AN INITIATIVE OF:

science, technology & innovation

Department:
Science, Technology and Innovation
REPUBLIC OF SOUTH AFRICA

CSIR
Touching lives through innovation

80th anniversary

nicis

NATIONAL INTEGRATED
CYBERINFRASTRUCTURE SYSTEM

DIRISA

Introduction to Data Visualization

Minenhle Mabuyakhulu

Rosemary Erawemen

24 June 2025

Introduction

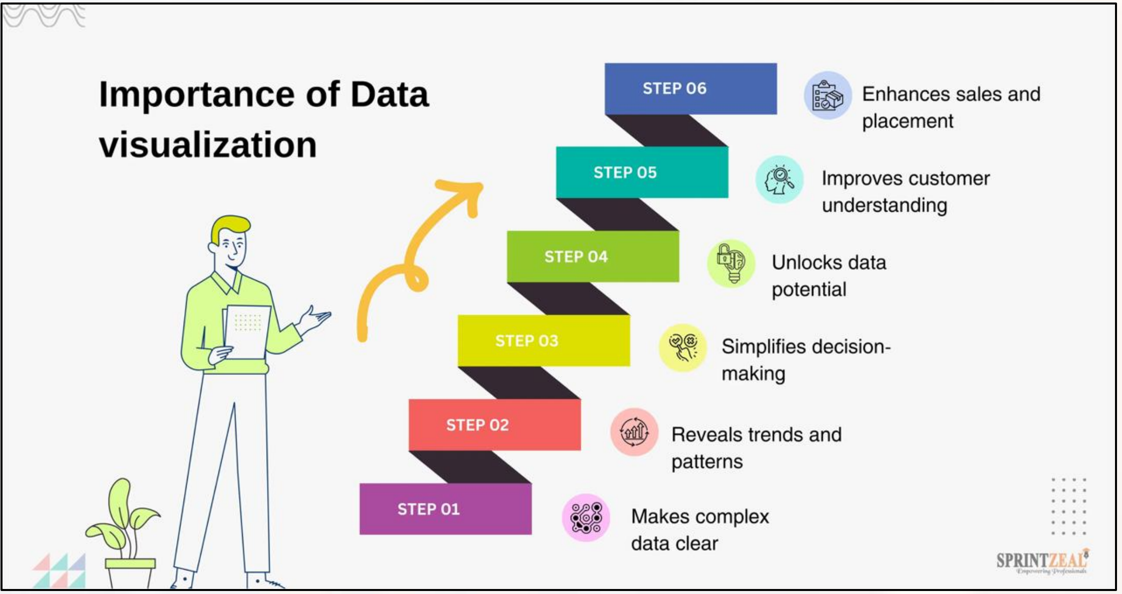
- Visualization can be used to:
- Better scrutinize a dataset (e.g., identifying outliers);
- Obtain a better understanding of data (breakdown);
- Find relationships and patterns in the dataset (pattern recognition)
- Sharing results and concepts with others (communication).

What is Data Visualization?

- It's a way to **turn numbers and data into pictures**.
- These pictures can be **charts, graphs, or maps**.
- It helps us **see and understand data more easily**.
- Makes complex data **look simple and clear**.
- Helps us to **find patterns**, **compare things**, and **tell stories with data**.

Goals of Data Visualization

- To **make data easy to understand**
- To **find patterns or trends** in the data
- To **spot mistakes or problems** quickly
- To **share results clearly** with others
- To **help in decision-making**


Importance of Data visualization
STEP 06
Enhances sales and placement
STEP 05
Improves customer understanding
STEP 04
Unlocks data potential
STEP 03
Simplifies decision-making
STEP 02
Reveals trends and patterns
STEP 01
Makes complex data clear
SPRINTZEAL

Source: [1]

Types of Visualization

- In python ***matplotlib*** or ***seaborn*** can be used.
- Some examples of different visualization methods are:
  - **Line Graph**
  - **Scatterplot**
  - **Histogram**
  - **Heat Map**
  - **Bar And Stacked Graph**
  - **Pie Chart**
  - **Violin And Swarm Plot**

Common Graphs:

- **Line Graph** – shows trends over time (e.g., sales over months)
- **Scatter Plot** – shows relationships between two things (e.g., height vs. weight)
- **Histogram** – shows how often values appear (e.g., test scores)
- **Heatmap** – uses color to show patterns (e.g., correlation between variables)
- **Bar Chart / Stacked Bar** – compares categories (e.g., number of students per course)
- **Pie Chart** – shows parts of a whole (e.g., percentage of budget spent)
- **Violin Plot / Swarm Plot** – shows how data is spread out and grouped

01

Understanding Data before Visualization

Purpose of Reading and Understanding the Dataset for Visualization

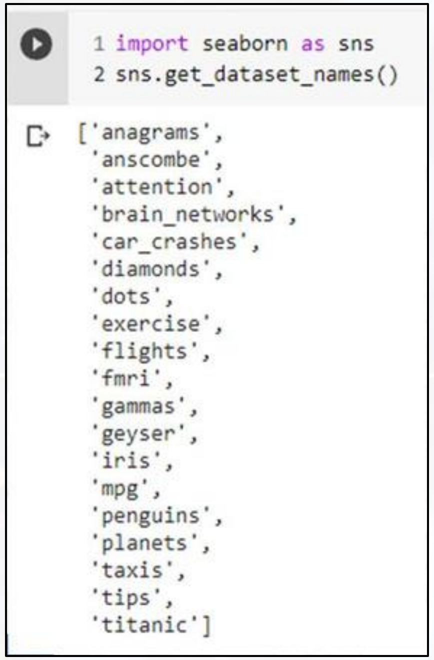
- Know what the data is about
  - Understand the topic and context.
- Understand each column or number
  - For example, "Sales," "Date," or "Product" — this helps choose the right chart.
- Check if the data fits visualization
  - Look at size, type (numerical or categorical), and if there are enough values.
- Identify the insight or message
  - Decide if you want to show trends, comparisons, distributions, or relationships.

Purpose of Reading and Understanding the Dataset for Visualization

- Make sure data is organized
  - It should be clear and ready for tools like Excel, Python, or Power BI.
- Spot basic issues
  - Watch out for missing values, outliers, or mislabeled columns to avoid confusing visuals.
- Select relevant data only
  - Focus on columns and values that support your story; exclude unnecessary data.

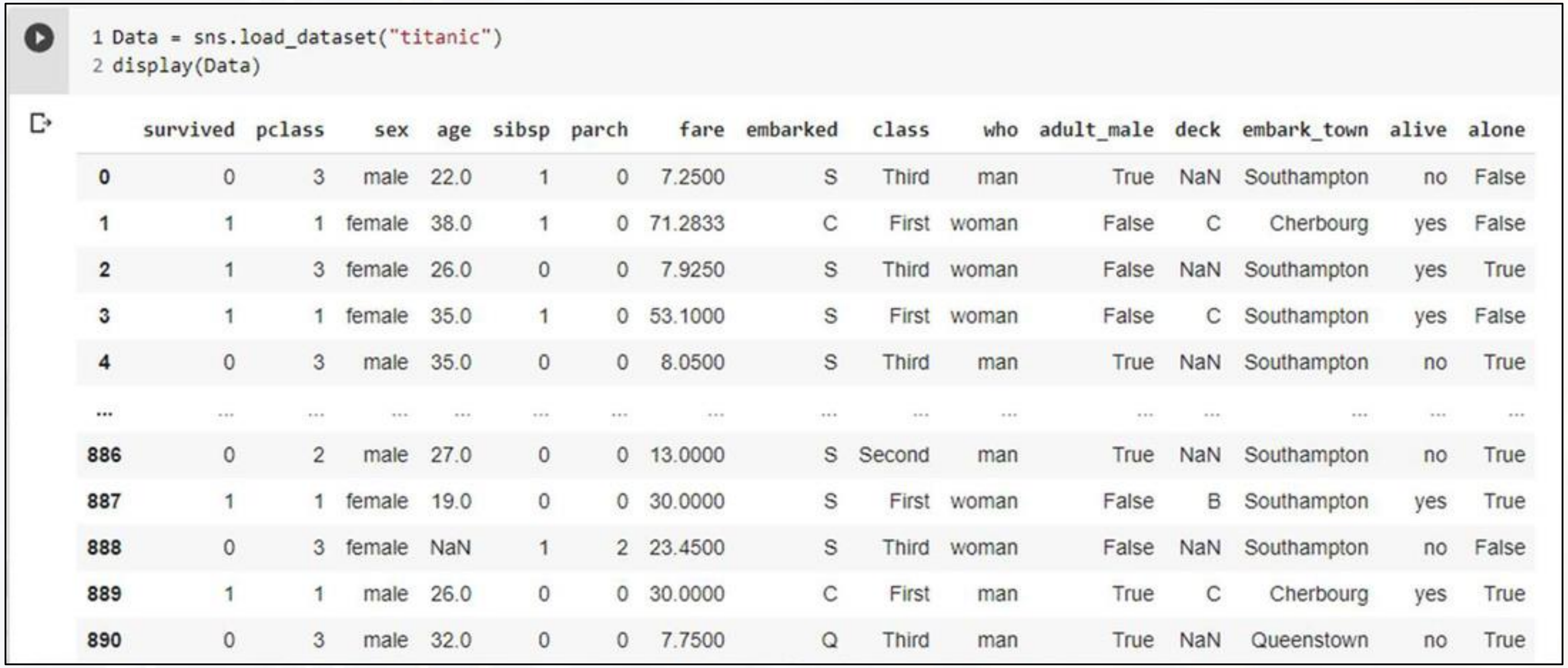
Dataset

- Small seaborn online datasets can be obtained to test some visualization techniques.



```python
import seaborn as sns
sns.get_dataset_names()
```

```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'exercise',
 'flights',
 'fmri',
 'gammas',
 'geyser',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'taxis',
 'tips',
 'titanic']
```



```python
Data = sns.load_dataset("titanic")
display(Data)
```

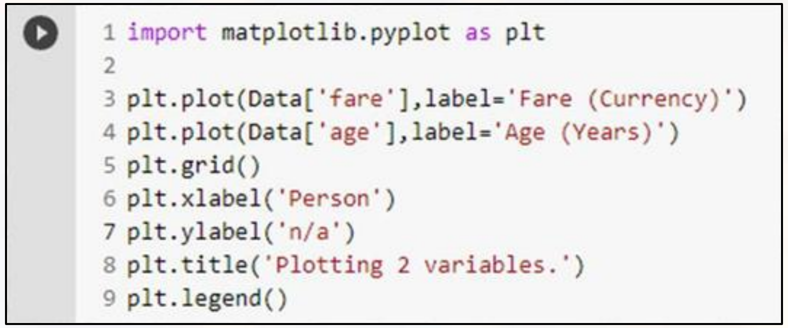
	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	deck	embark_town	alive	alone
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	NaN	Southampton	no	False
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	C	Cherbourg	yes	False
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	NaN	Southampton	yes	True
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	C	Southampton	yes	False
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	NaN	Southampton	no	True
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
886	0	2	male	27.0	0	0	13.0000	S	Second	man	True	NaN	Southampton	no	True
887	1	1	female	19.0	0	0	30.0000	S	First	woman	False	B	Southampton	yes	True
888	0	3	female	NaN	1	2	23.4500	S	Third	woman	False	NaN	Southampton	no	False
889	1	1	male	26.0	0	0	30.0000	C	First	man	True	C	Cherbourg	yes	True
890	0	3	male	32.0	0	0	7.7500	Q	Third	man	True	NaN	Queenstown	no	True

02

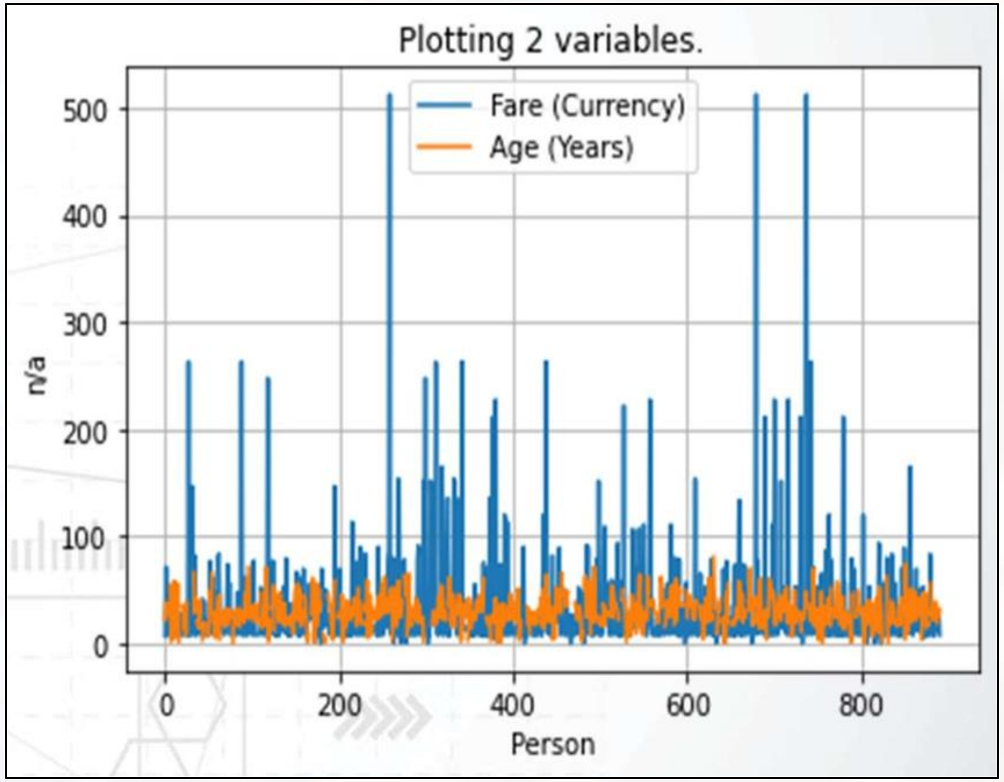
Common types of Graphs and When to Use Them

Line Graph

- Understanding of sequential such as time or the difference between similar variables.
  - It's used when you want to **see a trend or change**
  - Helps us see the **direction** and **speed** of change.

