and visualizations

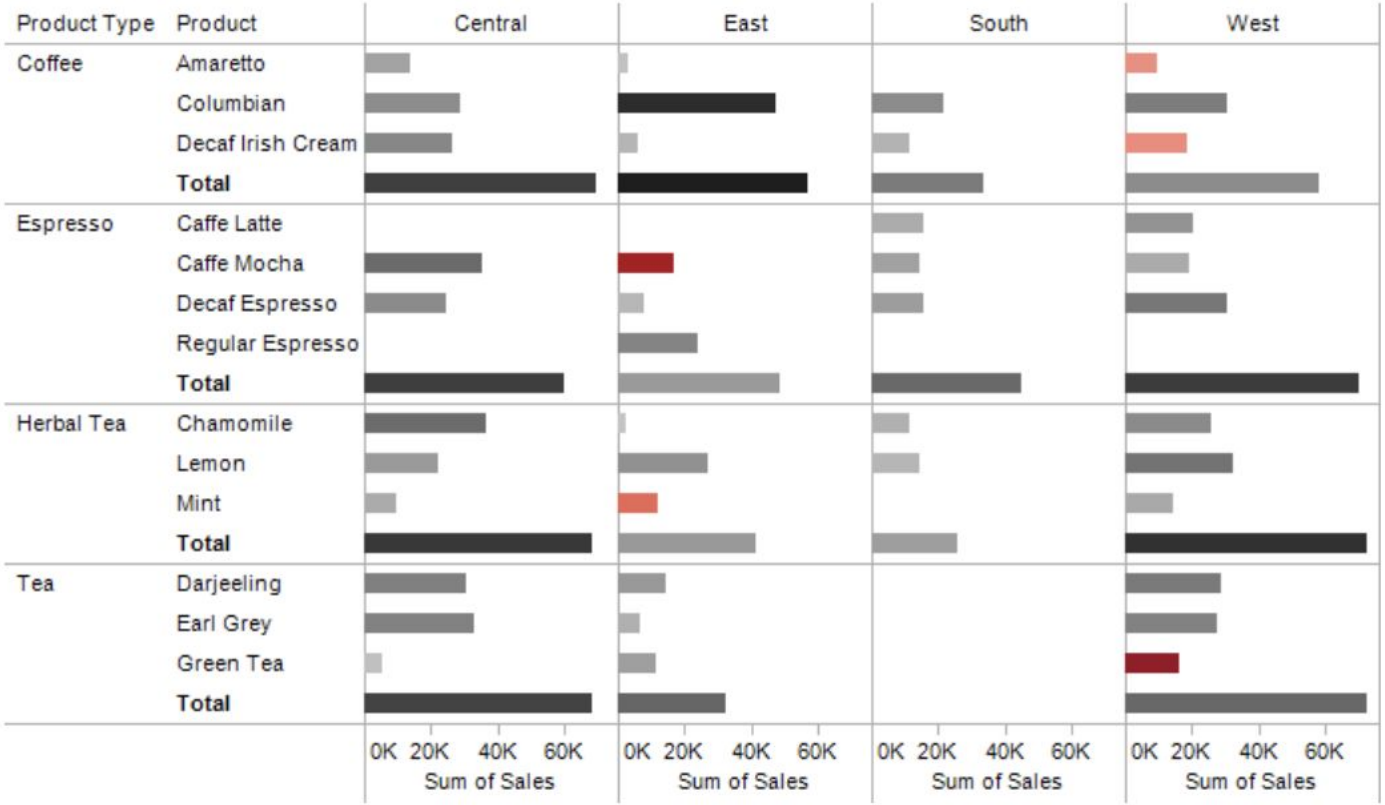
Things to Consider

- ❑ Clarity 🔎
- ❑ Accuracy ✅
- ❑ Efficiency 💯

Efficiency: Human Perception



		Central		East		South		West	
Product Type	Product	Sum of Profit	Sum of Sales	Sum of Profit	Sum of Sales	Sum of Profit	Sum of Sales	Sum of Profit	Sum of Sales
Coffee	Amaretto	$5,105	$14,011	$1,009	$2,993			($1,225)	$9,265
	Columbian	$8,528	$28,913	$27,253	$47,386	$8,767	$21,664	$11,253	$30,357
	Decaf Irish Cream	$9,632	$26,155	$2,727	$6,261	$2,933	$11,592	($1,305)	$18,235
	Total	$23,265	$69,080	$30,989	$56,640	$11,700	$33,256	$8,724	$57,856
Espresso	Caffe Latte					$3,872	$15,442	$7,502	$20,458
	Caffe Mocha	$14,640	$35,218	($6,230)	$16,646	$5,201	$14,163	$4,064	$18,876
	Decaf Espresso	$8,860	$24,485	$2,410	$7,722	$5,930	$15,384	$12,302	$30,578
	Regular Espresso			$10,062	$24,036				
	Total	$23,500	$59,703	$6,242	$48,405	$15,003	$44,989	$23,868	$69,911
Herbal Tea	Chamomile	$14,434	$36,570	$765	$2,194	$3,180	$11,186	$8,852	$25,632
	Lemon	$6,251	$21,978	$7,901	$27,176	$2,593	$14,497	$13,120	$32,274
	Mint	$4,069	$9,337	($2,242)	$11,992			$4,330	$14,380
	Total	$24,754	$67,885	$6,424	$41,362	$5,774	$25,683	$26,301	$72,285
Tea	Darjeeling	$10,772	$30,289	$6,497	$14,096			$11,780	$28,769
	Earl Grey	$10,331	$32,881	$3,405	$6,505			$10,425	$27,387
	Green Tea	$1,227	$5,211	$5,654	$11,571			($7,109)	$16,063
	Total	$22,330	$68,380	$15,557	$32,172			$15,097	$72,220


Product Type
Product
Central
East
South
West
Coffee
Amaretto
Columbian
Decaf Irish Cream
Total
Espresso
Caffe Latte
Caffe Mocha
Decaf Espresso
Regular Espresso
Total
Herbal Tea
Chamomile
Lemon
Mint
Total
Tea
Darjeeling
Earl Grey
Green Tea
Total
0K 20K 40K 60K
Sum of Sales

