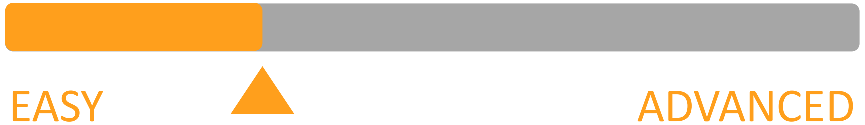
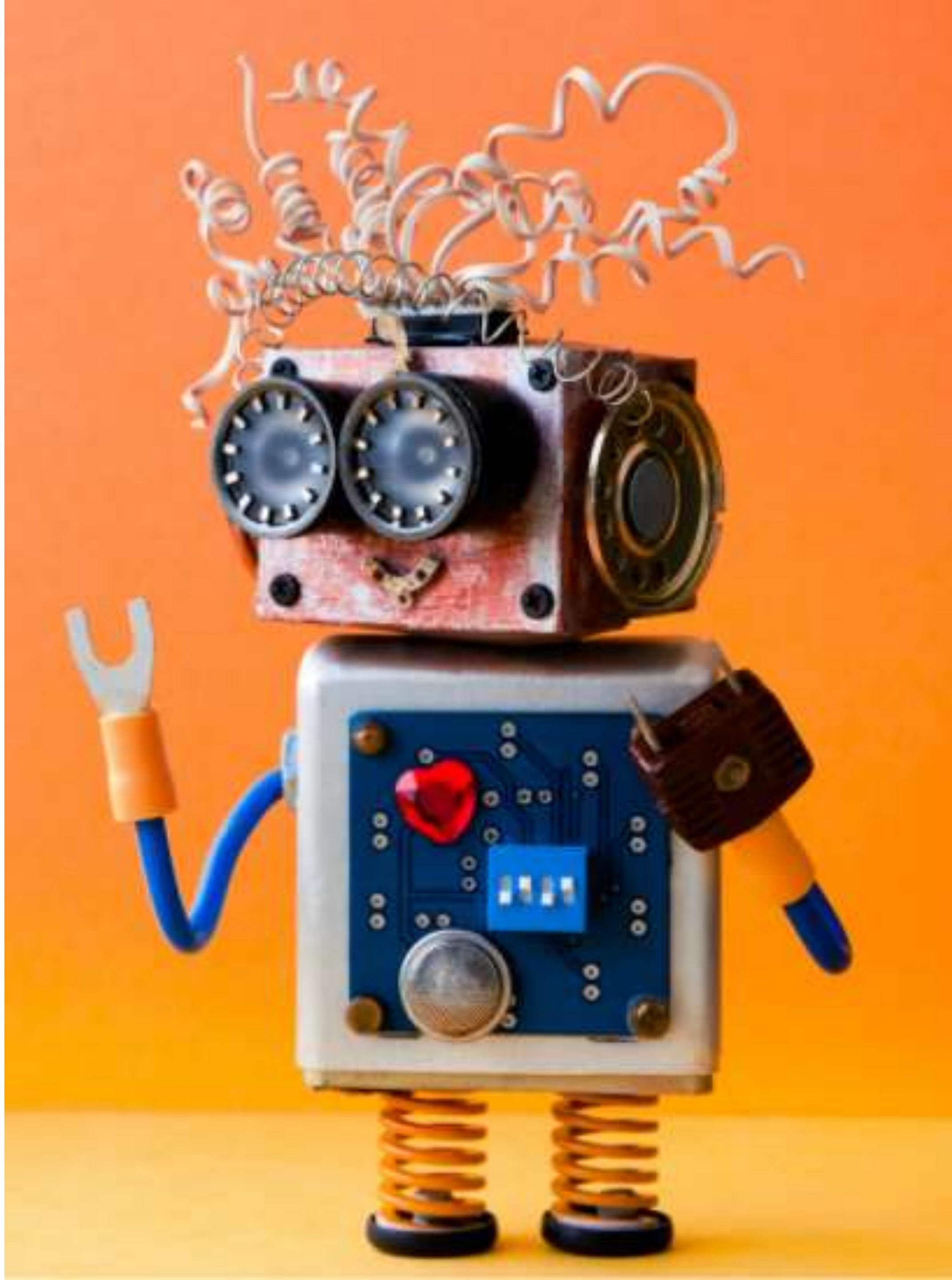
PROJECT OVERVIEW AND KEY LEARNING OUTCOMES