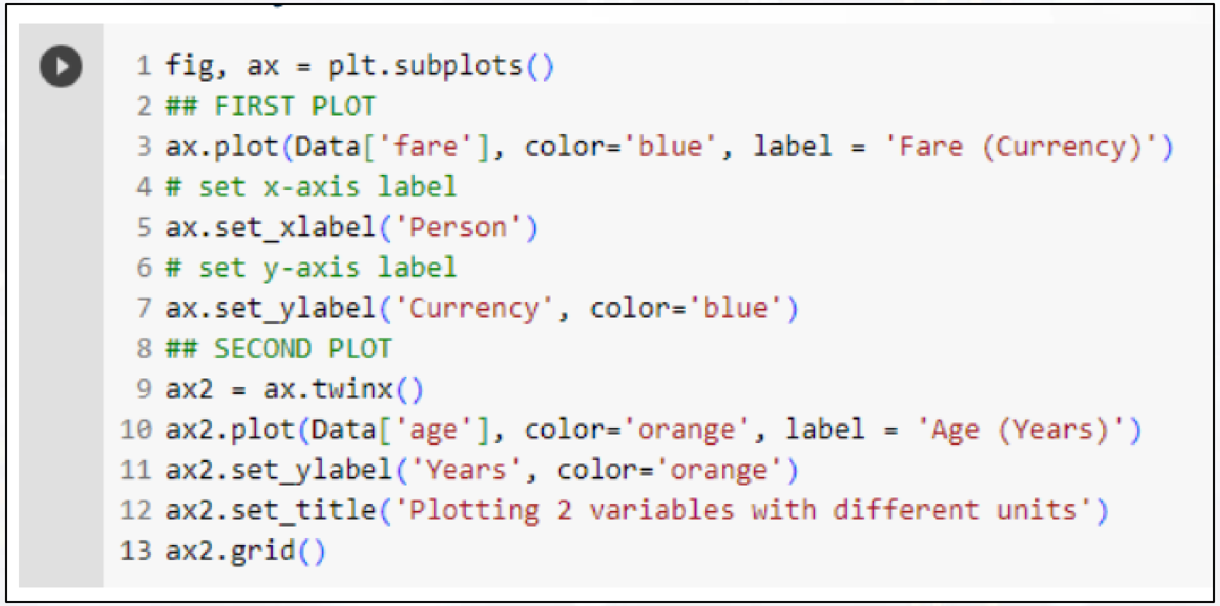


```
import matplotlib.pyplot as plt

plt.plot(Data['fare'],label='Fare (Currency)')
plt.plot(Data['age'],label='Age (Years)')
plt.grid()
plt.xlabel('Person')
plt.ylabel('n/a')
plt.title('Plotting 2 variables.')
plt.legend()
```

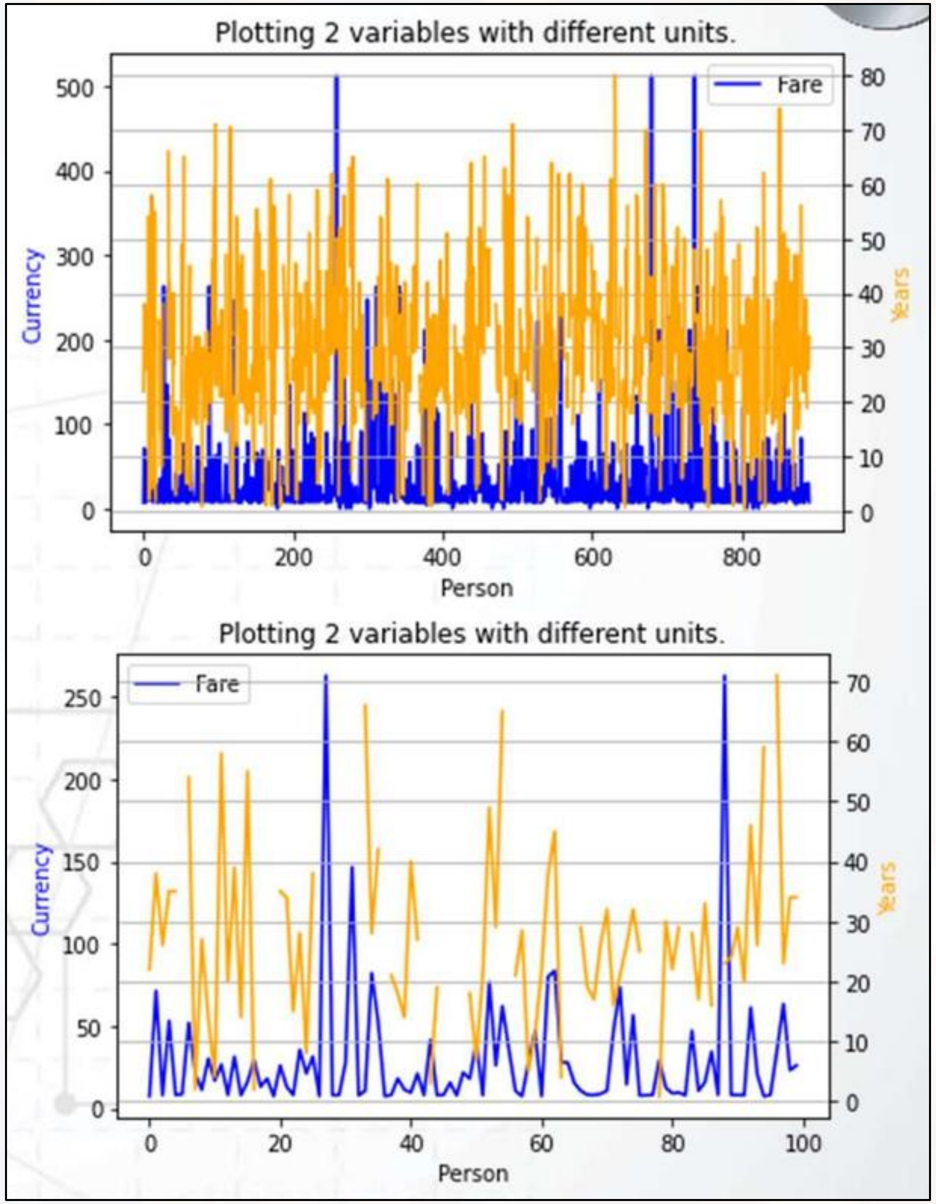


Line Graph

- In situations where you want to visually compare multiple variables with the same x values, but vastly different y ranges, a different y axis can be used for each variable.



```
fig, ax = plt.subplots()
## FIRST PLOT
ax.plot(Data['fare'], color='blue', label = 'Fare (Currency)')
# set x-axis label
ax.set_xlabel('Person')
# set y-axis label
ax.set_ylabel('Currency', color='blue')
## SECOND PLOT
ax2 = ax.twinx()
ax2.plot(Data['age'], color='orange', label = 'Age (Years)')
ax2.set_ylabel('Years', color='orange')
ax2.set_title('Plotting 2 variables with different units')
ax2.grid()
```

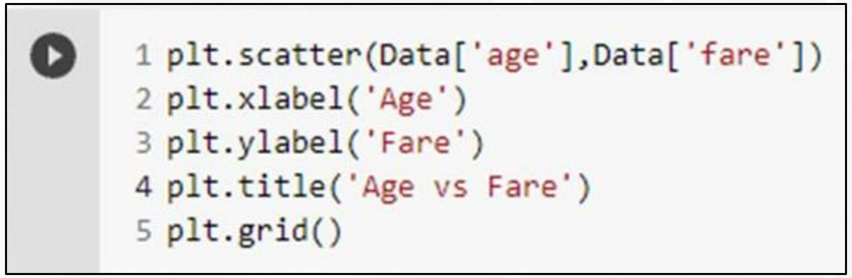


Plotting 2 variables with different units.

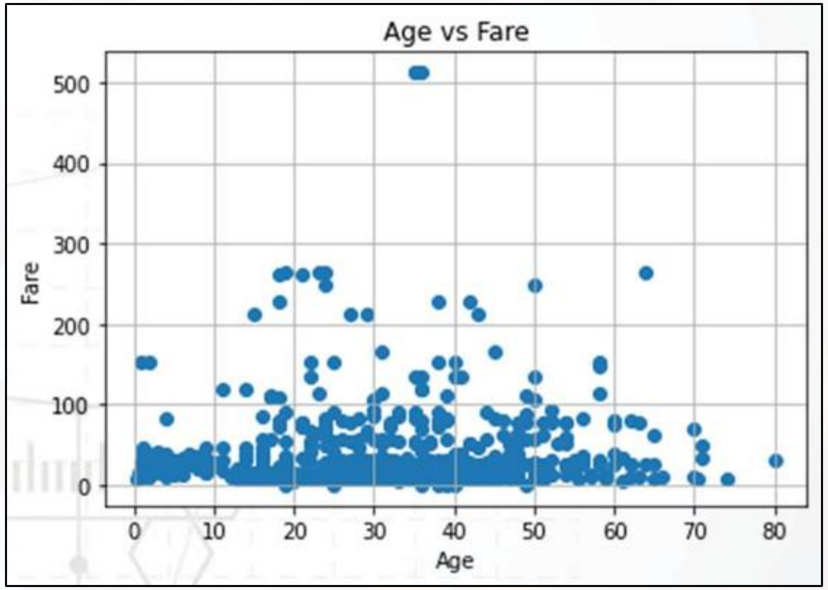
Plotting 2 variables with different units.

Scatterplot

- A Scatterplot is useful to visualize the relationship between 2 variables and show how 2 variables might be related.
- Both datasets must have the same resolution so that a 1-on-1 comparison can be done.
- Scatter plots help us understand **correlation** — how closely two things move together

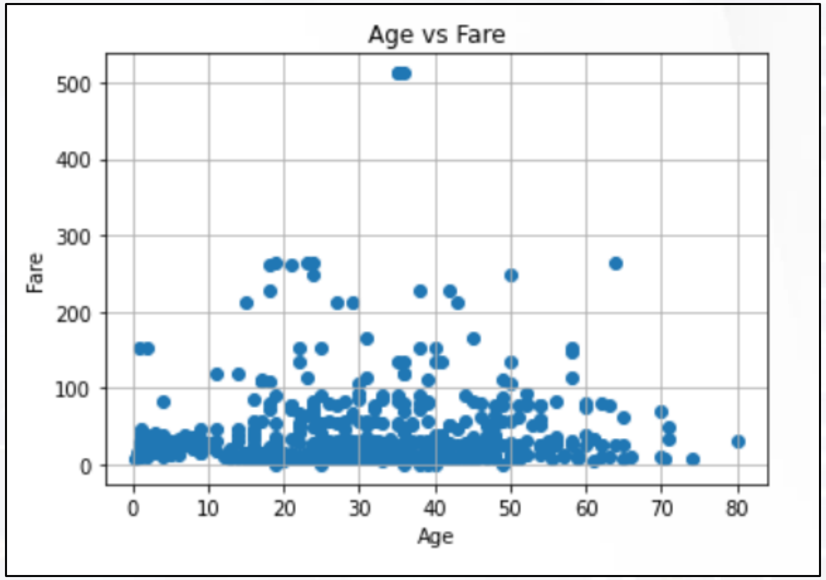


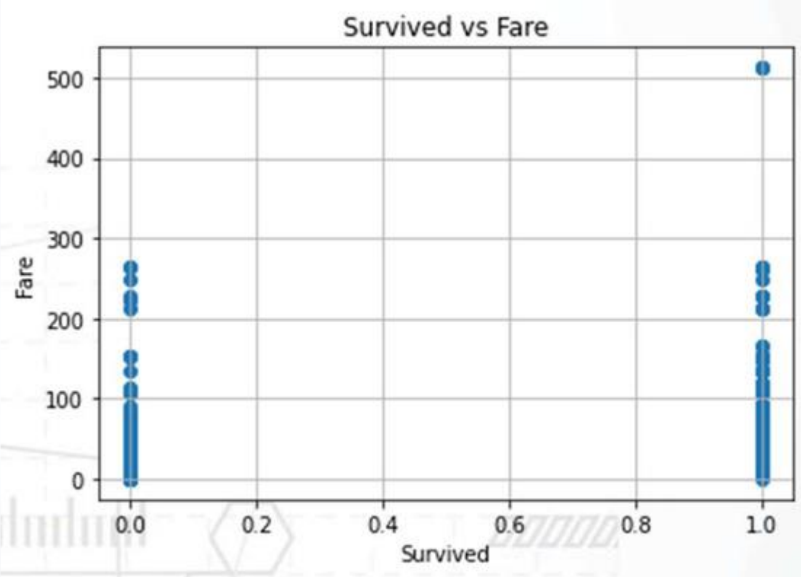
```
plt.scatter(Data['age'],Data['fare'])
plt.xlabel('Age')
plt.ylabel('Fare')
plt.title('Age vs Fare')
plt.grid()
```