Efficiency: Bertin's Three Levels of Reading

Elementary

Single value

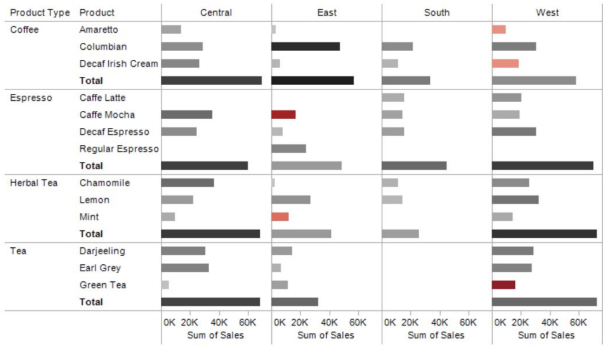
Intermediate

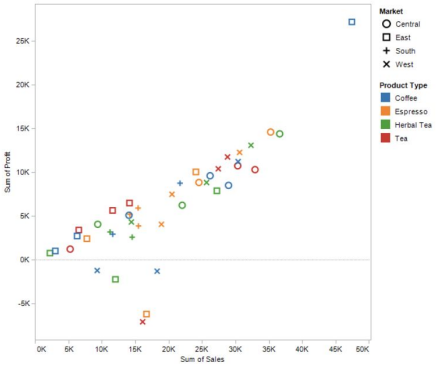
Relationship between values

Global

Relationship as a whole

		Central		East		South		West	
Product Type	Product	Sum of Profit	Sum of Sales	Sum of Profit	Sum of Sales	Sum of Profit	Sum of Sales	Sum of Profit	Sum of Sales
Coffee	Amaretto	$5,105	$14,011	$1,009	$2,993			($1,225)	$9,265
	Columbian	$8,528	$28,913	$27,253	$47,386	$8,767	$21,664	$11,253	$30,357
	Decaf Irish Cream	$9,632	$26,155	$2,727	$6,261	$2,933	$11,592	($1,305)	$18,235
	Total	$23,265	$69,080	$30,989	$56,640	$11,700	$33,256	$8,724	$57,856
Espresso	Caffe Latte					$3,872	$15,442	$7,502	$20,458
	Caffe Mocha	$14,640	$35,218	($6,230)	$16,646	$5,201	$14,163	$4,064	$18,876
	Decaf Espresso	$8,860	$24,485	$2,410	$7,722	$5,930	$15,384	$12,302	$30,578
	Regular Espresso			$10,062	$24,036				
	Total	$23,500	$59,703	$6,242	$48,405	$15,003	$44,989	$23,868	$69,911
Herbal Tea	Chamomile	$14,434	$36,570	$765	$2,194	$3,180	$11,186	$8,852	$25,632
	Lemon	$6,251	$21,978	$7,901	$27,176	$2,593	$14,497	$13,120	$32,274
	Mint	$4,069	$9,337	($2,242)	$11,992			$4,330	$14,380
	Total	$24,754	$67,885	$6,424	$41,362	$5,774	$25,683	$26,301	$72,285
Tea	Darjeeling	$10,772	$30,289	$6,497	$14,096			$11,780	$28,769
	Earl Grey	$10,331	$32,881	$3,405	$6,505			$10,425	$27,387
	Green Tea	$1,227	$5,211	$5,654	$11,571			($7,109)	$16,063
	Total	$22,330	$68,380	$15,557	$32,172			$15,097	$72,220

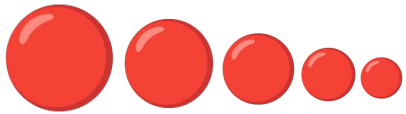
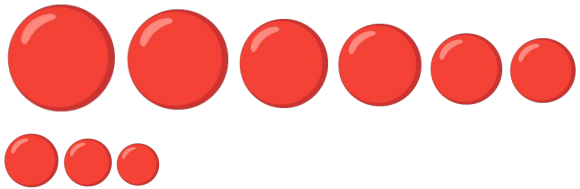




Presenting Different Data Types

Data Type	Description	Examples
Nominal	Data that represents categories without any inherent order. The values are just labels or names.	Apple, Banana, Orange Eagle, Jay, Hawk
Ordinal	Data that represents categories with a meaningful order, but the difference between categories is not quantifiable.	Poor, Fair, Good, Excellent Monday, Tuesday, Wednesday, ...
Quantitative	Data that consists of numerical values where arithmetic operations are meaningful.	150 cm, 160 cm, 170 cm 2.4, 5.98, 10.1, ...

Presenting Different Data Types via Area

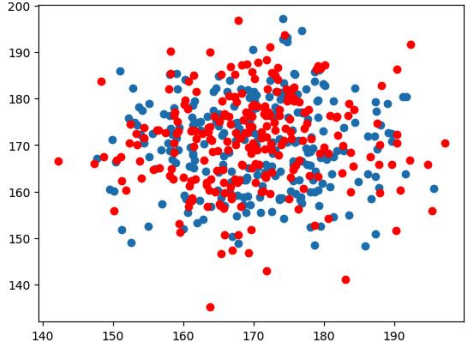
Data Type	Description	Examples
Nominal	Data that represents categories without any inherent order. The values are just labels or names.	(Conveys ordering… 😬)
Ordinal	Data that represents categories with a meaningful order, but the difference between categories is not quantifiable.	
Quantitative	Data that consists of numerical values where arithmetic operations are meaningful.	

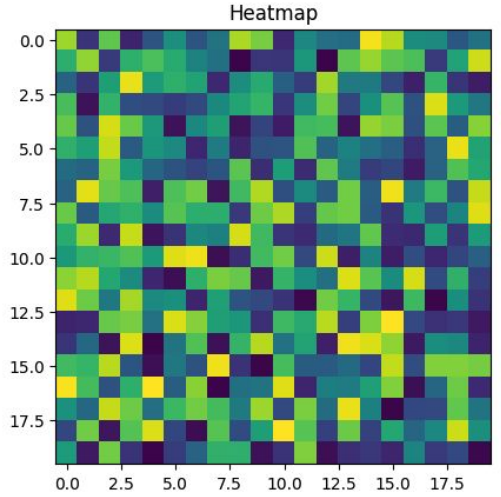
Presenting Different Data Types via Color

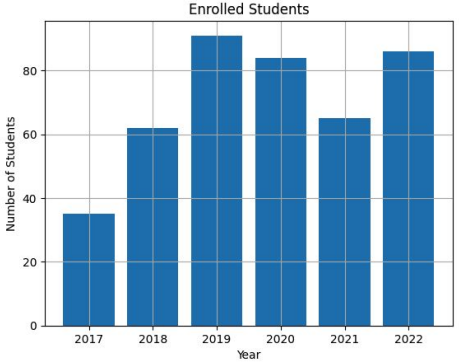
Data Type	Description	Examples
Nominal	Data that represents categories without any inherent order. The values are just labels or names.	
Ordinal	Data that represents categories with a meaningful order, but the difference between categories is not quantifiable.	
Quantitative	Data that consists of numerical values where arithmetic operations are meaningful.	

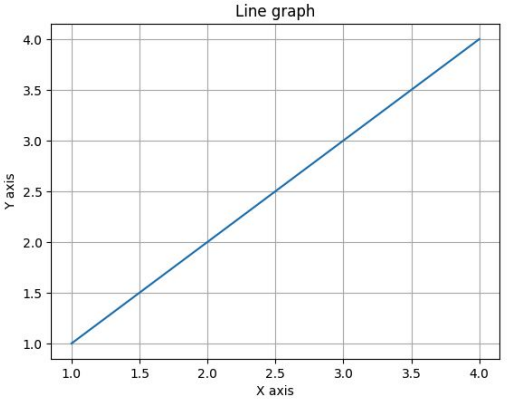
Various Formats

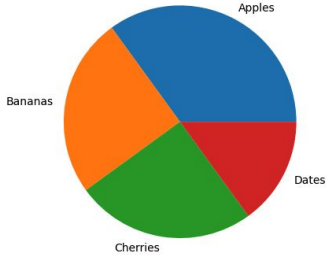
- Heatmap
- Histogram
- Bar chart
- Line chart
- Pie chart
- Scatter plot
- Dashboard
- … and more











Heatmap

- Special case of a table
- Purpose: Showing magnitude using color intensity.
- Best for: Correlation matrices, frequency distributions.