EASY

ADVANCED

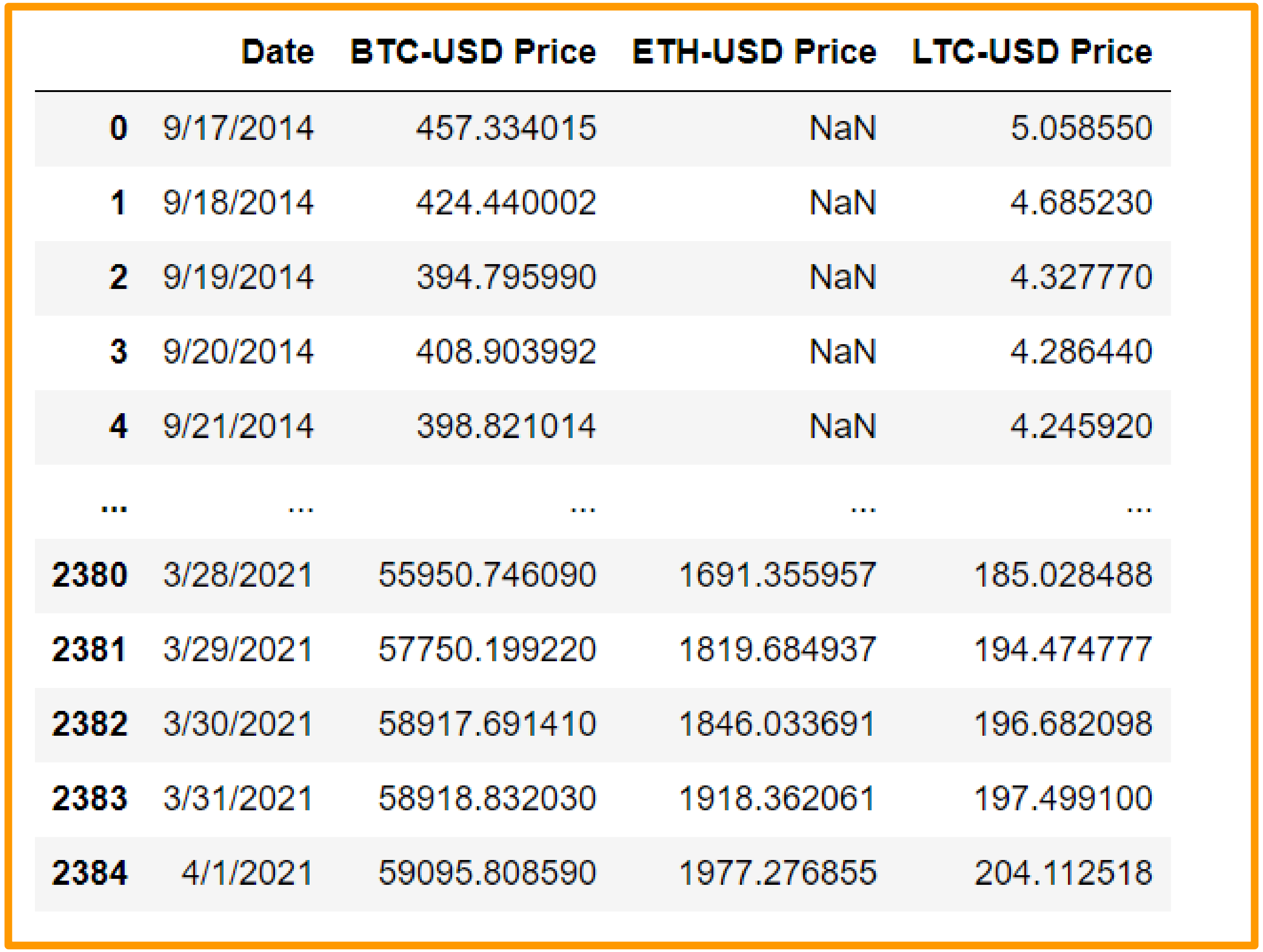


PROJECT OVERVIEW

- We will analyze cryptocurrency prices and daily returns such Bitcoin (BTC), Ethereum (ETH), Litecoin (LTC), Cardano (ADA) and Ripple (XRP) using Matplotlib and Seaborn libraries in AWS SageMaker Studio.
- Cryptocurrency is a decentralized digital currency that uses cryptography to secure transactions and do not have a centralized issuing authority (Government or banks).
- We will also analyze cancer datasets in AWS SageMaker Studio.
- We will learn how to:
  1. Perform data visualization using Seaborn and Matplotlib libraries
  2. Plot single line plot
  3. Plot pie charts
  4. Plot multiple subplots
  5. Plot pairplot and countplot using Seaborn
  6. Plot correlations and heatmaps
  7. Plot distribution plot (distplot)
  8. Plot Histograms
  9. Plot Scatterplots