Scatterplot

- Comparison: It is important to understand the meaning of the data plotted

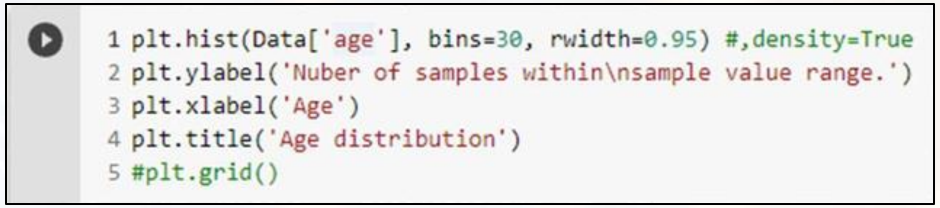




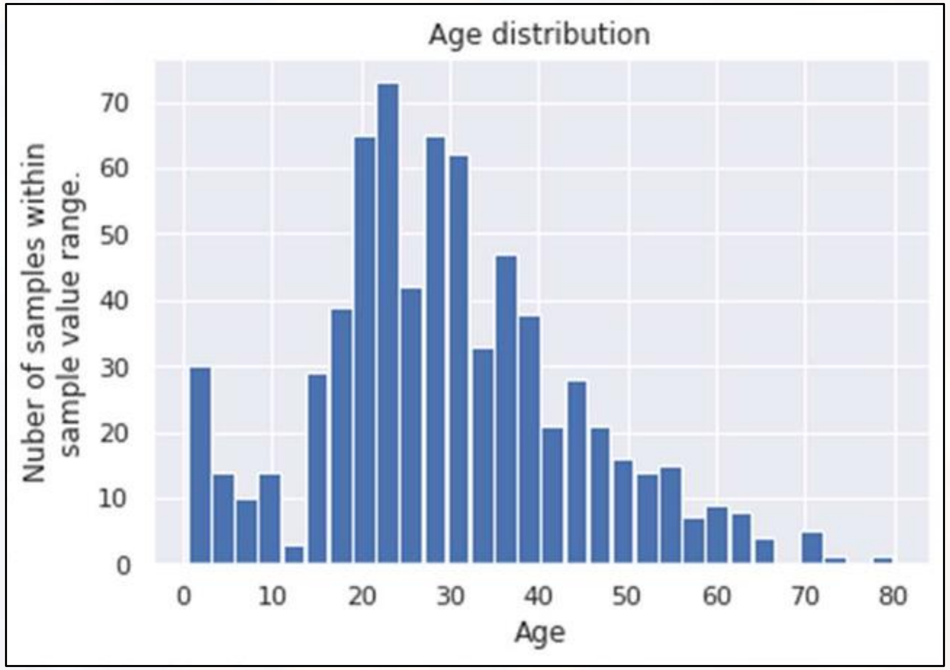
- In this example both graphs show no clear relationship between the 2 variable.

Histogram

- A histogram displays a frequency distribution of a variable, and this can be used to determine if there are any ’intensity’ ranges that appears more (or less) in the sample set.
- To show the **distribution** of data and also to group data into ranges and count how many values are in each.
- It helps us understand the **shape of the data** — like whether it’s balanced, or has more values on the high or low side.

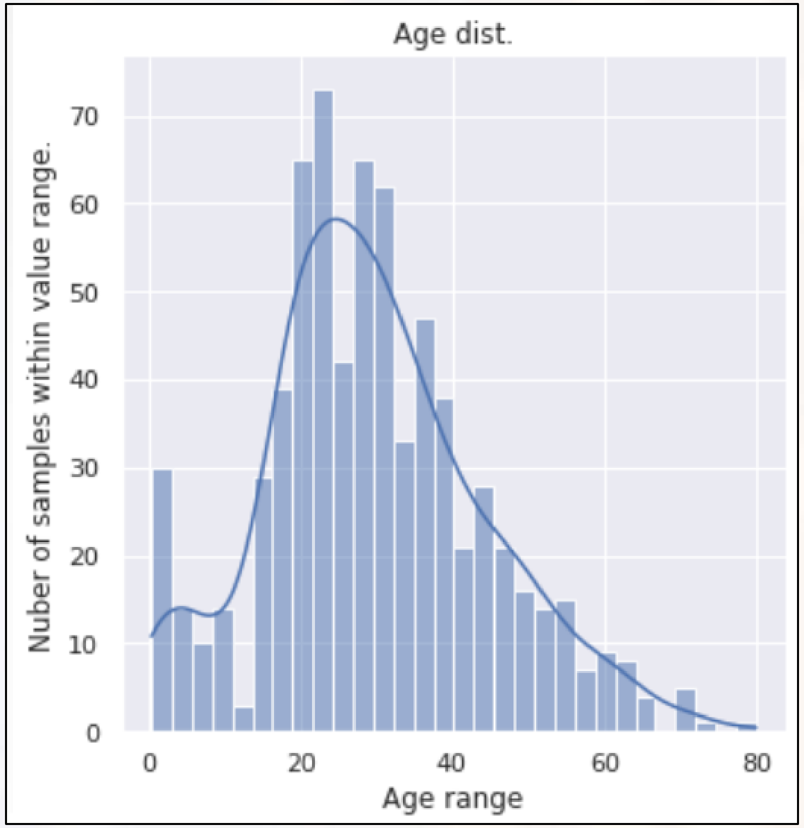


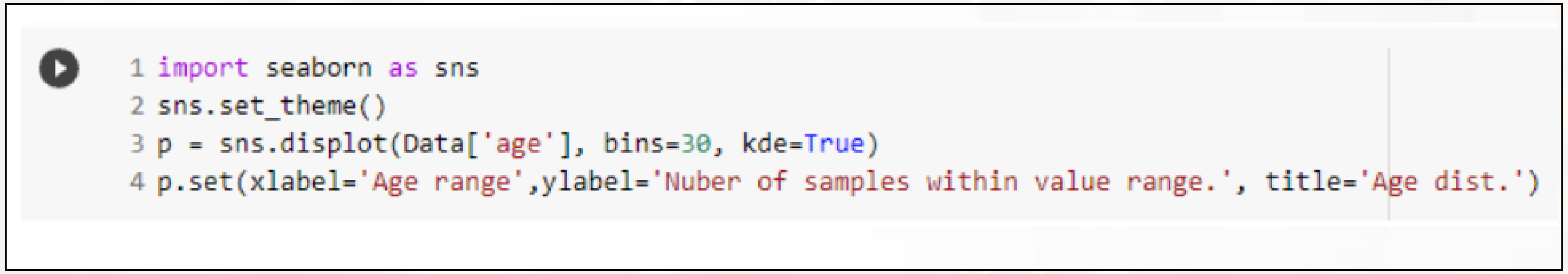
```python
plt.hist(Data['age'], bins=30, rwidth=0.95) #,density=True
plt.ylabel('Nuber of samples within\nsample value range.')
plt.xlabel('Age')
plt.title('Age distribution')
#plt.grid()
```



Histogram

- Seaborn can also be used for plotting and is in many cases simpler to use than matplotlib.
- The number of bins (distribution resolution) can be set using ‘bins’ and the ‘kde’ (kernel density [probability] estimate) can be calculated and added to the graph using kde = *True*.

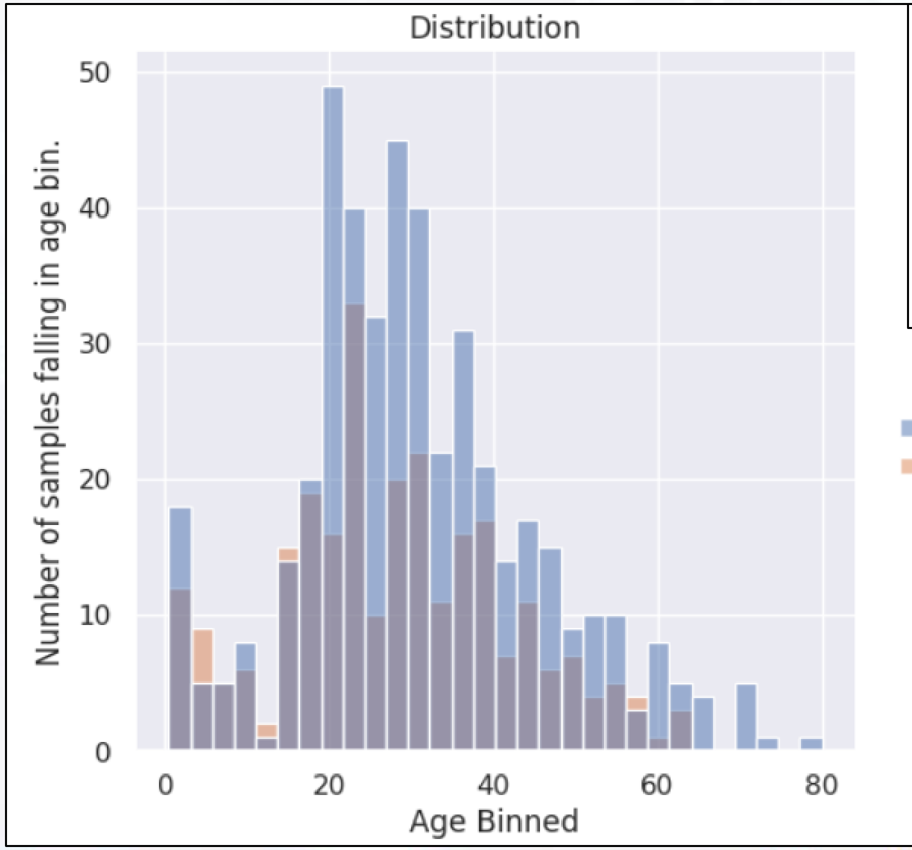


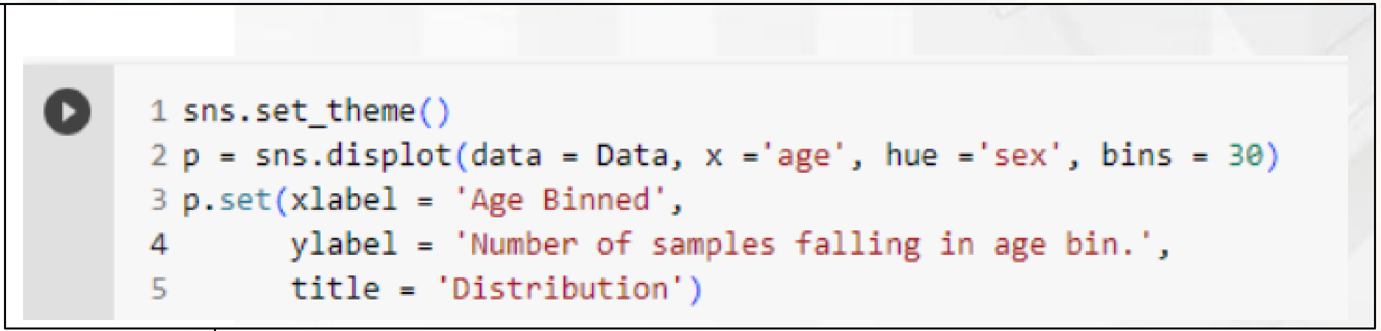


```python
import seaborn as sns
sns.set_theme()
p = sns.displot(Data['age'], bins=30, kde=True)
p.set(xlabel='Age range',ylabel='Nuber of samples within value range.', title='Age dist.')
```

Histogram

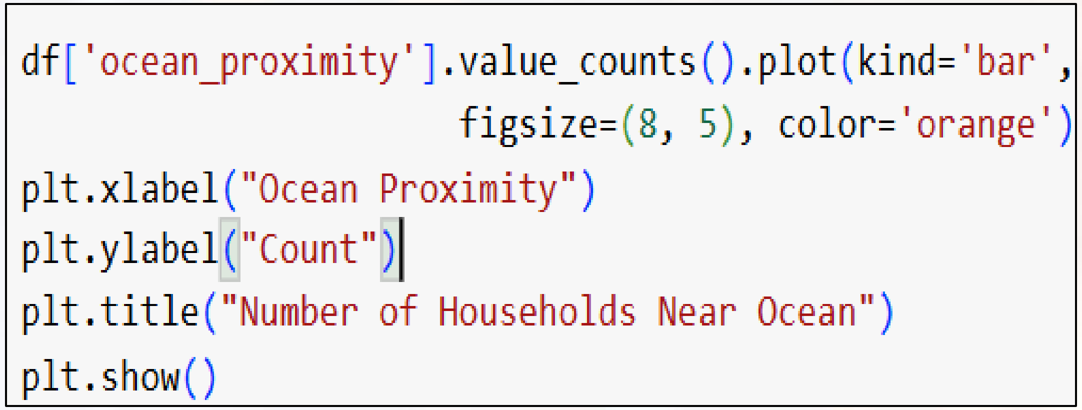
- Seaborn can also plot dataframes directly (multiple variables) without any conversion needed.