Table

	A	B	C
Category 1	15%	22%	42%
Category 2	40%	36%	20%
Category 3	35%	17%	34%
Category 4	30%	29%	26%
Category 5	55%	30%	58%
Category 6	11%	25%	49%

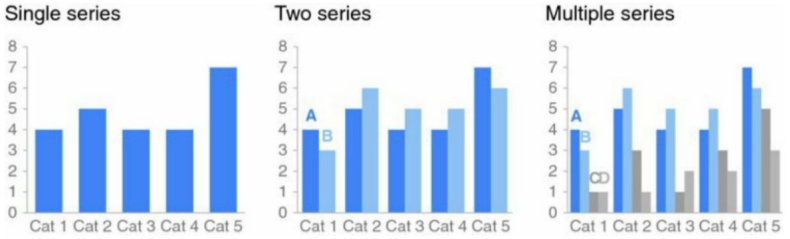
Heatmap

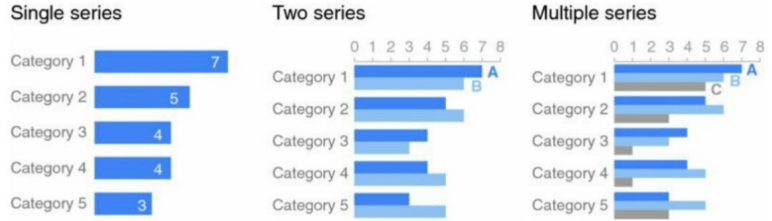
LOW-HIGH

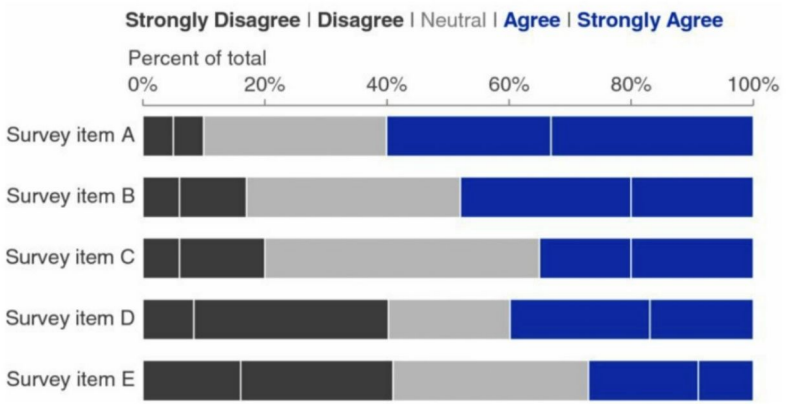
	A	B	C
Category 1	15%	22%	42%
Category 2	40%	36%	20%
Category 3	35%	17%	34%
Category 4	30%	29%	26%
Category 5	55%	30%	58%
Category 6	11%	25%	49%

Bar Chart

- Each bar represents a category
- Its height corresponds to the value or frequency of that category.
- Various forms:
  - Vertical
  - Horizontal
  - Stacked

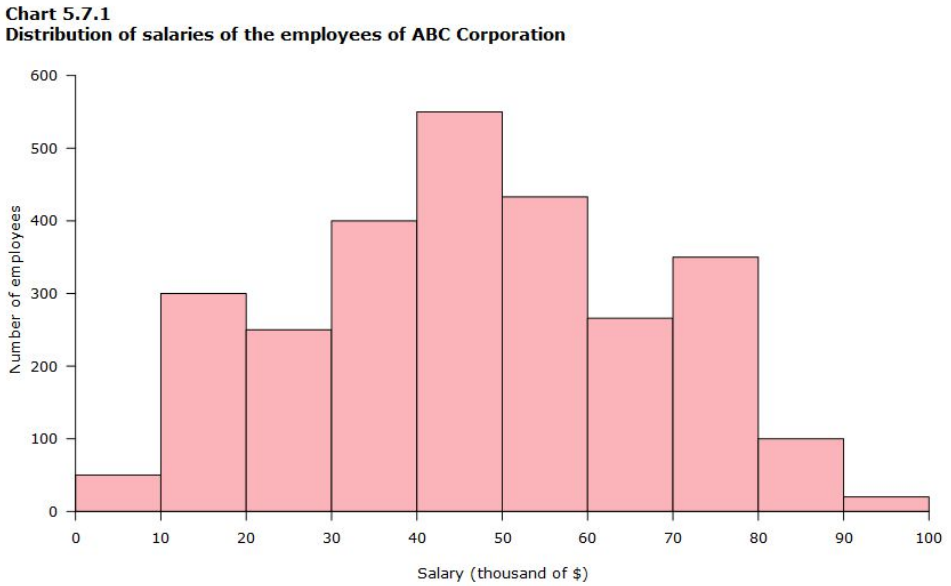






Histogram

- Consists of bars
- Each bar represents the frequency (or count) of data points within a specific range (or bin).



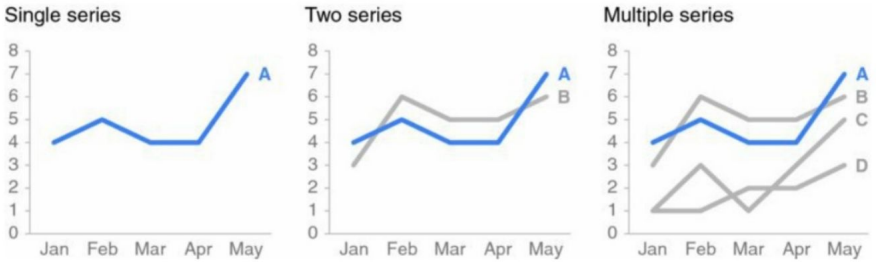
Ref: https://www150.statcan.gc.ca/n1/edu/power-pouvoir/ch9/histo/5214822-eng.htm