PROJECT OVERVIEW: DATASET #1

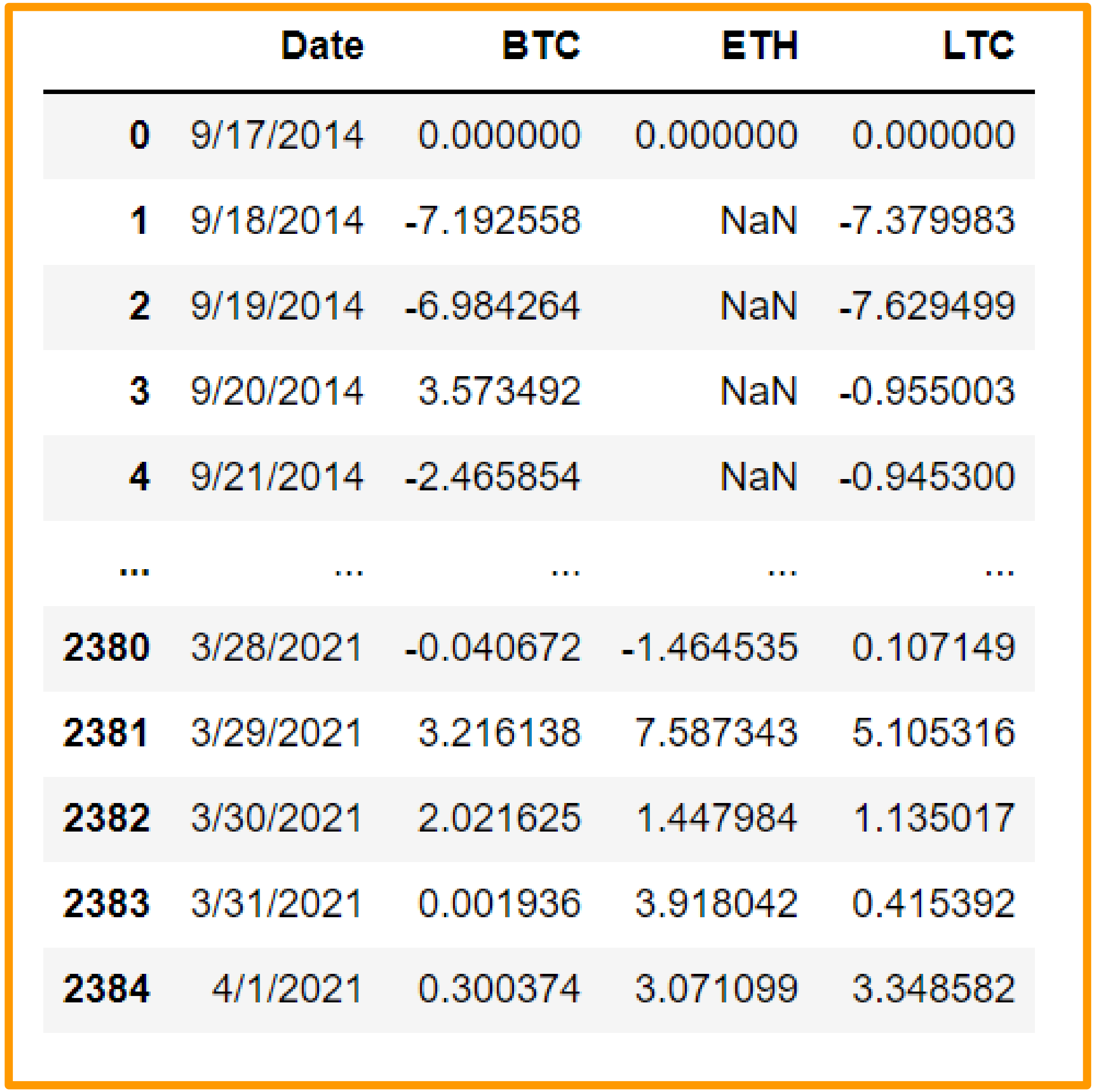
CRYPTOCURRENCY PRICES



	Date	BTC-USD Price	ETH-USD Price	LTC-USD Price
0	9/17/2014	457.334015	NaN	5.058550
1	9/18/2014	424.440002	NaN	4.685230
2	9/19/2014	394.795990	NaN	4.327770
3	9/20/2014	408.903992	NaN	4.286440
4	9/21/2014	398.821014	NaN	4.245920
...	...	...	...	...
2380	3/28/2021	55950.746090	1691.355957	185.028488
2381	3/29/2021	57750.199220	1819.684937	194.474777
2382	3/30/2021	58917.691410	1846.033691	196.682098
2383	3/31/2021	58918.832030	1918.362061	197.499100
2384	4/1/2021	59095.808590	1977.276855	204.112518

PROJECT OVERVIEW: DATASET #2

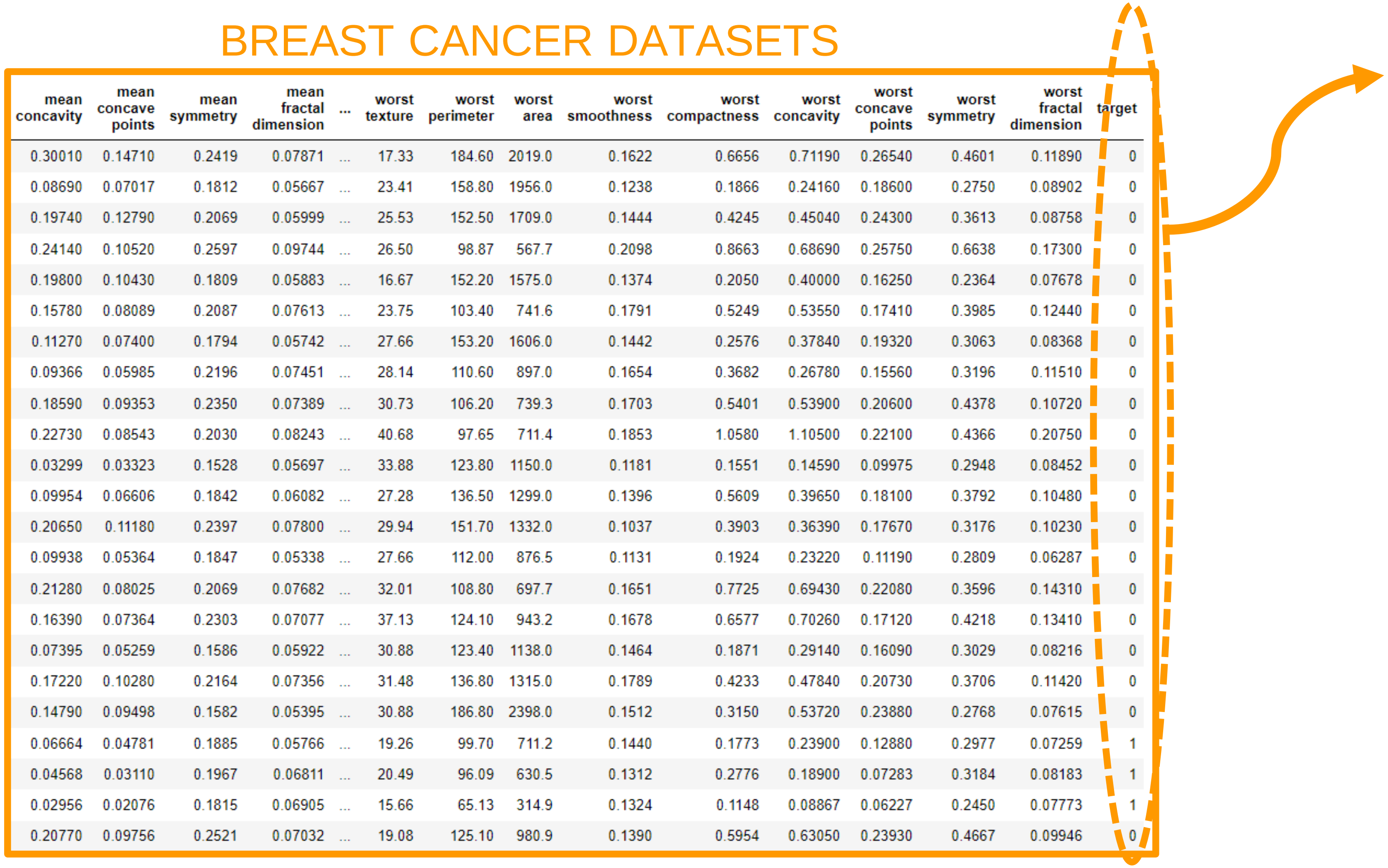
CRYPTOCURRENCY RETURNS



	Date	BTC	ETH	LTC
0	9/17/2014	0.000000	0.000000	0.000000
1	9/18/2014	-7.192558	NaN	-7.379983
2	9/19/2014	-6.984264	NaN	-7.629499
3	9/20/2014	3.573492	NaN	-0.955003
4	9/21/2014	-2.465854	NaN	-0.945300
...	...	...	...	...
2380	3/28/2021	-0.040672	-1.464535	0.107149
2381	3/29/2021	3.216138	7.587343	5.105316
2382	3/30/2021	2.021625	1.447984	1.135017
2383	3/31/2021	0.001936	3.918042	0.415392
2384	4/1/2021	0.300374	3.071099	3.348582