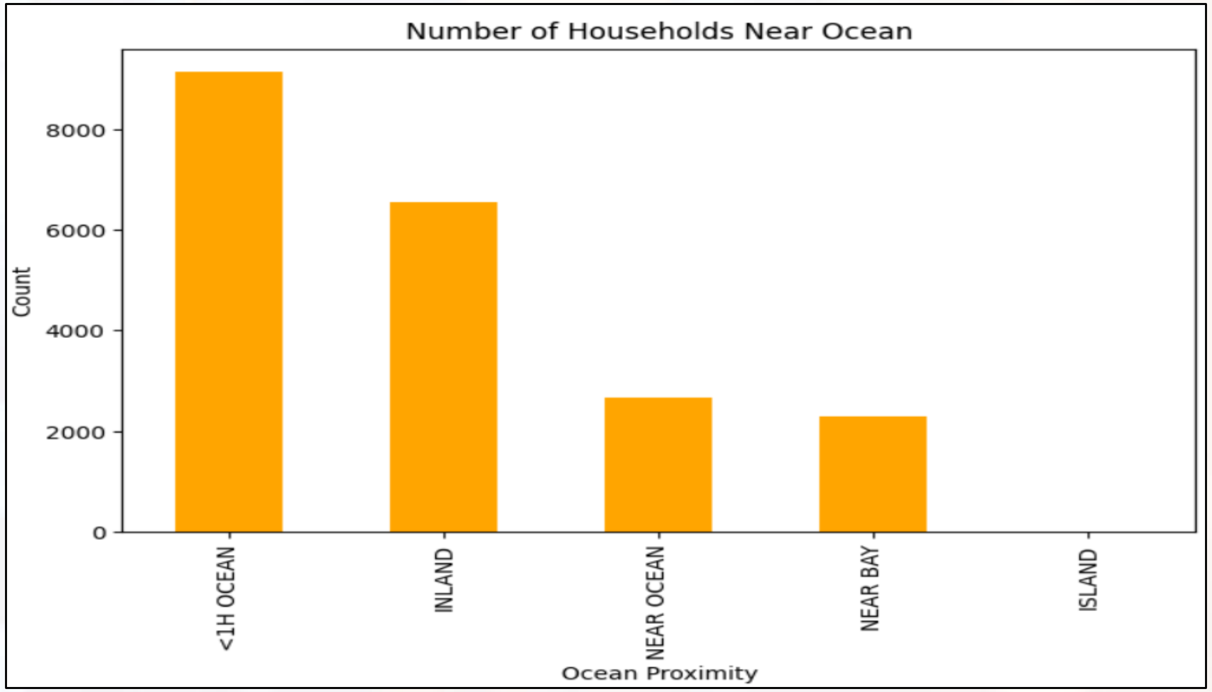

```python
sns.set_theme()
p = sns.displot(data = Data, x ='age', hue ='sex', bins = 30)
p.set(xlabel = 'Age Binned',
      ylabel = 'Number of samples falling in age bin.',
      title = 'Distribution')
```

Bar Graph

- We use bar graphs to **compare things side by side** — like how much different neighborhoods earn, or how many homes were sold in each city.
- Bar graphs make it easy to see **which group is the biggest or smallest**.
- You can also see patterns — like if one group is far ahead of the rest, or if values are close together. It helps make quick decisions or spot trends in simple data.

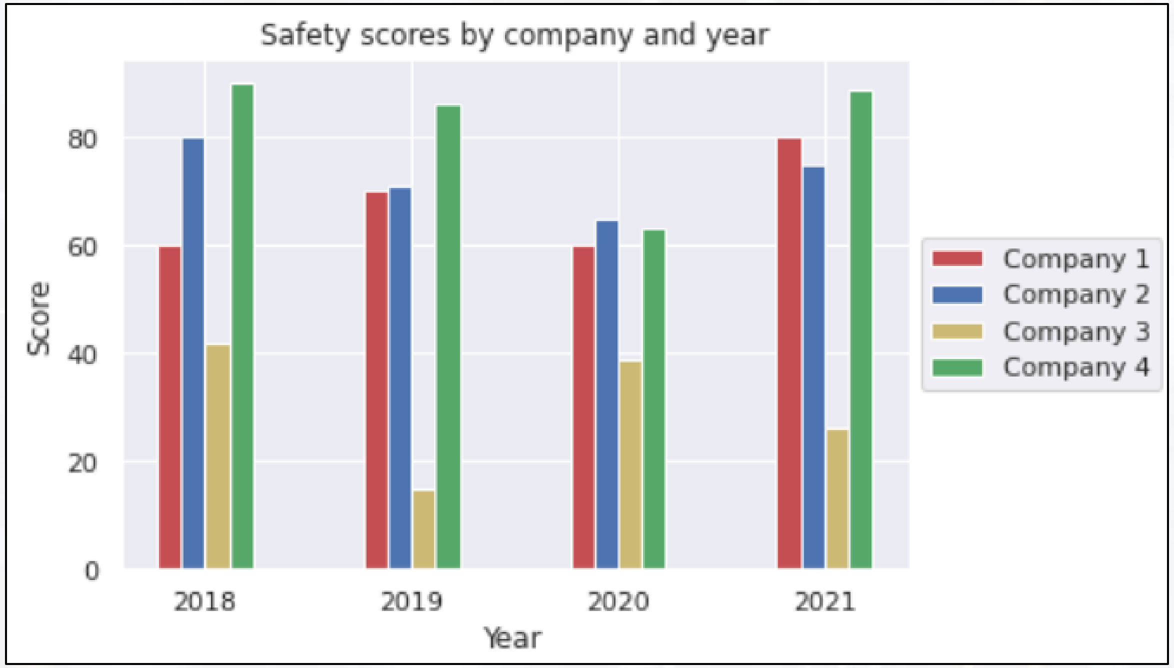


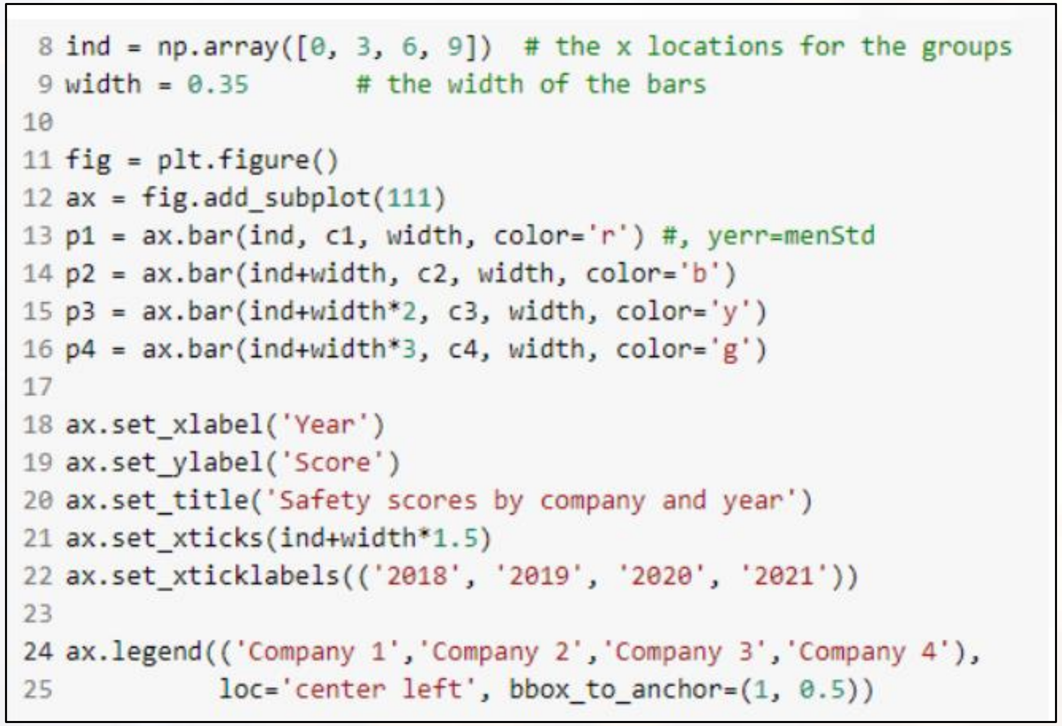
```
df['ocean_proximity'].value_counts().plot(kind='bar',
                                      figsize=(8, 5), color='orange')
plt.xlabel("Ocean Proximity")
plt.ylabel("Count")
plt.title("Number of Households Near Ocean")
plt.show()
```



Bar Graphs - Stacked

- This can be used to display categorical data eg. When there are several variables with multiple attributes that the user wants to compare.

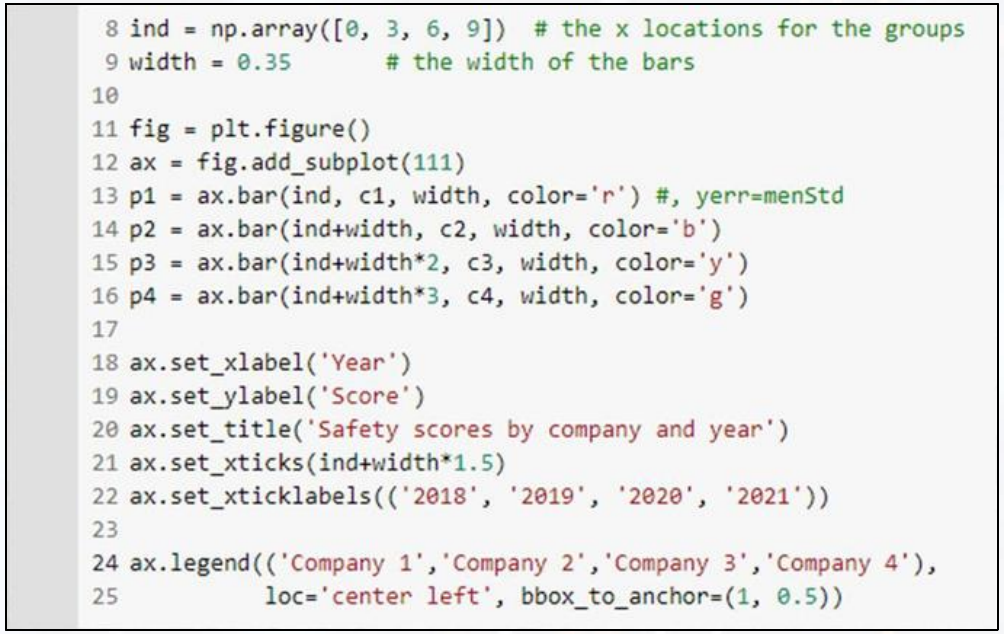




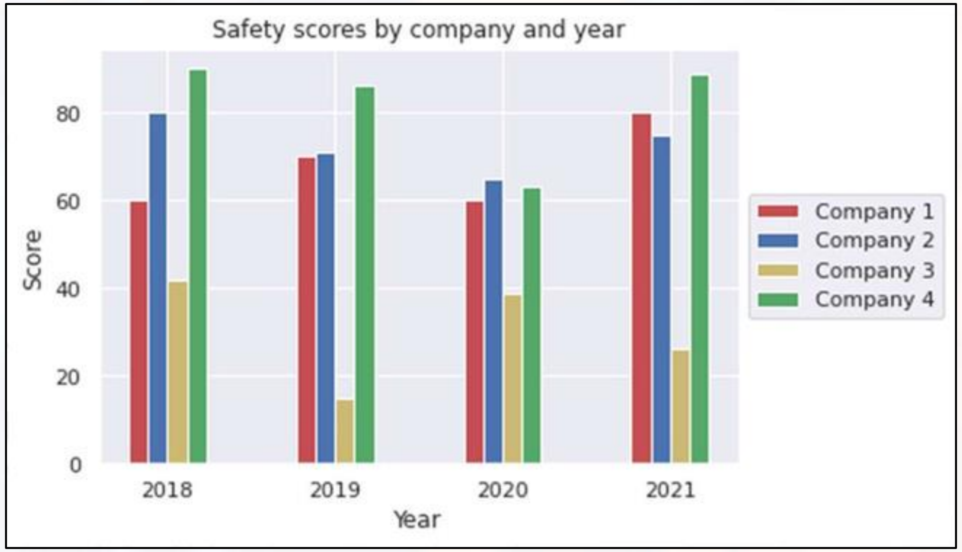
```python
ind = np.array([0, 3, 6, 9])  # the x locations for the groups
width = 0.35       # the width of the bars

fig = plt.figure()
ax = fig.add_subplot(111)
p1 = ax.bar(ind, c1, width, color='r') #, yerr=menStd
p2 = ax.bar(ind+width, c2, width, color='b')
p3 = ax.bar(ind+width*2, c3, width, color='y')
p4 = ax.bar(ind+width*3, c4, width, color='g')

ax.set_xlabel('Year')
ax.set_ylabel('Score')
ax.set_title('Safety scores by company and year')
ax.set_xticks(ind+width*1.5)
ax.set_xticklabels(('2018', '2019', '2020', '2021'))

ax.legend(('Company 1','Company 2','Company 3','Company 4'),
          loc='center left', bbox_to_anchor=(1, 0.5))
```

Bar Graphs – Side by Side

- There are different options for how to display a bar graph e.g., stacked, on the side axis, side-by- side, etc.