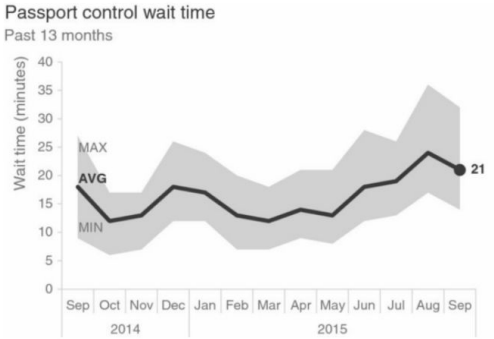
Histogram VS Bar Chart

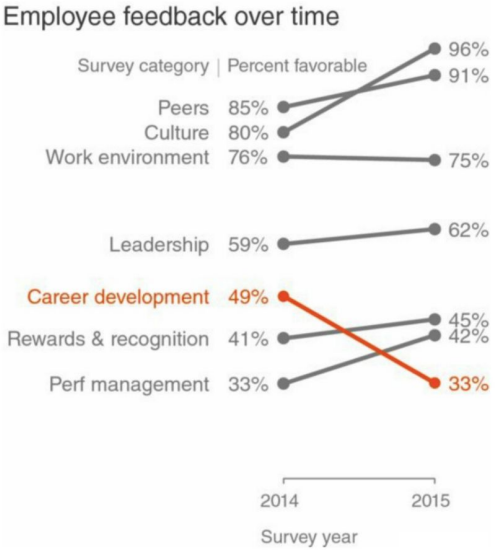
	Histogram	Bar Chart
Type of Data	Continuous	Categorical (discrete)
X-axis	Ranges (bins of numbers)	Distinct categories
Purpose	To show distribution of data	To compare values between categories
Use Cases	Analyzing frequency, shape, and spread of data (e.g., test scores, temperatures)	Comparing groups (e.g., sales by region, customer preferences)

Line Chart

- Used to plot continuous data
- Show trend
- Various forms:
  - Line
  - Line with range
  - Slopegraph

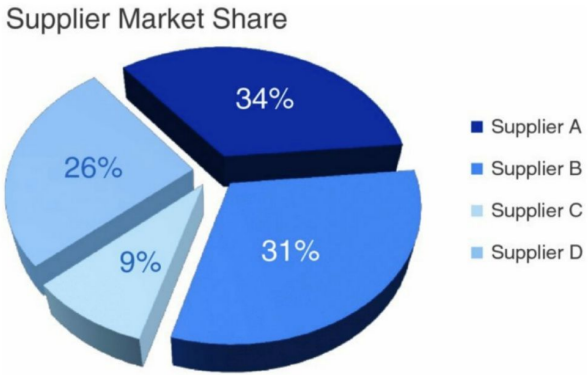


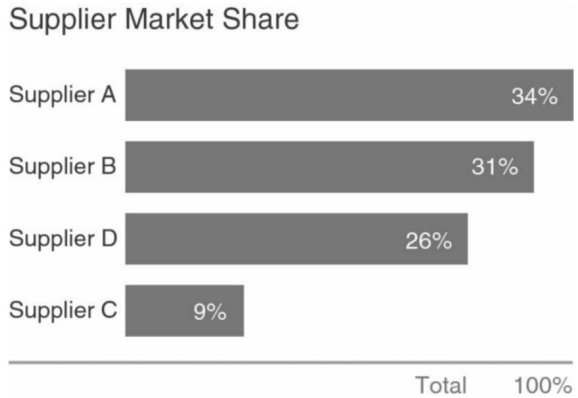




Pie Chart

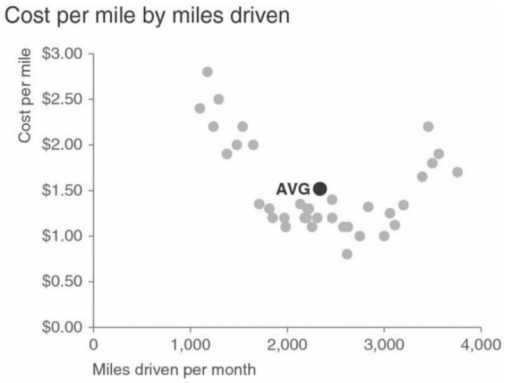
- A circular statistical graphic divided into slices
- Each slice represents a proportion of a whole
- Could be hard to read
- Not recommended when precision and clarity are important

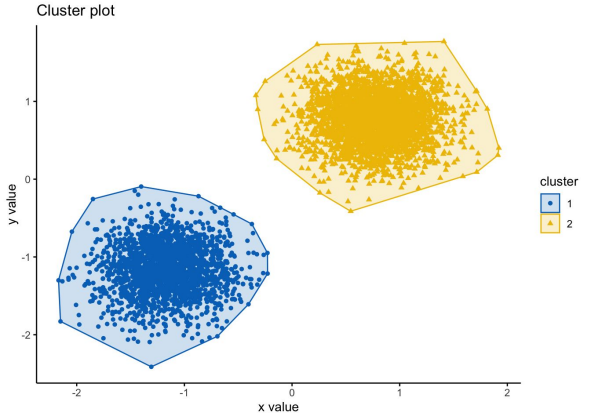




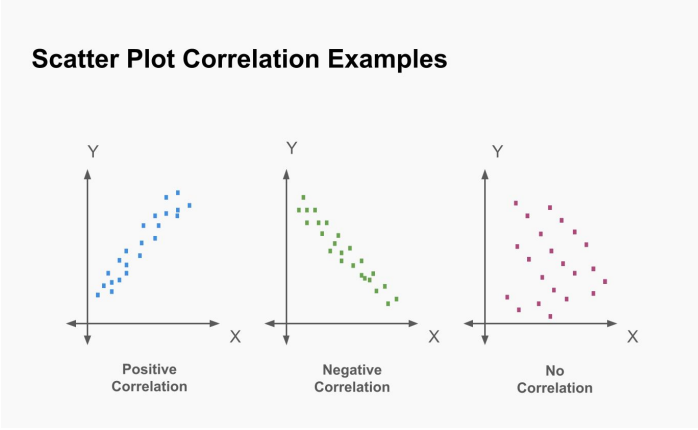
Scatter Plot

- Points
- Purpose: To visualize relationships or correlations between two numerical variables.
- Convey:
  - Interesting point
  - Trend
  - Cluster





Ref: https://bookdown.org/brittany_davidson1993/bookdown-demo/cluster-analysis-part-1.html



Ref: https://planyway.com/blog/how-to-make-a-scatter-plot

Dashboard

- A panel to display all your data visualizations in one place
- Various types of visual data
- Quick understanding at a glance

Sales

$297K
this month
▲ $16K vs last month

$9.6K
today

$20.6K
yesterday

Biggest deals this month

Alice	$8,600
Jared	$8,500
Heather	$7,540
Shaun	$7,450
Marsha	$6,530
Jared	$4,565
Heather	$4,560
Polly	$4,215
Dalisu	$3,560

Social followers

19.5K
LinkedIn
▲ 11 v yday

10.5K
Twitter
▲ 22 v yday

Website (past 7 days)

27.2K
Users
▲ 1.6K vs last week

126
Enquiries
▼ 28 vs last week

Active users

NPS (past 30 days)

61

Recent feedback

OK
14 days ago

Very Helpful!!
2 months ago

very good "thumbs up"
2 months ago

Daily pulse dashboard

13:33

- S. Tarnpradab -

Ref: https://www.geckoboard.com/dashboard-examples/

Dashboard vs Report

Dashboard

- High-level view of the data
- Created to answer a single question

Report

- More narrow focus
- In-depth view into a dataset
- Tend to concentrate on a single item or event