PROJECT OVERVIEW: DATASET #3

BREAST CANCER DATASETS



mean concavity	mean concave points	mean symmetry	mean fractal dimension	...	worst texture	worst perimeter	worst area	worst smoothness	worst compactness	worst concavity	worst concave points	worst symmetry	worst fractal dimension	target
0.30010	0.14710	0.2419	0.07871	...	17.33	184.60	2019.0	0.1622	0.6656	0.71190	0.26540	0.4601	0.11890	0
0.08690	0.07017	0.1812	0.05667	...	23.41	158.80	1956.0	0.1238	0.1866	0.24160	0.18600	0.2750	0.08902	0
0.19740	0.12790	0.2069	0.05999	...	25.53	152.50	1709.0	0.1444	0.4245	0.45040	0.24300	0.3613	0.08758	0
0.24140	0.10520	0.2597	0.09744	...	26.50	98.87	567.7	0.2098	0.8663	0.68690	0.25750	0.6638	0.17300	0
0.19800	0.10430	0.1809	0.05883	...	16.67	152.20	1575.0	0.1374	0.2050	0.40000	0.16250	0.2364	0.07678	0
0.15780	0.08089	0.2087	0.07613	...	23.75	103.40	741.6	0.1791	0.5249	0.53550	0.17410	0.3985	0.12440	0
0.11270	0.07400	0.1794	0.05742	...	27.66	153.20	1606.0	0.1442	0.2576	0.37840	0.19320	0.3063	0.08368	0
0.09366	0.05985	0.2196	0.07451	...	28.14	110.60	897.0	0.1654	0.3682	0.26780	0.15560	0.3196	0.11510	0
0.18590	0.09353	0.2350	0.07389	...	30.73	106.20	739.3	0.1703	0.5401	0.53900	0.20600	0.4378	0.10720	0
0.22730	0.08543	0.2030	0.08243	...	40.68	97.65	711.4	0.1853	1.0580	1.10500	0.22100	0.4366	0.20750	0
0.03299	0.03323	0.1528	0.05697	...	33.88	123.80	1150.0	0.1181	0.1551	0.14590	0.09975	0.2948	0.08452	0
0.09954	0.06606	0.1842	0.06082	...	27.28	136.50	1299.0	0.1396	0.5609	0.39650	0.18100	0.3792	0.10480	0
0.20650	0.11180	0.2397	0.07800	...	29.94	151.70	1332.0	0.1037	0.3903	0.36390	0.17670	0.3176	0.10230	0
0.09938	0.05364	0.1847	0.05338	...	27.66	112.00	876.5	0.1131	0.1924	0.23220	0.11190	0.2809	0.06287	0
0.21280	0.08025	0.2069	0.07682	...	32.01	108.80	697.7	0.1651	0.7725	0.69430	0.22080	0.3596	0.14310	0
0.16390	0.07364	0.2303	0.07077	...	37.13	124.10	943.2	0.1678	0.6577	0.70260	0.17120	0.4218	0.13410	0
0.07395	0.05259	0.1586	0.05922	...	30.88	123.40	1138.0	0.1464	0.1871	0.29140	0.16090	0.3029	0.08216	0
0.17220	0.10280	0.2164	0.07356	...	31.48	136.80	1315.0	0.1789	0.4233	0.47840	0.20730	0.3706	0.11420	0
0.14790	0.09498	0.1582	0.05395	...	30.88	186.80	2398.0	0.1512	0.3150	0.53720	0.23880	0.2768	0.07615	0
0.06664	0.04781	0.1885	0.05766	...	19.26	99.70	711.2	0.1440	0.1773	0.23900	0.12880	0.2977	0.07259	1
0.04568	0.03110	0.1967	0.06811	...	20.49	96.09	630.5	0.1312	0.2776	0.18900	0.07283	0.3184	0.08183	1
0.02956	0.02076	0.1815	0.06905	...	15.66	65.13	314.9	0.1324	0.1148	0.08867	0.06227	0.2450	0.07773	1
0.20770	0.09756	0.2521	0.07032	...	19.08	125.10	980.9	0.1390	0.5954	0.63050	0.23930	0.4667	0.09946	0

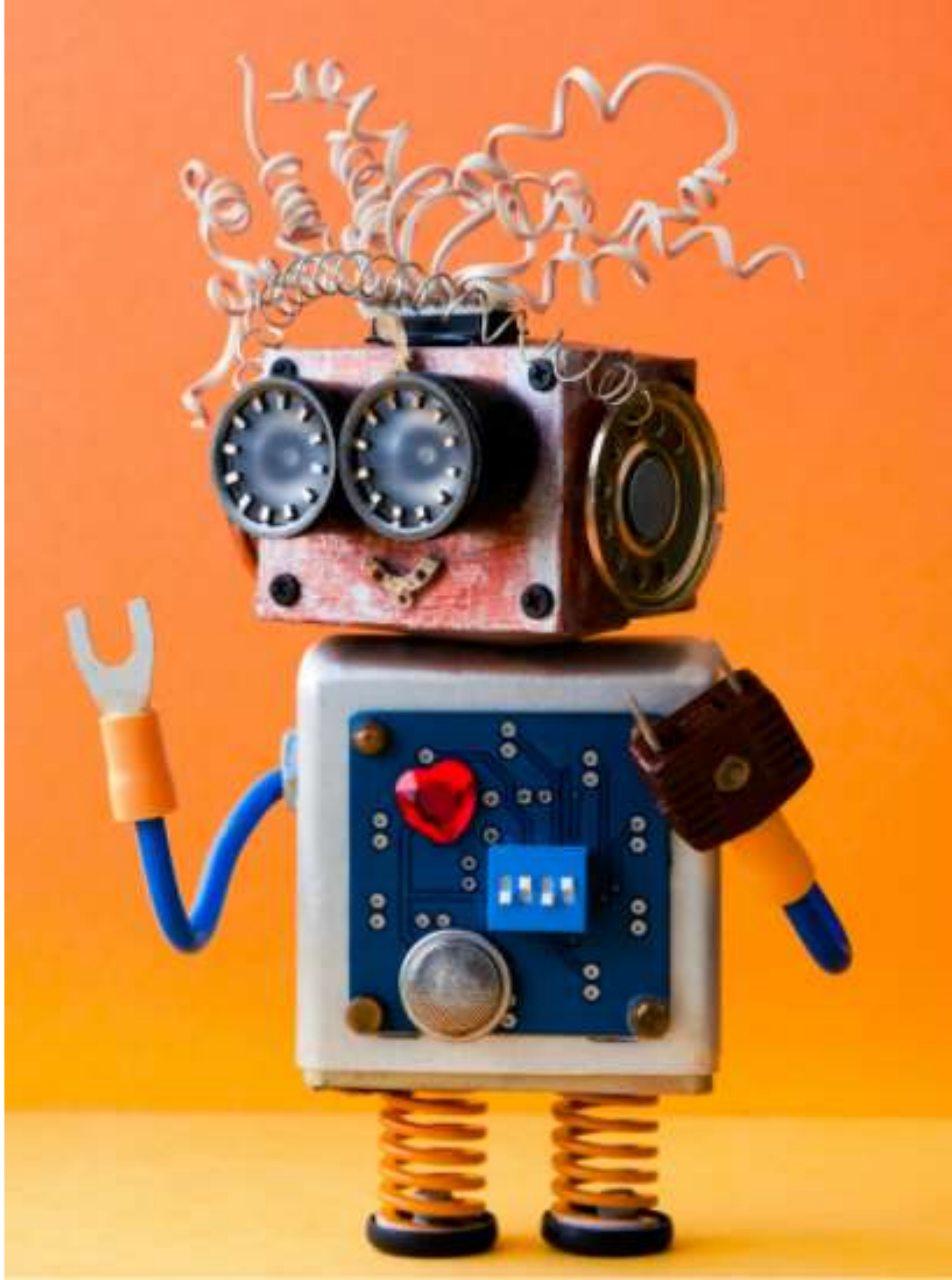
TARGET CLASS
MALIGNANT OR
BENIGN

Data Source: https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)