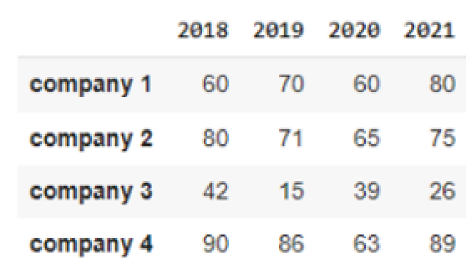


```python
ind = np.array([0, 3, 6, 9])  # the x locations for the groups
width = 0.35       # the width of the bars

fig = plt.figure()
ax = fig.add_subplot(111)
p1 = ax.bar(ind, c1, width, color='r') #, yerr=menStd
p2 = ax.bar(ind+width, c2, width, color='b')
p3 = ax.bar(ind+width*2, c3, width, color='y')
p4 = ax.bar(ind+width*3, c4, width, color='g')

ax.set_xlabel('Year')
ax.set_ylabel('Score')
ax.set_title('Safety scores by company and year')
ax.set_xticks(ind+width*1.5)
ax.set_xticklabels(('2018', '2019', '2020', '2021'))

ax.legend(('Company 1','Company 2','Company 3','Company 4'),
          loc='center left', bbox_to_anchor=(1, 0.5))
```

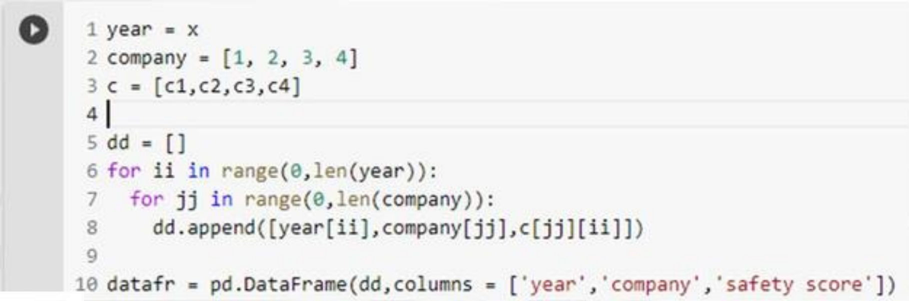


Bar Graph – Side by Side

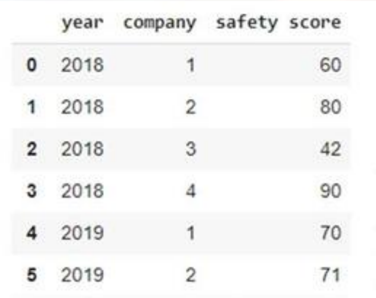
- Sometimes (depending on the input data format) seaborn is not the easiest option even if, at first glance, it looks like it…
- Even if it looks simple, this took more time to figure out what the data format should be than plotting the previous example.



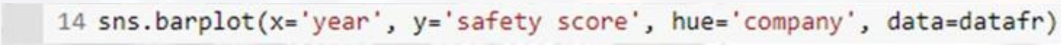
	2018	2019	2020	2021
company 1	60	70	60	80
company 2	80	71	65	75
company 3	42	15	39	26
company 4	90	86	63	89



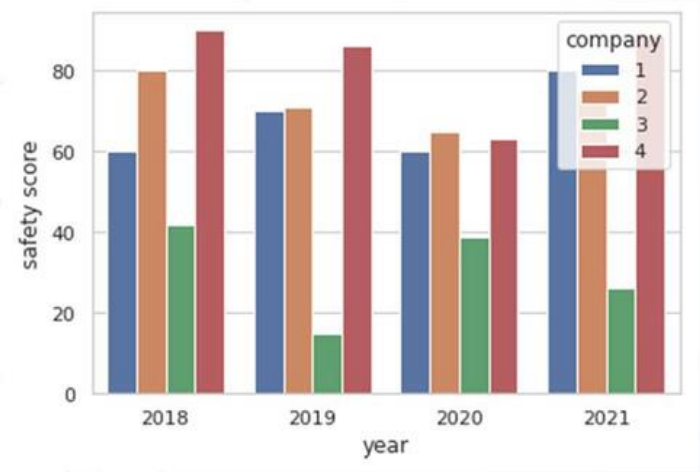
```python
year = x
company = [1, 2, 3, 4]
c = [c1,c2,c3,c4]

dd = []
for ii in range(0,len(year)):
  for jj in range(0,len(company)):
    dd.append([year[ii],company[jj],c[jj][ii]])

datafr = pd.DataFrame(dd,columns = ['year','company','safety score'])
```



	year	company	safety score
0	2018	1	60
1	2018	2	80
2	2018	3	42
3	2018	4	90
4	2019	1	70
5	2019	2	71

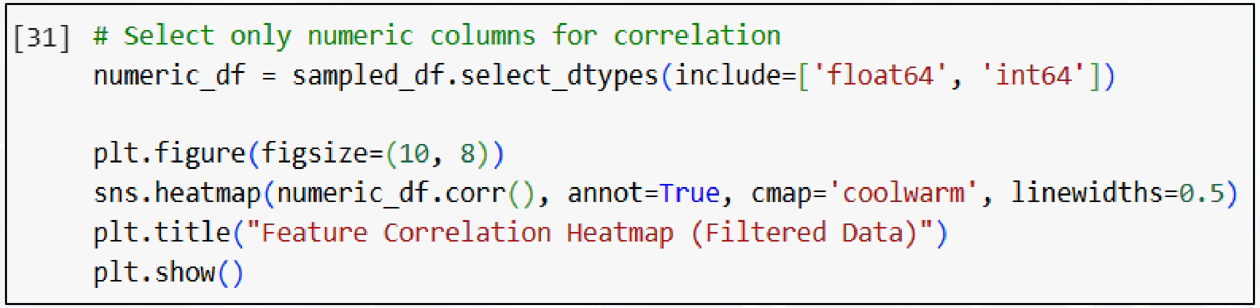


```python
sns.barplot(x='year', y='safety score', hue='company', data=datafr)
```

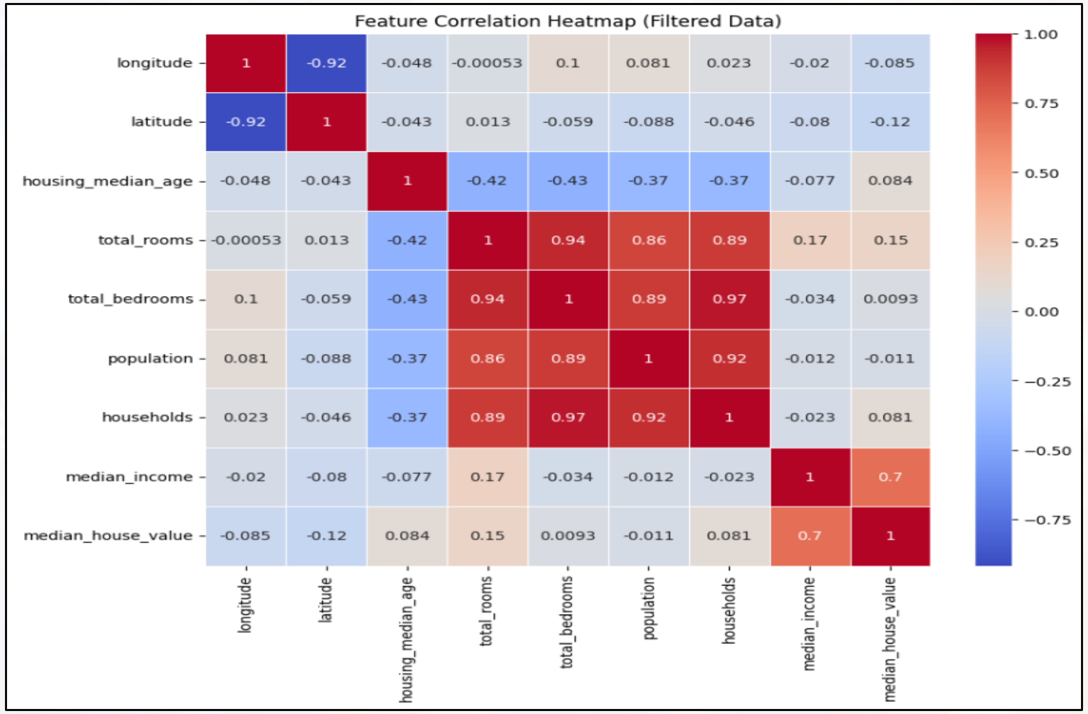


Heatmap

- A heatmap allows for data to be visualized in 2 dimensions with the magnitude indicated by the colour.
  - Darker or brighter colors mean higher or lower numbers.
- It helps show patterns in large amounts of data.



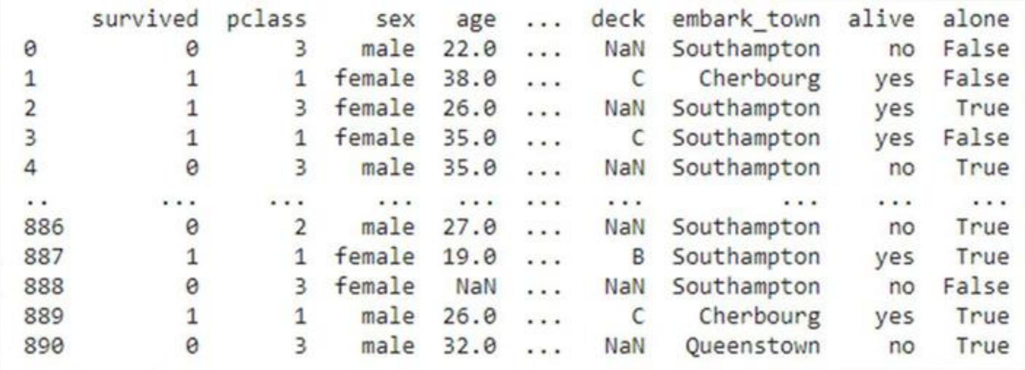
```
[31] # Select only numeric columns for correlation
     numeric_df = sampled_df.select_dtypes(include=['float64', 'int64'])

     plt.figure(figsize=(10, 8))
     sns.heatmap(numeric_df.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
     plt.title("Feature Correlation Heatmap (Filtered Data)")
     plt.show()
```



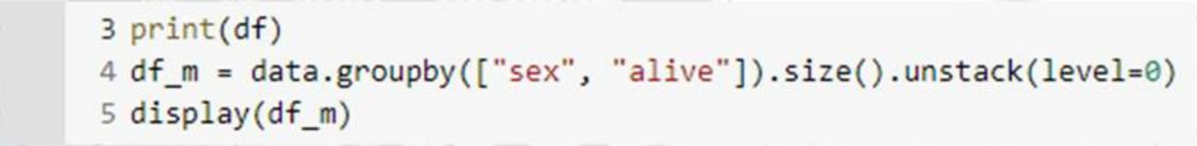
	longitude	latitude	housing_median_age	total_rooms	total_bedrooms	population	households	median_income	median_house_value
longitude	1	-0.92	-0.048	-0.00053	0.1	0.081	0.023	-0.02	-0.085
latitude	-0.92	1	-0.043	0.013	-0.059	-0.088	-0.046	-0.08	-0.12
housing_median_age	-0.048	-0.043	1	-0.42	-0.43	-0.37	-0.37	-0.077	0.084
total_rooms	-0.00053	0.013	-0.42	1	0.94	0.86	0.89	0.17	0.15
total_bedrooms	0.1	-0.059	-0.43	0.94	1	0.89	0.97	-0.034	0.0093
population	0.081	-0.088	-0.37	0.86	0.89	1	0.92	-0.012	-0.011
households	0.023	-0.046	-0.37	0.89	0.97	0.92	1	-0.023	0.081
median_income	-0.02	-0.08	-0.077	0.17	-0.034	-0.012	-0.023	1	0.7
median_house_value	-0.085	-0.12	0.084	0.15	0.0093	-0.011	0.081	0.7	1

Heatmap

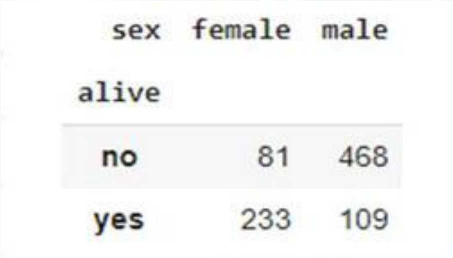
- Another method to get the dataframe data in the needed format is to use ‘groupby’ and ‘unstack’ (count the number of instances).