Application Examples

- Customer metrics
- Logistics information
- Sales information
- Human resources data
- Web analytics
- Project management

Python Libraries

Static

- Matplotlib
- Seaborn

Interactive

- Plotly
- Bokeh
- pygal
- Folium
- PowerBI

→ Freedom to fully explore data
→ More engaging
→ Better understanding of data

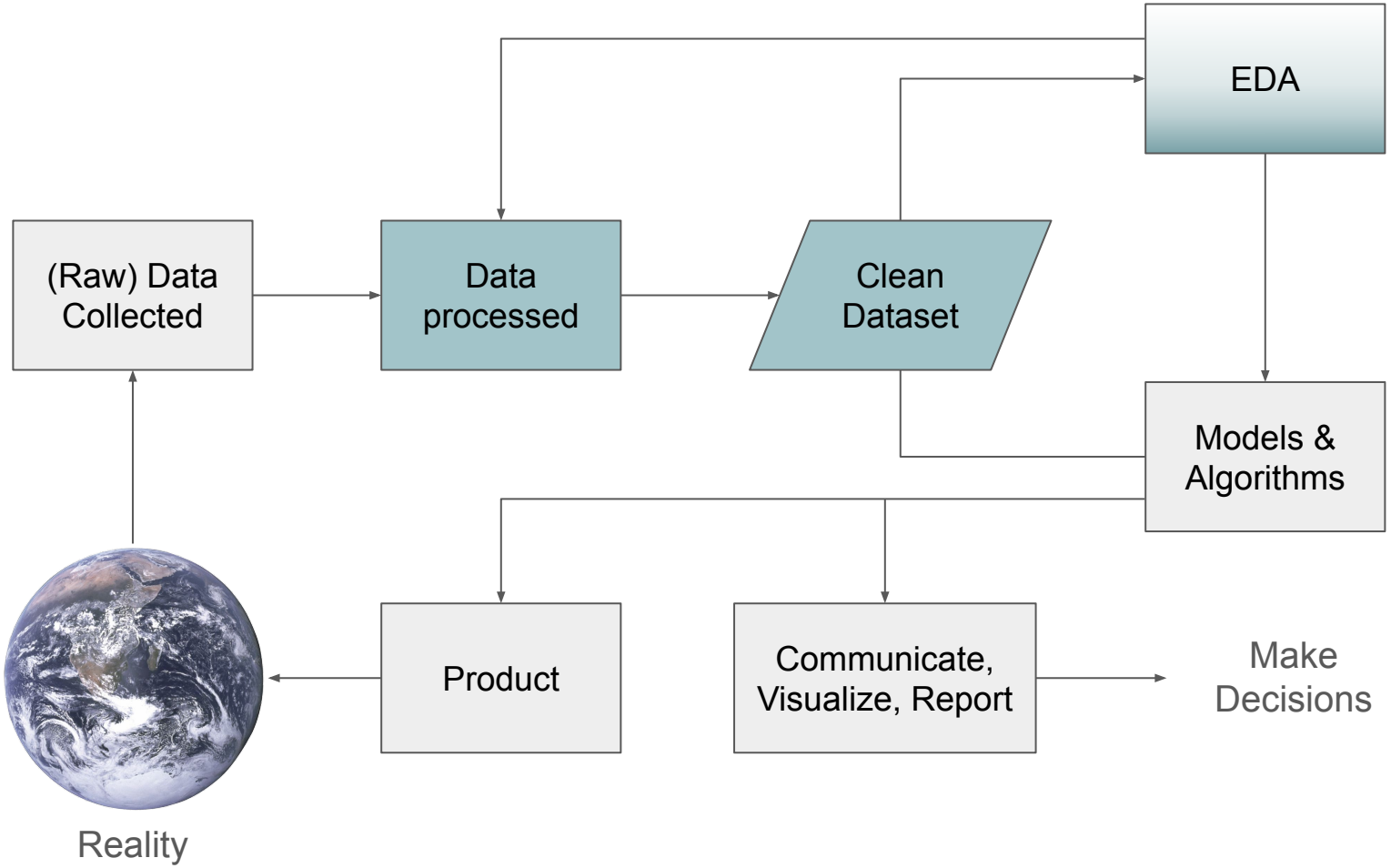
Interactive Visualization: Pros & Cons

Pros

- Allow users to explore data at their own pace and level
- Provide user with more control
  - Hypothesis testing
  - Compare scenario
  - Perform what-if analysis
- Enhance storytelling and presentation

Cons

- Require time, effort, skills to develop than static charts
- Could confuse users
  - Many options → Distraction
- Readers might come up with a different conclusion


EDA
(Raw) Data Collected
Data processed
Clean Dataset
Models & Algorithms
Product
Communicate, Visualize, Report
Make Decisions
Reality

In Summary

- Intro to Data Visualization
- Effective Visuals
- Types of Charts
- Dashboard
- Intro to Python libraries for Data Visualization
  - Static
  - Interactive

Ref: https://www.reddit.com/r/sciencememes/comments/13qp2wk/data_visualization_meme/

Q & A

Resources

matplotlib Cheat sheet

Version 3.5.0

Quick start

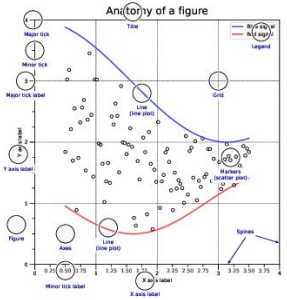
```
import numpy as np
import matplotlib as mpl
import matplotlib.pyplot as plt


X = np.linspace(0, 2*np.pi, 100)
Y = np.cos(X)

fig, ax = plt.subplots()
ax.plot(X, Y, color='green')

fig.savefig("figure.pdf")
plt.show()
```

Anatomy of a figure



Subplots layout

```
subplot[s](rows,cols,…)
fig, axs = plt.subplots(3, 3)

G = gridspec(rows,cols,…)
ax = G[0,:]

ax.inset_axes(extent)

d=make_axes_locatable(ax)
ax = d.new_horizontal('10%')
```

Getting help

- matplotlib.org
- github.com/matplotlib/matplotlib/issues
- discourse.matplotlib.org
- stackoverflow.com/questions/tagged/matplotlib
- https://gitter.im/matplotlib/matplotlib
- twitter.com/matplotlib
- Matplotlib users mailing list

Basic plots

```
plot([X],Y,[fmt],…)
X, Y, fmt, color, marker, linestyle
```

```
scatter(X,Y,…)
X, Y, [s]izes, [c]olors, marker, cmap
```

```
bar[h](x,height,…)
x, height, width, bottom, align, color
```

```
imshow(Z,…)
Z, cmap, interpolation, extent, origin
```

```
contour[f]([X],[Y],Z,…)
X, Y, Z, levels, colors, extent, origin
```

```
pcolormesh([X],[Y],Z,…)
X, Y, Z, vmin, vmax, cmap
```

```
quiver([X],[Y],U,V,…)
X, Y, U, V, C, units, angles
```

```
pie(X,…)
Z, explode, labels, colors, radius
```

```
text(x,y,text,…)
x, y, text, va, ha, size, weight, transform
```

```
fill[_between][x](…)
X, Y1, Y2, color, where
```