DATA VISUALIZATION 101

EASY

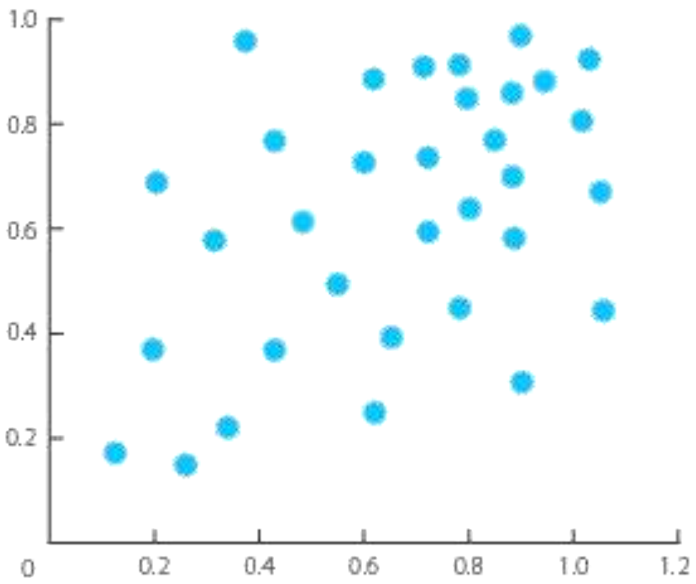
ADVANCED



RELATIONSHIPS

SCATTERPLOT

“Scatterplot demonstrates the relationship between two variables (X, Y)”



BUBBLE CHART

“Bubble chart demonstrates the relationship between three variables (X, Y, Bubble Size)

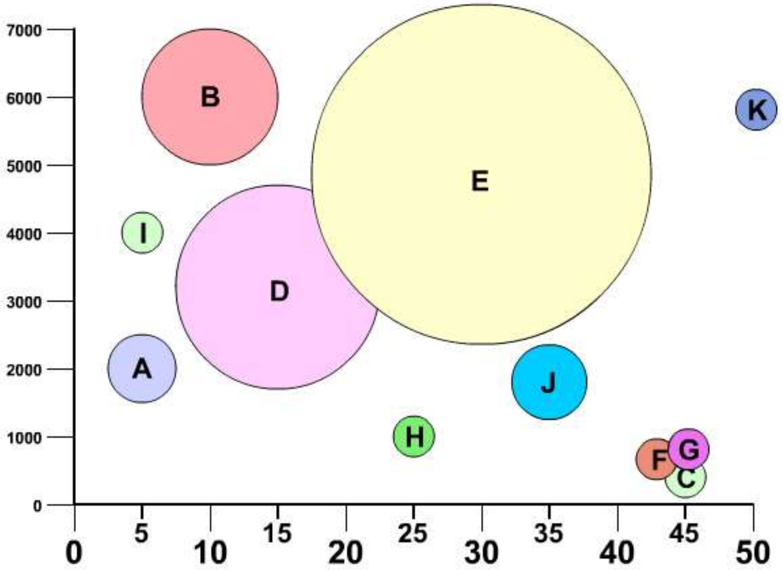
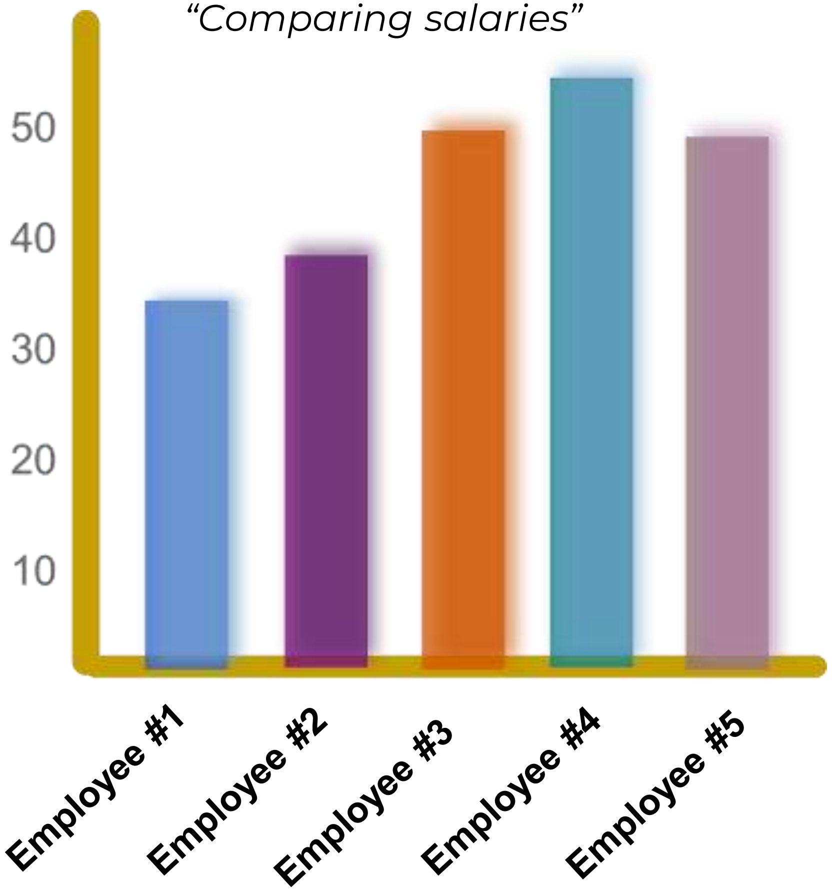