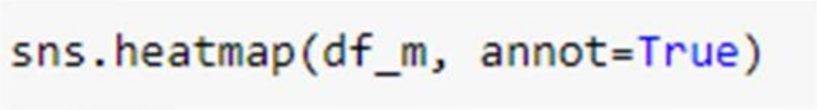
```
     survived  pclass     sex   age  ...  deck  embark_town  alive  alone
0           0       3    male  22.0  ...   NaN  Southampton     no  False
1           1       1  female  38.0  ...     C    Cherbourg    yes  False
2           1       3  female  26.0  ...   NaN  Southampton    yes   True
3           1       1  female  35.0  ...     C  Southampton    yes  False
4           0       3    male  35.0  ...   NaN  Southampton     no   True
..        ...     ...     ...   ...  ...   ...          ...    ...    ...
886         0       2    male  27.0  ...   NaN  Southampton     no   True
887         1       1  female  19.0  ...     B  Southampton    yes   True
888         0       3  female   NaN  ...   NaN  Southampton     no  False
889         1       1    male  26.0  ...     C    Cherbourg    yes   True
890         0       3    male  32.0  ...   NaN   Queenstown     no   True
```



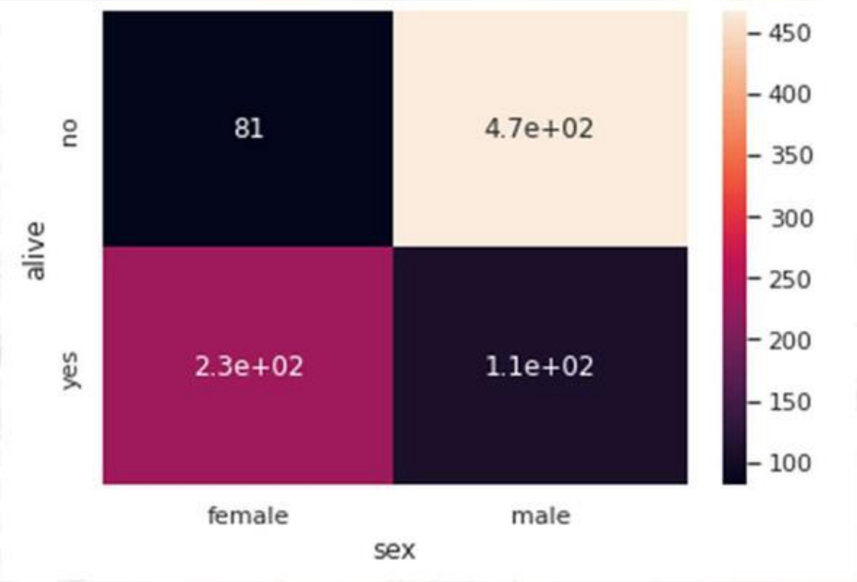
```
print(df)
df_m = data.groupby(["sex", "alive"]).size().unstack(level=0)
display(df_m)
```



sex	female	male
alive		
no	81	468
yes	233	109

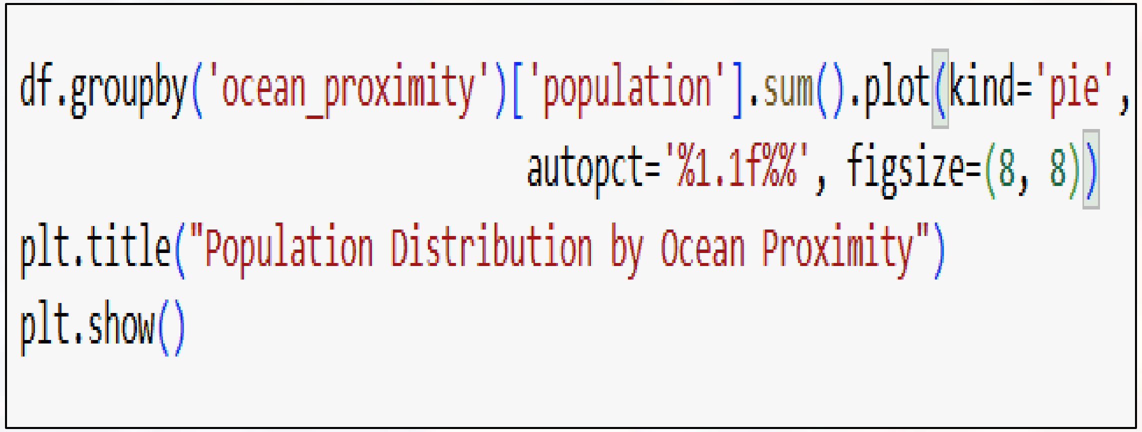


```
sns.heatmap(df_m, annot=True)
```

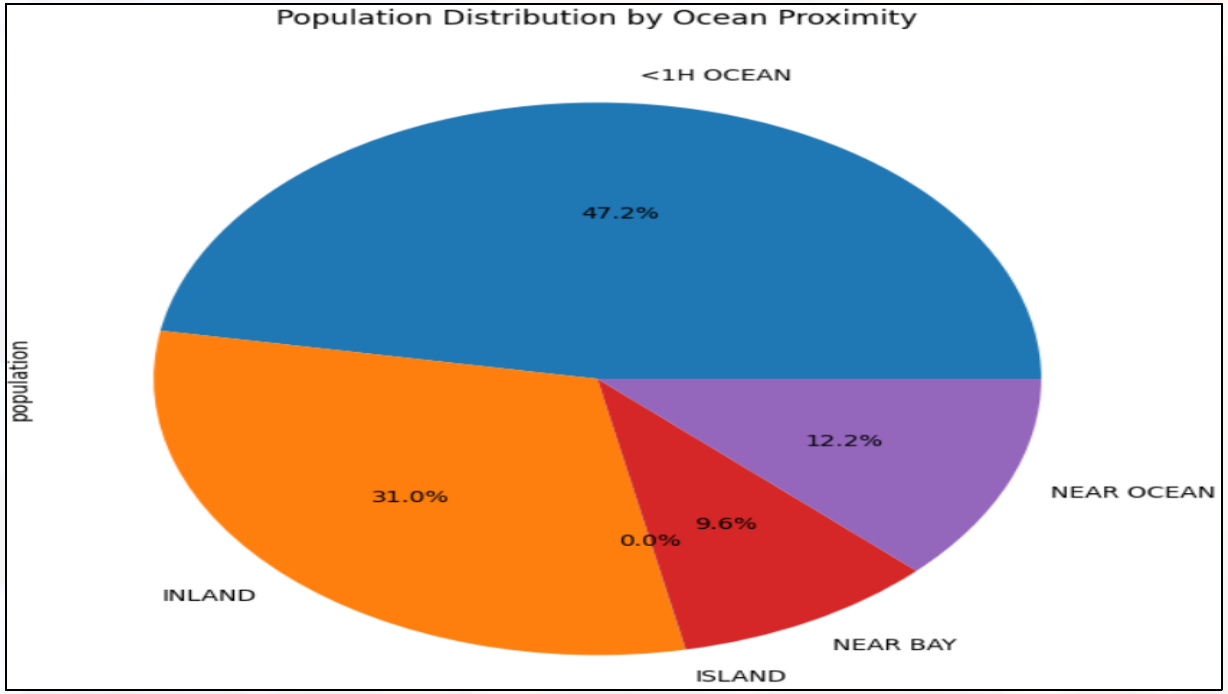


Pie Chart

- Used to show the share of the total for each variable or class.
- Each slice represents a percentage or part of a group.
- Helps see how different parts compare to each other.

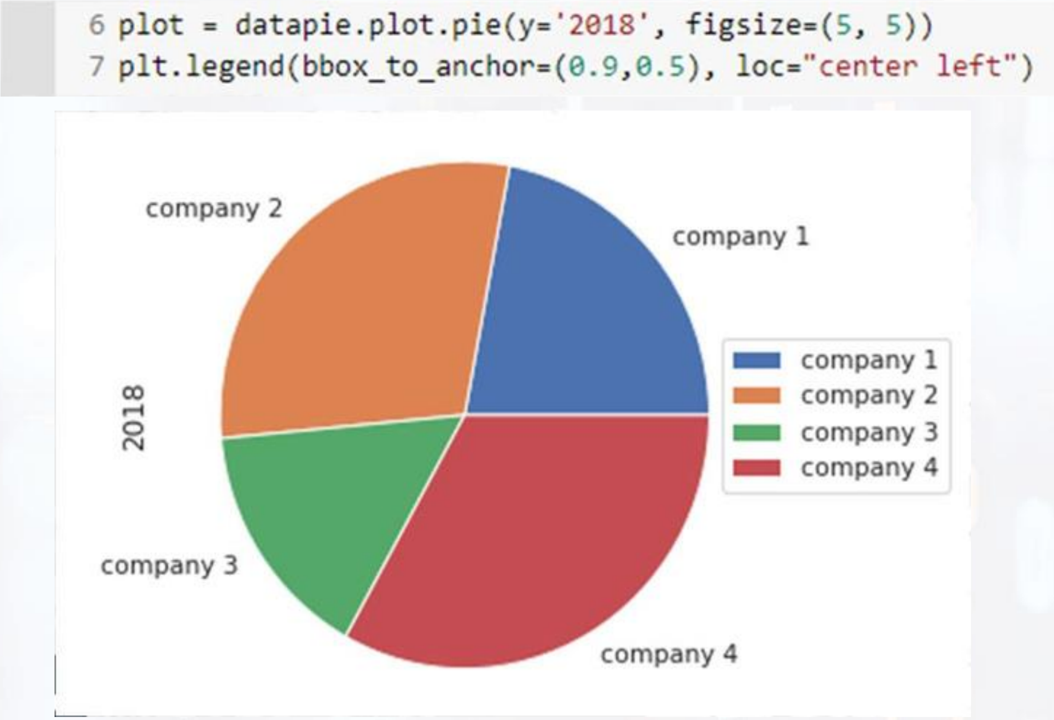


```
df.groupby('ocean_proximity')['population'].sum().plot(kind='pie',
                                                       autopct='%1.1f%%', figsize=(8, 8))
plt.title("Population Distribution by Ocean Proximity")
plt.show()
```

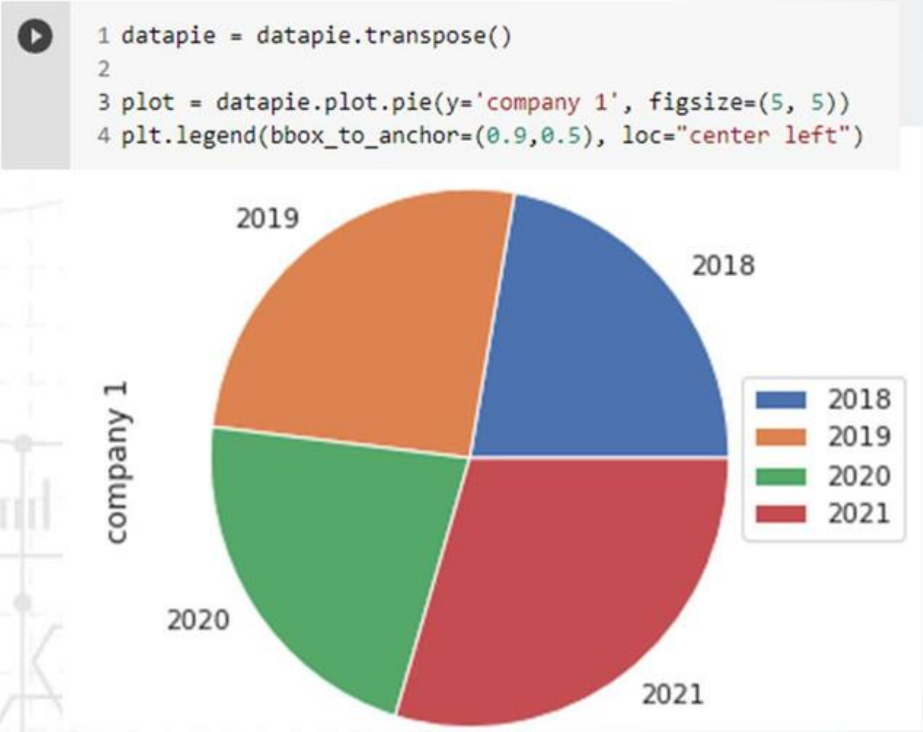


Pie Char

- Similar to heatmap, ‘groupby’ and ‘unstack’ can be used to count the number of instances.