Advanced plots

```
step(X,Y,[fmt],…)
X, Y, fmt, color, marker, where
```

```
boxplot(X,…)
X, notch, sym, bootstrap, widths
```

```
errorbar(X,Y,xerr,yerr,…)
X, Y, xerr, yerr, fmt
```

```
hist(X, bins, …)
X, bins, range, density, weights
```

```
violinplot(D,…)
D, positions, widths, vert
```

```
barbs([X],[Y], U, V, …)
X, Y, U, V, C, length, pivot, sizes
```

```
eventplot(positions,…)
positions, orientation, lineoffsets
```

```
hexbin(X,Y,C,…)
X, Y, C, gridsize, bins
```

Scales

```
ax.set_[xy]scale(scale,…)
```

linear — any values
log — values > 0
symlog — any values
logit — 0 < values < 1

Projections

```
subplot(…,projection=p)
```

p='polar'
p='3d'

```
p=ccrs.Orthographic()
import cartopy.crs as ccrs
```

Lines

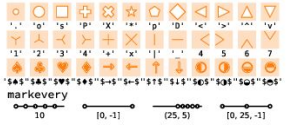
linestyle or ls

"-" "--" "-." ":" (0,(0.01,2))

capstyle or dash_capstyle

"butt" "round" "projecting"

Markers



markevery

10 [0, -1] (25, 5) [0, 25, -1]

Colors

'Cn' 'x' 'name' (R,G,B[,A]) '#RRGGBB[AA]' 'x.y'

Colormaps

```
plt.get_cmap(name)
```

Uniform: viridis magma plasma

Sequential: Greys YlOrBr Wistia

Diverging: Spectral coolwarm RdGy

Qualitative: tab10 tab20

Cyclic: twilight

Tick locators

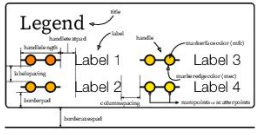
```
from matplotlib import ticker
ax.[xy]axis.set_[minor|major]_locator(locator)
ticker.NullLocator()
ticker.MultipleLocator(0.5)
ticker.FixedLocator([0, 1, 5])
ticker.LinearLocator(numticks=3)
ticker.IndexLocator(base=0.5, offset=0.25)
ticker.AutoLocator()
ticker.MaxNLocator(n=4)
ticker.LogLocator(base=10, numticks=15)
```

Tick formatters

```
from matplotlib import ticker
ax.[xy]axis.set_[minor|major]_formatter(formatter)
ticker.NullFormatter()
ticker.FixedFormatter(['zero', 'one', 'two', …])
ticker.FuncFormatter(lambda x, pos: "[%.2f]" % x)
ticker.FormatStrFormatter('>%d<')
ticker.ScalarFormatter()
ticker.StrMethodFormatter('{x}')
ticker.PercentFormatter(xmax=5)
```

Ornaments

```
ax.legend(…)
handles, labels, loc, title, frameon
```




```
ax.colorbar(…)
mappable, ax, cax, orientation
```

```
ax.annotate(…)
text, xy, xytext, xycoords, textcoords, arrowprops
```

Event handling

```
fig, ax = plt.subplots()
def on_click(event):
  print(event)
fig.canvas.mpl_connect(
  'button_press_event', on_click)
```

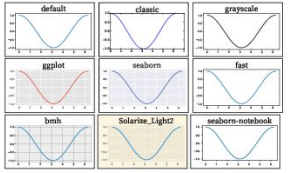
Animation

```
import matplotlib.animation as mpla

T = np.linspace(0, 2*np.pi, 100)
S = np.sin(T)
line, = plt.plot(T, S)
def animate(i):
    line.set_ydata(np.sin(T+i/50))
anim = mpla.FuncAnimation(
    plt.gcf(), animate, interval=5)
plt.show()
```

Styles

```
plt.style.use(style)
```



Quick reminder

```
ax.grid()
ax.set_[xy]lim(vmin, vmax)
ax.set_[xy]label(label)
ax.set_[xy]ticks(ticks, [labels])
ax.set_[xy]ticklabels(labels)
ax.set_title(title)
ax.tick_params(width=10, …)
ax.set_axis_[on|off]()

fig.suptitle(title)
fig.tight_layout()
plt.gcf(), plt.gca()
mpl.rc('axes', linewidth=1, …)
[fig|ax].patch.set_alpha(0)
text=r'$\frac{-e^{i\pi}}{2^n}$'
```

Keyboard shortcuts

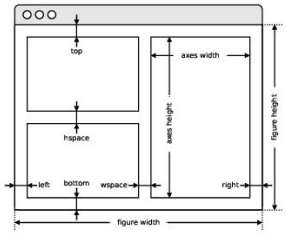
ctrl + s	Save	ctrl + w	Close plot
r	Reset view	f	Fullscreen 0/1
f	View forward	b	View back
p	Pan view	o	Zoom to rect
x	X pan/zoom	y	Y pan/zoom
g	Minor grid 0/1	G	Major grid 0/1
l	X axis log/linear	L	Y axis log/linear

Ten simple rules

1. Know your audience
2. Identify your message
3. Adapt the figure
4. Captions are not optional
5. Do not trust the defaults
6. Use color effectively
7. Do not mislead the reader
8. Avoid "chartjunk"
9. Message trumps beauty
10. Get the right tool

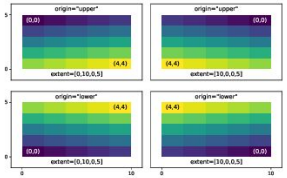
Axes adjustments

plt.subplots_adjust(...)



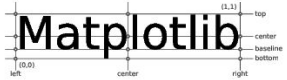
Extent & origin

ax.imshow(extent=..., origin=...)



Text alignments

ax.text(..., ha=... , va=..., ...)



Text parameters

ax.text(..., family=..., size=..., weight=...)
ax.text(..., fontproperties=...)