Photo Credit: https://commons.wikimedia.org/wiki/File:Example_of_Scatter_Plot.jpg
Photo Credit: https://commons.wikimedia.org/wiki/File:Bubble_chart.jpg

COMPARISONS

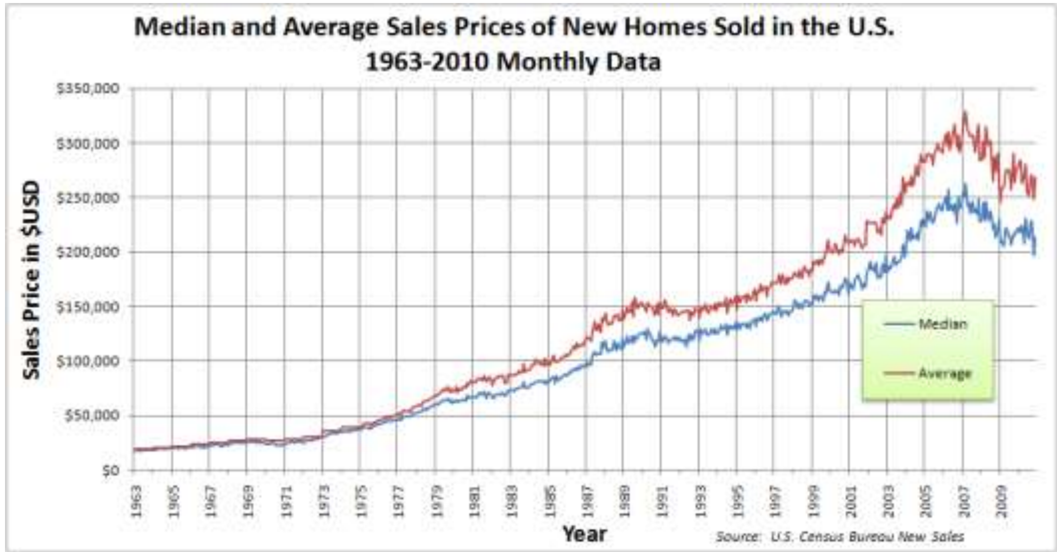
BAR CHART

“Comparing salaries”



LINE CHART

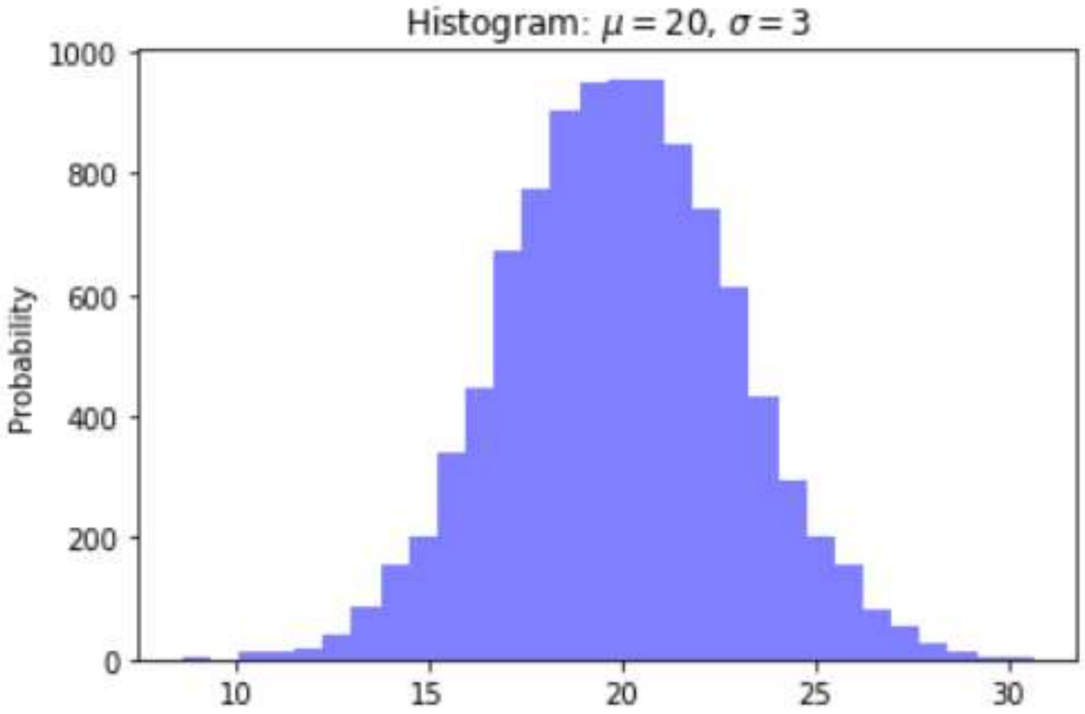
“Comparing median and average House prices over the years”



https://www.needpix.com/photo/89660/productivity-statistics-bar-chart-chart-graph-diagram-results
https://commons.wikimedia.org/wiki/File:Median_and_Average_Sales_Prices_of_New_Homes_Sold_in_the_US_1963-2010_Monthly.png