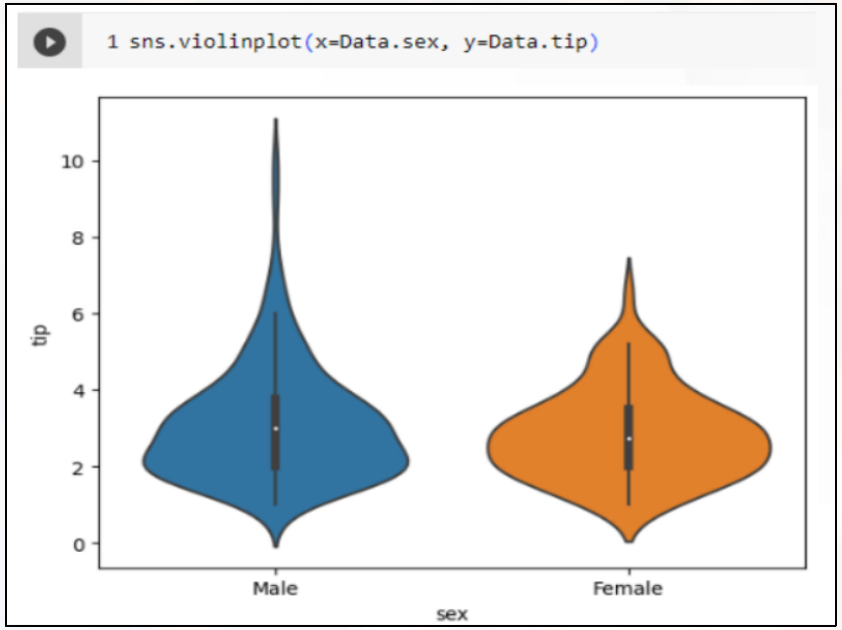
```
plot = datapie.plot.pie(y='2018', figsize=(5, 5))
plt.legend(bbox_to_anchor=(0.9,0.5), loc="center left")
```



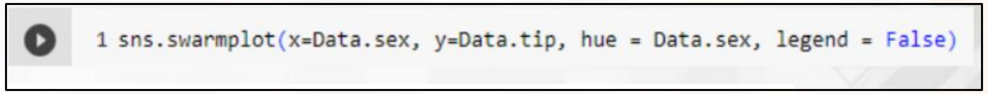
```
datapie = datapie.transpose()

plot = datapie.plot.pie(y='company 1', figsize=(5, 5))
plt.legend(bbox_to_anchor=(0.9,0.5), loc="center left")
```

Violin / Swarm Plot

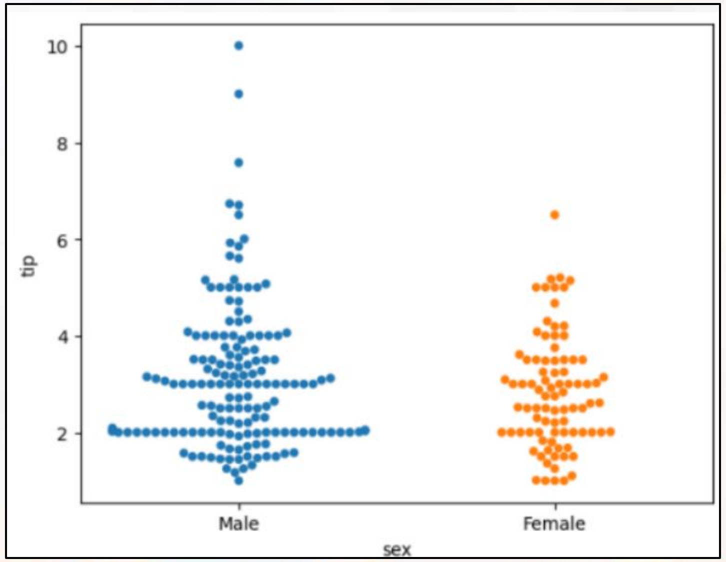
- A violin plot shows how data is spread out – whether data is evenly spread or skewed.
- To compare how data is spread across groups.

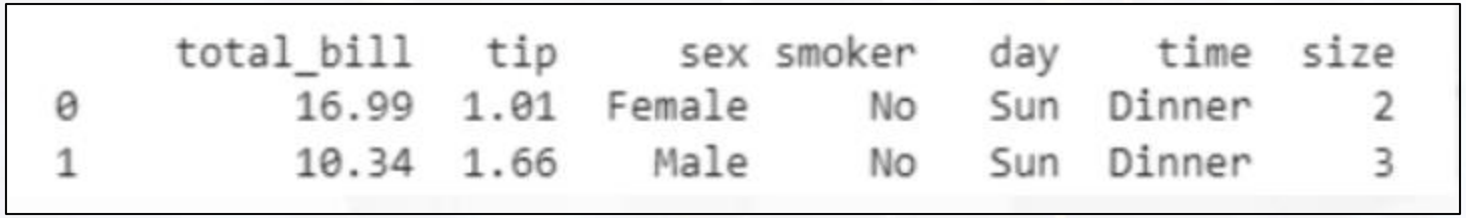


```
sns.violinplot(x=Data.sex, y=Data.tip)
```



```
sns.swarmplot(x=Data.sex, y=Data.tip, hue = Data.sex, legend = False)
```

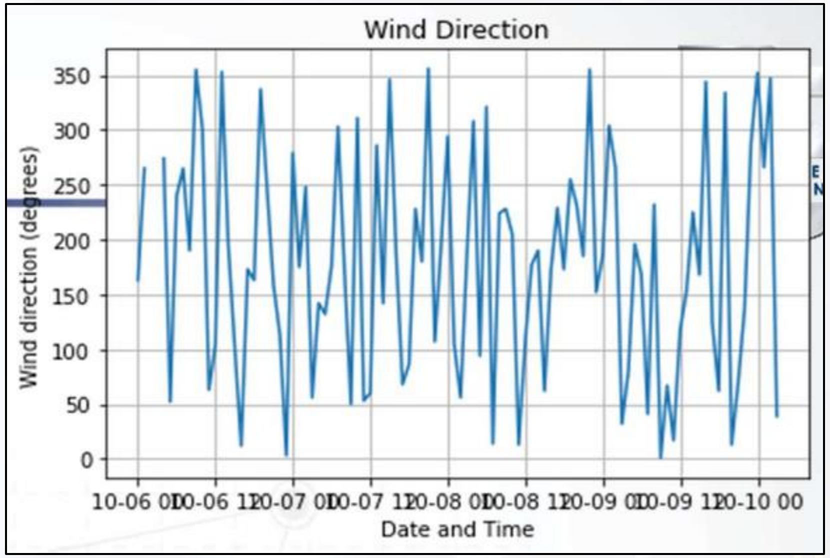


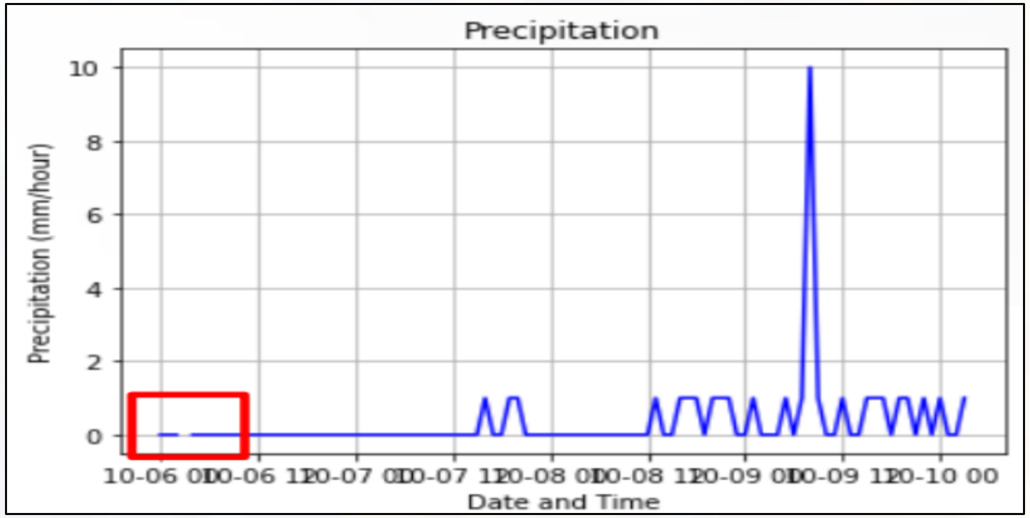


```
   total_bill   tip     sex smoker  day    time  size
0       16.99  1.01  Female     No  Sun  Dinner     2
1       10.34  1.66    Male     No  Sun  Dinner     3
```

Functions – Scrutinizing Datasets

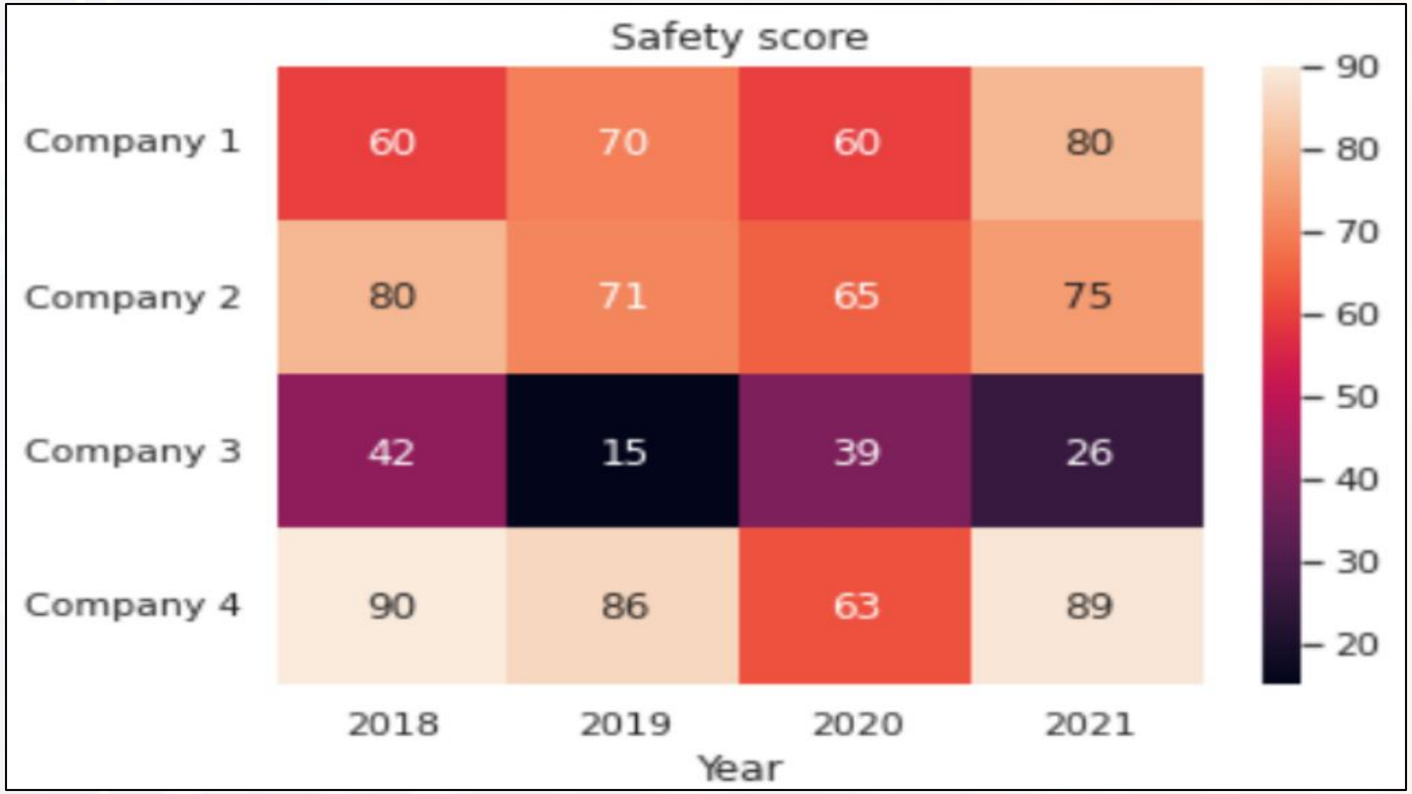
- Data quality can sometimes be determined by plotting the datasets.
- Missing values or outliers can be visually identified as well as the general frequency and location.
- E.g. in the figure there is a missing value close to the beginning of the dataset and there is one entry that is an order of magnitude different from the rest. This may be a correct value or a mistake in the dataset (in many cases a domain expert would have to be consulted).