The quick brown fox — xx-large (1.73)
The quick brown fox — x-large (1.44)
The quick brown fox — large (1.20)
The quick brown fox — medium (1.00)
The quick brown fox — small (0.83)
The quick brown fox — x-small (0.69)
The quick brown fox — xx-small (0.58)

The quick brown fox jumps over the lazy dog — black (900)
The quick brown fox jumps over the lazy dog — bold (700)
The quick brown fox jumps over the lazy dog — semibold (600)
The quick brown fox jumps over the lazy dog — normal (400)
The quick brown fox jumps over the lazy dog — ultralight (100)

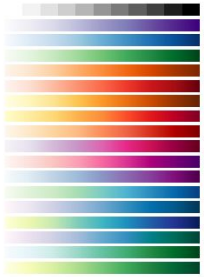
The quick brown fox jumps over the lazy dog — monospace
The quick brown fox jumps over the lazy dog — serif
The quick brown fox jumps over the lazy dog — sans
The quick brown fox jumps over the lazy dog — cursive

The quick brown fox jumps over the lazy dog — italic
The quick brown fox jumps over the lazy dog — normal

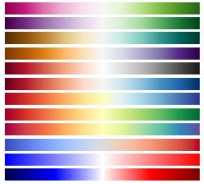
THE QUICK BROWN FOX JUMPS OVER THE LAZY DOG — small-caps
The quick brown fox jumps over the lazy dog — normal

Uniform colormaps

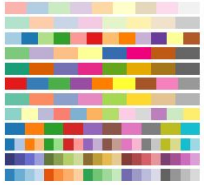
Sequential colormaps



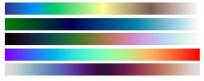
Diverging colormaps



Qualitative colormaps

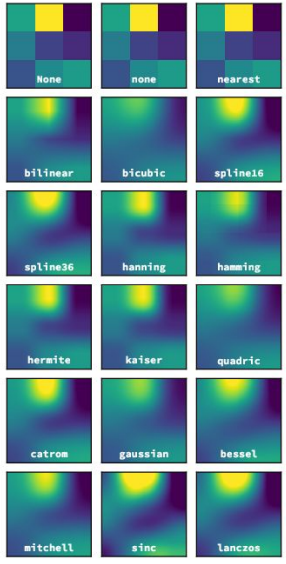


Miscellaneous colormaps

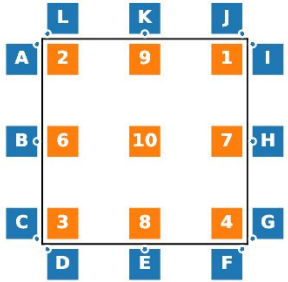


Color names

Image interpolation



Legend placement

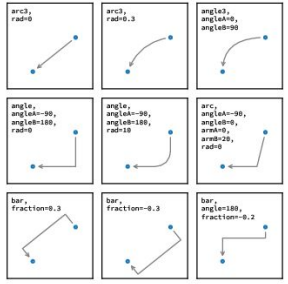


ax.legend(loc="string", bbox_to_anchor=(x,y))

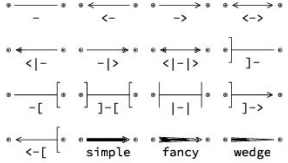
2: upper left
6: center left
3: lower left
9: upper center
10: center
8: lower center
1: upper right
7: center right
4: lower right

A: upper right / (-0.1,0.9)
B: center right / (-0.1,0.5)
C: lower right / (-0.1,0.1)
D: upper left / (0.1,-0.1)
E: upper center / (0.5,-0.1)
F: upper right / (0.9,-0.1)
G: lower left / (1.1,0.1)
H: center left / (1.1,0.5)
I: upper left / (1.1,0.9)
J: lower right / (0.9,1.1)
K: lower center / (0.5,1.1)
L: lower left / (0.1,1.1)

Annotation connection styles



Annotation arrow styles



How do I ...

... resize a figure?
→ fig.set_size_inches(w, h)
... save a figure?
→ fig.savefig("figure.pdf")
... save a transparent figure?
→ fig.savefig("figure.pdf", transparent=True)
... clear a figure/an axes?
→ fig.clear() → ax.clear()
... close all figures?
→ plt.close("all")
... remove ticks?
→ ax.set_[xy]ticks([])
... remove tick labels ?
→ ax.set_[xy]ticklabels([])
... rotate tick labels ?
→ ax.tick_params(axis="x", rotation=90)
... hide top spine?
→ ax.spines['top'].set_visible(False)
... hide legend border?
→ ax.legend(frameon=False)
... show error as shaded region?
→ ax.fill_between(X, Y+error, Y-error)
... draw a rectangle?
→ ax.add_patch(plt.Rectangle((0, 0), 1, 1)
... draw a vertical line?
→ ax.axvline(x=0.5)
... draw outside frame?
→ ax.plot(..., clip_on=False)
... use transparency?
→ ax.plot(..., alpha=0.25)
... convert an RGB image into a gray image?
→ gray = 0.2989*R + 0.5870*G + 0.1140*B
... set figure background color?
→ fig.patch.set_facecolor("grey")
... get a reversed colormap?
→ plt.get_cmap("viridis_r")
... get a discrete colormap?
→ plt.get_cmap("viridis", 10)
... show a figure for one second?
→ fig.show(block=False), time.sleep(1)

Performance tips

```
scatter(X, Y)                                  slow
plot(X, Y, marker="o", ls="")                  fast

for i in range(n): plot(X[i])                  slow
plot(sum([x+[None] for x in X],[]))            fast

cla(), imshow(…), canvas.draw()                slow
im.set_data(…), canvas.draw()                  fast
```

Beyond Matplotlib

Seaborn: Statistical data visualization
Cartopy: Geospatial data processing
yt: Volumetric data visualization
mpld3: Bringing Matplotlib to the browser
Datashader: Large data processing pipeline
plotnine: A grammar of graphics for Python

NUMFOCUS
OPEN CODE = BETTER SCIENCE

plotly

GETTING STARTED

1. Install

In the terminal

```
sudo pip install plotly
```

2. Sign Up & Configure

http://www.plot.ly/python/getting-started

3. Boilerplate Imports

```
import plotly.plotly as py
import plotly.graph_objs as go
```

4. A Hello World Figure

```
trace = { 'x' : [1, 2] , 'y' : [1, 2] }
data = [ trace ]
layout = { }
fig = go.Figure (
   data = data, layout = layout )
```

5. Plot the Figure!

In the terminal:

```
plot_url = py.plot ( fig )
```

Or in the IPython notebook:

```
py.iplot ( fig )
```

BASIC CHARTS

Line Plots

```
trace1 = go.Scatter (
   x = [ 1, 2 ] , y = [ 1, 2 ] )
trace2 = go.Scatter (
   x = [ 1, 2 ] , y = [ 2, 1 ] )
py.iplot ( [ trace1, trace2 ] )
```

Bubble Charts

```
trace = go.Scatter (
  x = [ 1, 2, 3 ] , y = [ 1, 2, 3 ] ,
  marker = dict (
    color = [ 'red', 'blue' ,
    'green' ]
    size = [ 30, 80, 200 ] ) ,
    mode = 'markers' )
py.iplot ( [ trace ] )
```

Scatter Plots

```
trace1 = go.Scatter (
   x = [ 1, 2, 3 ] , y = [ 1 ,2, 3 ] ,
   text = [ 'A','B','C' ] ,
   textposition = 'top center'
   mode = 'markers+text' )
mode = [ trace ]
py.iplot ( data )
```

Heatmaps

```
trace = go.Heatmap (
   z = [ [ 1, 2, 3, 4 ] ,
       [ 5, 6, 7, 8 ] ] )
data = [ trace ]
py.iplot ( data )
```