DISTRIBUTIONS

HISTOGRAMS



BOX PLOT

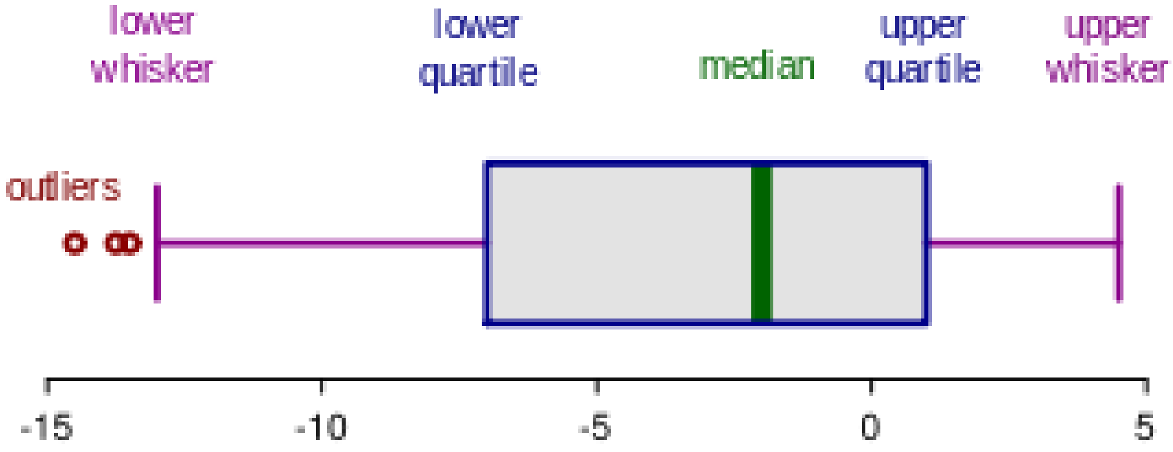
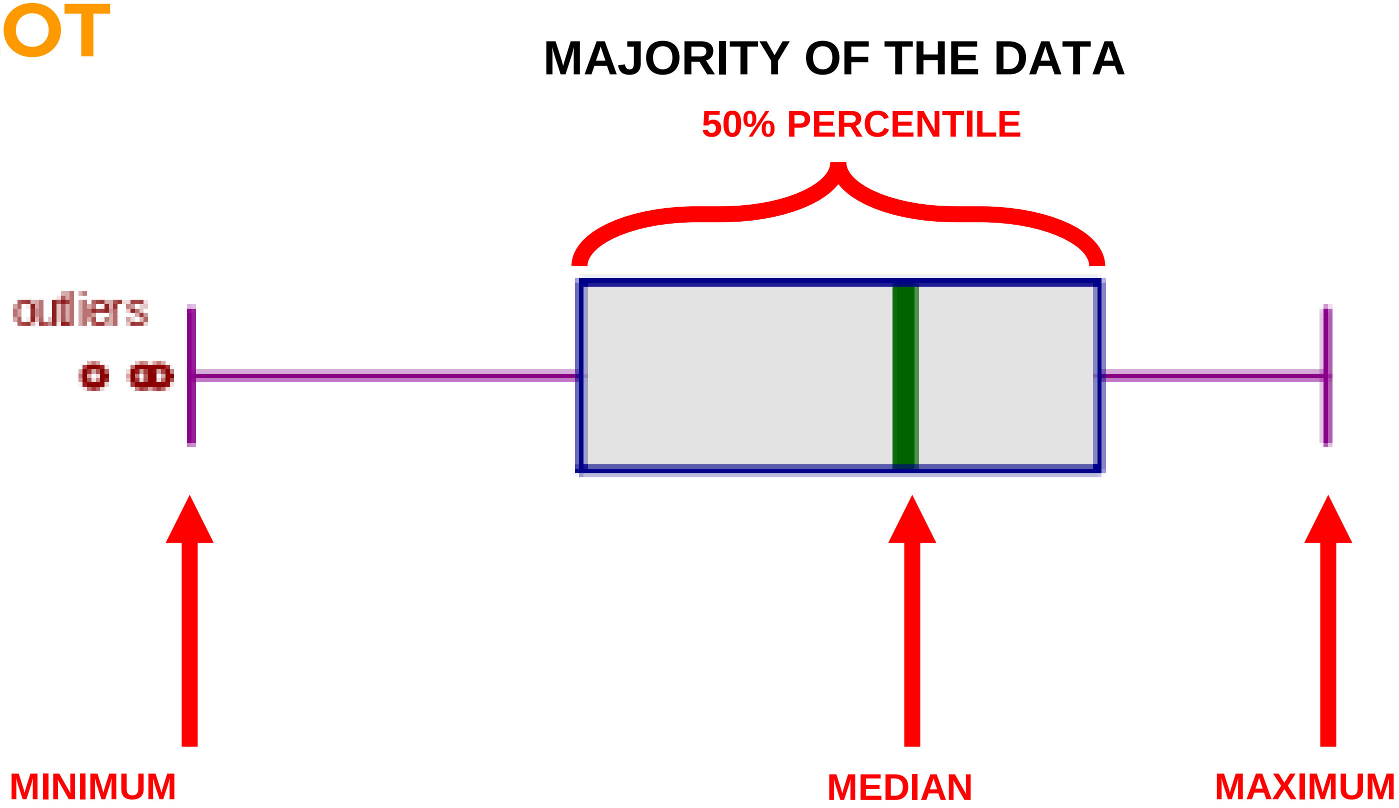


Photo Credit: https://commons.wikimedia.org/wiki/File:Elements_of_a_boxplot_en.svg

BOX PLOT



MAJORITY OF THE DATA

50% PERCENTILE

MINIMUM

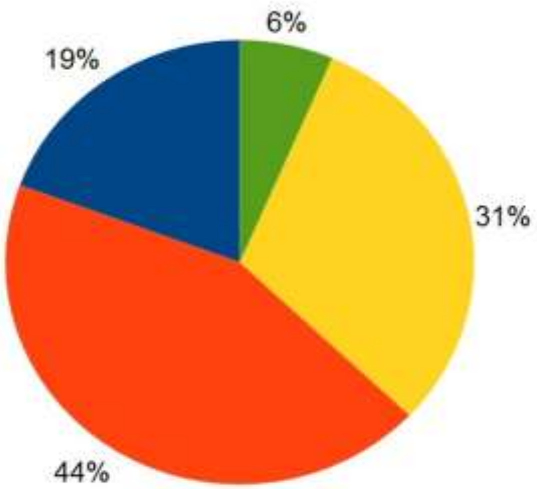
MEDIAN

MAXIMUM

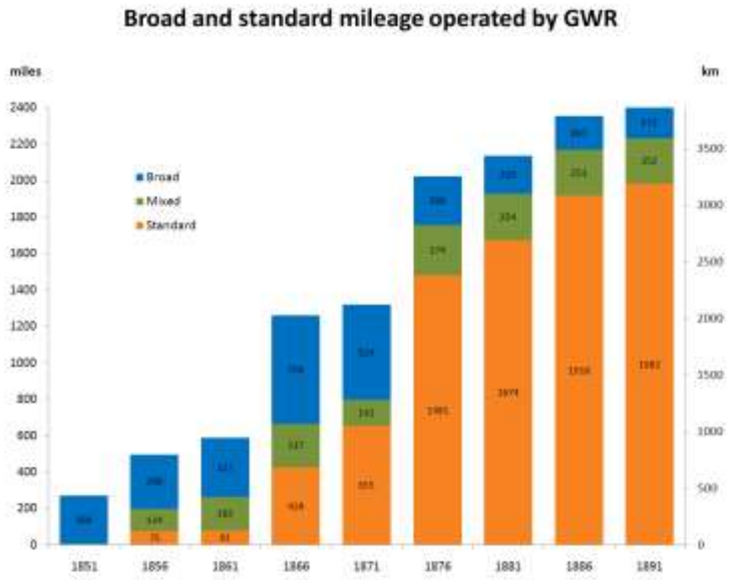
Photo Credit: https://commons.wikimedia.org/wiki/File:Elements_of_a_boxplot_en.svg