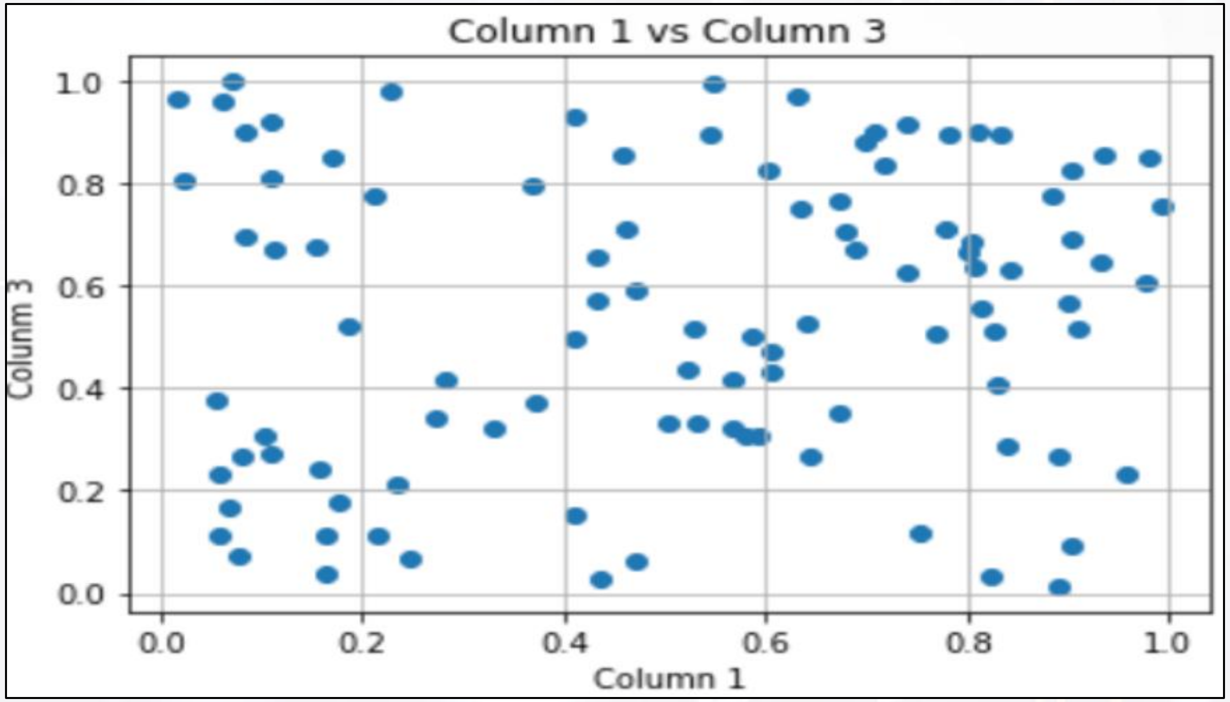
Understanding Datasets

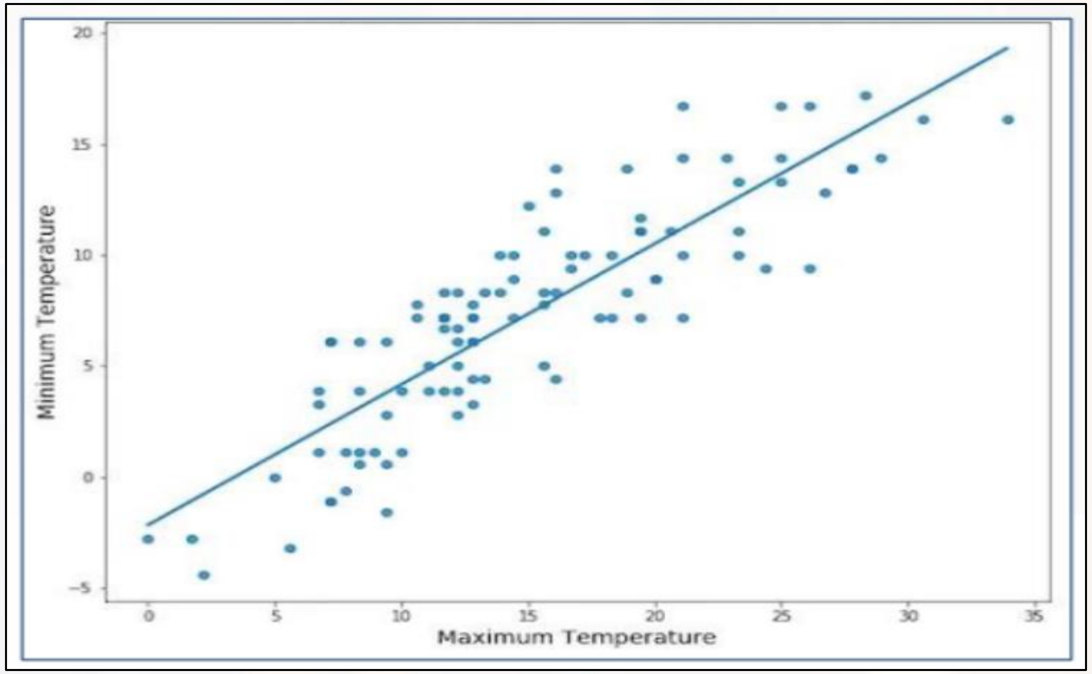
- Information that is not always obvious when just looking at the data.
- Some features or trends can be highlighted using visualization.
- e.g., From this heat map there was, in general, poor safety in 2020 and Company 3 is a place where you would not want to work.



Pattern Recognition

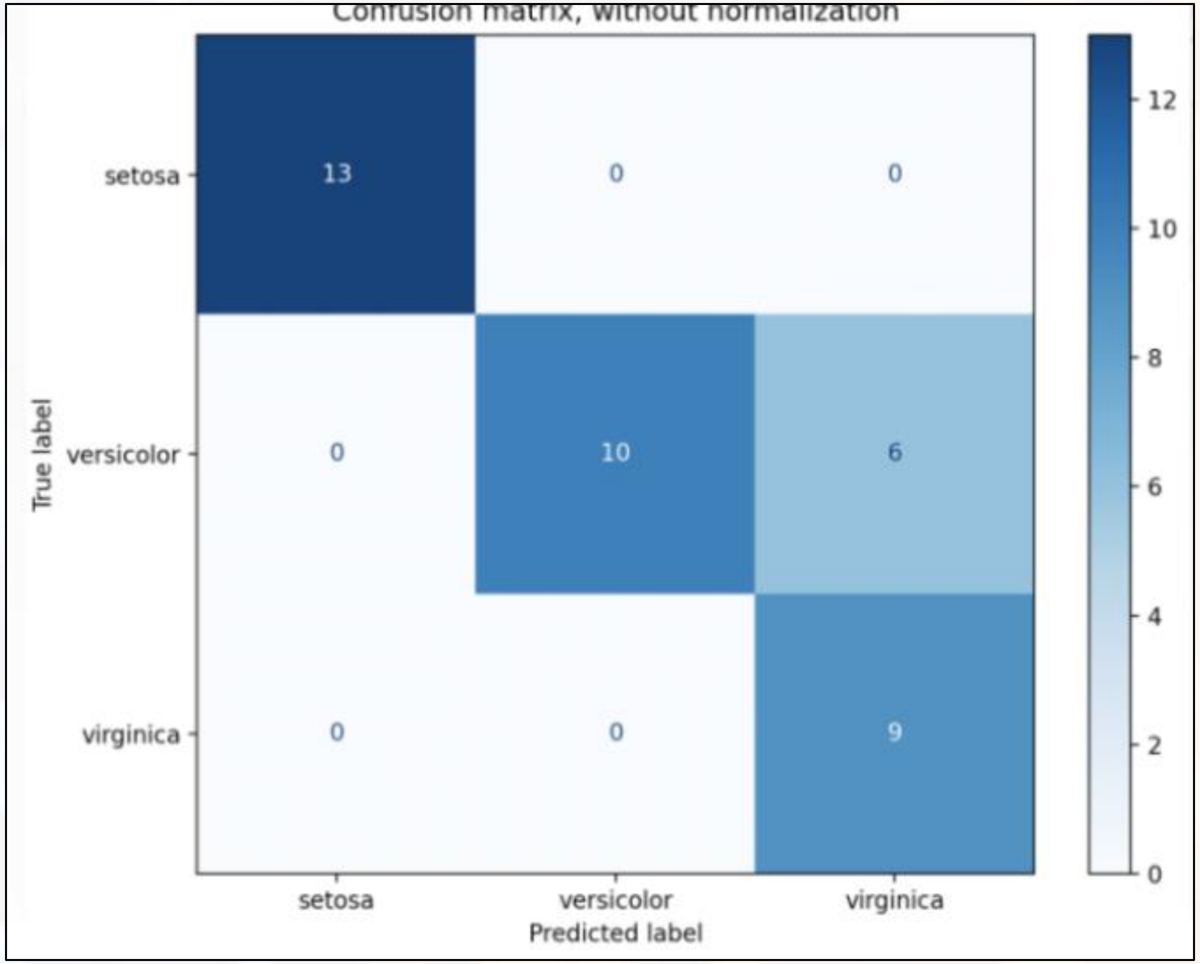
- Clearly there is no obvious relationship between Column 1 and Column 3 in Fig 1 – which is good since it's randomized data…
- But in the second figure there is a clear linear relationship.





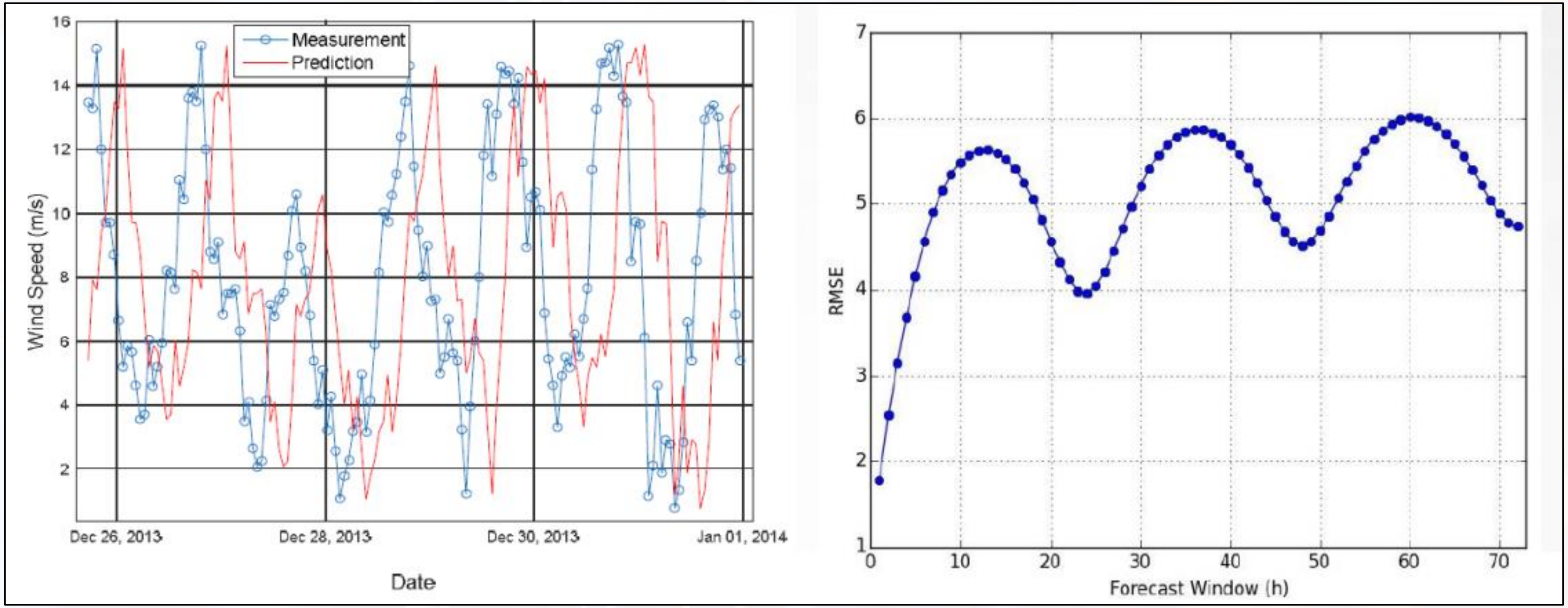
Results and Concepts Communication

- A user can better understand results from analytics and models when it is visualized.
- In cases where the actual values are of importance, the data can be displayed in an accompanying table as well.
- E.g., a confusion matrix is a very handy visual tool to help a reader understand results from a classification model.



Results and Concepts Communication

- There are also cases where it would make more sense to communicate a concept using a graph/diagram. Example:



Source: [2]

Summary

- Understanding your dataset is the first step to creating meaningful visuals
- Choosing the right type of graph depends on the message you want to show
- Well-prepared and clean data makes your visualizations more accurate and powerful
- Focus on clarity — visuals should be easy to read and support your story
- Good graphs turn raw data into insights that inform, explain, and persuade

Final Thoughts

- Never forget to label you axis, a label can change the way a graph is interpreted, and the reader or audience member will not be able to read it properly (or even yourself in 6 months' time…) if unlabelled.
- When displaying a figure or graph in a document (such as a report or thesis), always make sure to refer to it in the text and explain what is happening in the figure.
- On the other hand, in presentations, it is better to have less text. So that you would not have a written explanation of the graph on the slide, but rather explain it verbally (there are always exceptions e.g., if there is an expectation that it will be revisited at a later stage).
- When you use visualization, you should still know WHAT visualization is optimal for your problem or the concept you want to convey.
- Numerous visualization methods, as well as ways to get the graph you want, is just Google or Bing search away

References

- [1] - https://www.sprintzeal.com/blog/fundamentals-of-data-visualization
- [2] - https://researchspace.csir.co.za/dspace/handle/10204/9368

Questions???
Let’s do the Data Visualization practical

nicis
NATIONAL INTEGRATED
CYBERINFRASTRUCTURE SYSTEM
DIRISA