Bar Charts

```
trace = go.Bar (
   x = [ 1, 2 ] , y = [ 1, 2 ] )
data = [ trace ]
py.iplot ( data )
```

Area Plots

```
trace = go.Scatter (
   x = [ 1, 2 ] , y = [ 1, 2 ] ,
   fill = 'tonexty' )
data = [ trace ]
py.iplot ( data )
```

LAYOUT

Legends

```
trace1 = go.Scatter (
   name = 'Calvin'
   x = [ 1, 2 ] , y = [ 1, 2 ] )

trace2 = go.Scatter (
   name = 'Hobbes'
   x = [ 2, 1 ] , y = [ 2, 1 ] )

layout = go.Layout (
   showlegend = True ,
   legend = dict (
      x = 0.2 , y = 0.5 )
   )

data =  [ trace1 , trace2 ]
fig = go.Figure (
   data = data ,
   layout = layout )
py.iplot ( fig )
```

Axes

```
trace = go.Scatter (
   x = [ 1, 2, 3, 4 ] ,
   y = [ 1, 2, 3, 6 ] )

axis_template = dict (
   showgrid = False ,
   zeroline = False ,
   nticks = 20 ,
   showline = True ,
   title = 'X AXIS'
   mirror = 'all' )
layout = go.Layout (
   xaxis = axis_template ,
   yaxis = axis_template ,
   )

data = [ trace ]
fig = go.Figure (
   data = data
   layout = layout
py.iplot ( fig )
```

PYTHON CLIENT **PLOT.LY/PYTHON** ALL LAYOUTS **PLOT.LY/PYTHON/REFERENCE/#LAYOUT**

STATISTICAL CHARTS

Histograms

```
trace = go.Histogram (
   x = [ 1, 2, 3, 3, 3, 4, 5 ] )
data = [ trace ]
py.iplot ( data )
```

Box Plots

```
trace = go.Box (
   x = [ 1, 2, 3, 3, 3, 4, 5 ] )
data = [ trace ]
py.iplot ( data )
```

2D Histogram

```
trace = go.Historgram2d (
   x = [ 1, 2, 3, 3, 3, 4, 5 ],
   x = [ 1, 2, 3, 3, 3, 4, 5 ] )
data = [ trace ]
py.iplot ( data )
```

MAPS

Bubble Map

```
trace = dict (
  type = 'scattergeo' ,
  lon = [ 100, 400 ] , lat = [ 0, 0] ,
  marker = dict (
    marker = [ 'red', 'blue' ]
    size = [ 30, 50 ] ) ,
  mode = 'markers' )
py.iplot ( [ trace ] )
```

Choropleth Map

```
trc = dict (
   type = 'choropleth' ,
   locations = [ 'AZ' , 'CA' , 'VT' ] ,
   locationmode = 'USA-states' ,
   colorscale = [ 'Viridis' ] ,
   z = [ 10, 20, 40 ] )
lyt = dict ( geo = dict ( scope = 'usa' ) )
map = go.Figure ( data = [ trc ] ,
   layout = lyt )
py.iplot ( map)
```

Scatter Map

```
trace = dict (
   type = 'scattergeo' ,
   lon = [ 42, 39 ] , lat = [ 12, 22 ] ,
   marker = [ 'Rome' , 'Greece' ] ,
   mode = 'markers' )
py.iplot ( [ trace ] )
```

3D CHARTS

3D Surface Plots

```
trace = go.Surface (
   colorscale = 'Viridis' ,
   z = [ [ 3, 5, 8, 13] ,
       [ 21, 13, 8, 5 ] )
data =  [ trace ]
py.iplot ( data )
```

3D Line Plots

```
trace = go.Scatter3D (
   x = [ 9, 8, 5, 1] , y = [ 1, 2, 4, 8 ] ,
   z = [ 11, 8, 15, 3 ] ,
   mode = 'lines' )
data = [ trace ]
py.iplot ( data )
```

3D Scatter Plots

```
trace = go.Scatter3D (
   x = [ 9, 8, 5, 1] , y = [ 1, 2, 4, 8 ] ,
   z = [ 11, 8, 15, 3 ] ,
   mode = 'markers' )
data = [ trace ]
py.iplot ( data )
```

FIGURE HIERARCHY

Figure { }

```
DATA [ ]
  TRACE { }
    x, y, z [ ]
    color, text, size [ ]
    colorscale ABC or [ ]
    MARKER { }
      color ABC
      symbol ABC
      LINE { }
        color ABC
        width 123
LAYOUT { }
  title ABC
  XAXIS, YAXIS { }
  SCENE { }
    XAXIS, YAXIS, ZAXIS { }
  GE0 { }
  LEGEND { }
  ANNOTATIONS { }
```

{ } = dictionary
[ ] = list
ABC = string
123 = number

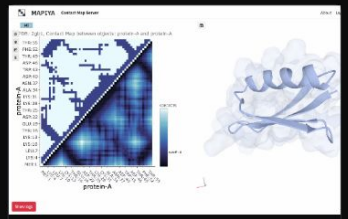
PYTHON CLIENT **PLOT.LY/PYTHON** ALL LAYOUTS **PLOT.LY/PYTHON/REFERENCE**

plotly DASH ENTERPRISE DOCS EXAMPLE APPS COMPANY PRICING DEMO DASH

Bioinformatics & Life Sciences

Singapore Institute for Clinical Sciences (SICS) is not the first to visualize health data through Dash's interactive platform. Explore examples of bioinformatics in a variety of Python and Dash applications, ranging from genomics, public health, clinical trials, 3D images, and more.

← ALL APPS

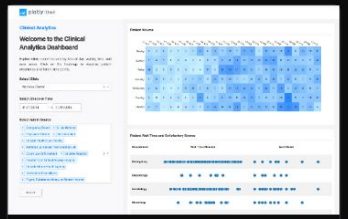


Biomolecular Visualization

Visualization of biomolecular interactions in proteins and biological complexes.

LEARN MORE

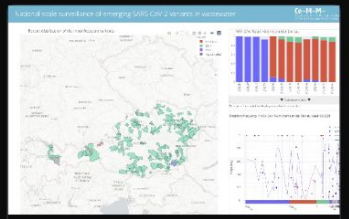
By Sebastian Kmiecik



Clinical Patient Dashboard

Explore clinic patient volume by time of day, waiting time, and care score.

By Plotly

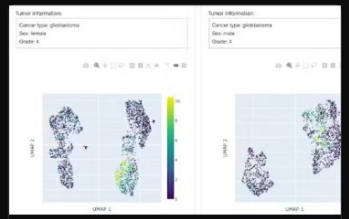


National Wastewater Surveillance

A realtime dashboard exploring emerging variants of SARS-CoV-2 in wastewater in Austria.

LEARN MORE

By Petr Triska



Characterizing Tumor Subpopulations

Exploring scRNA-seq cancer data sets in the NCBI's Gene Expression Omnibus.

LEARN MORE

By Morgridge Instutite

https://plotly.com/examples/bioinformatics/