COMPOSITIONS

PIE CHART



STACKED BAR CHART



STACKED AREA CHART

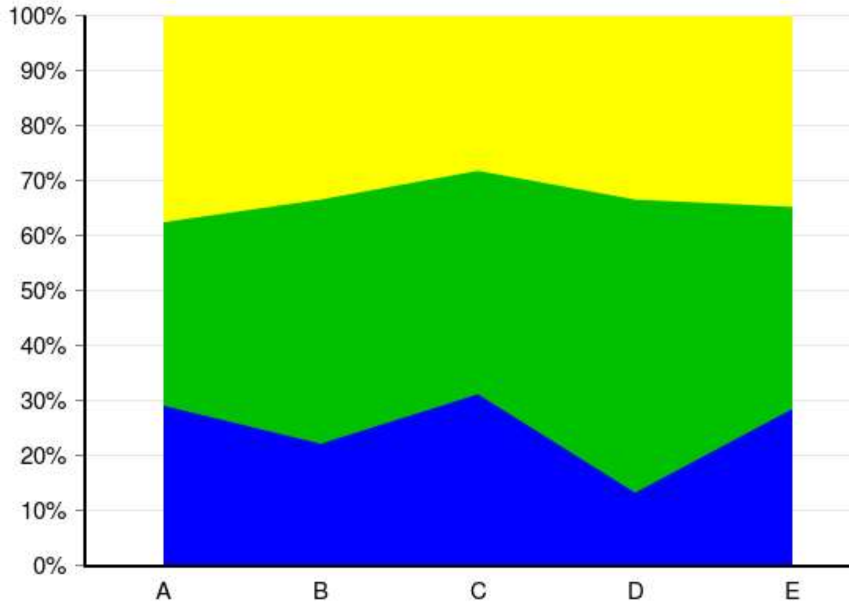
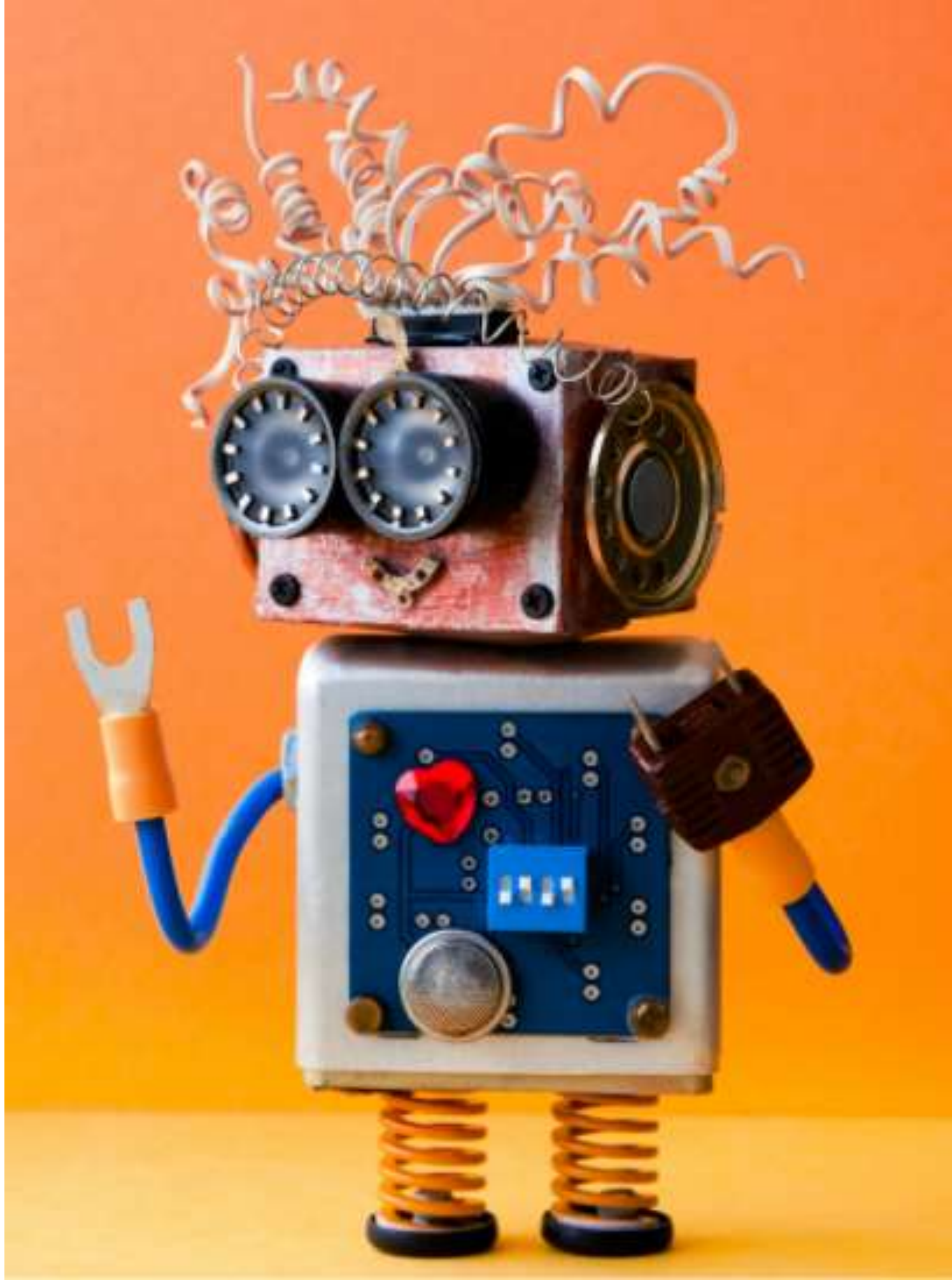


Photo Credit: https://commons.wikimedia.org/wiki/File:Broad_and_standard_mileage_operated_by_GWR.png
Photo Credit: https://commons.wikimedia.org/wiki/File:Charts_SVG_Example_12_-_Stacked_100%25_Area_Chart.svg
Photo Credit: https://commons.wikimedia.org/wiki/File:Pie-chart.jpg

MATPLOTLIB 101

EASY ▲ ADVANCED



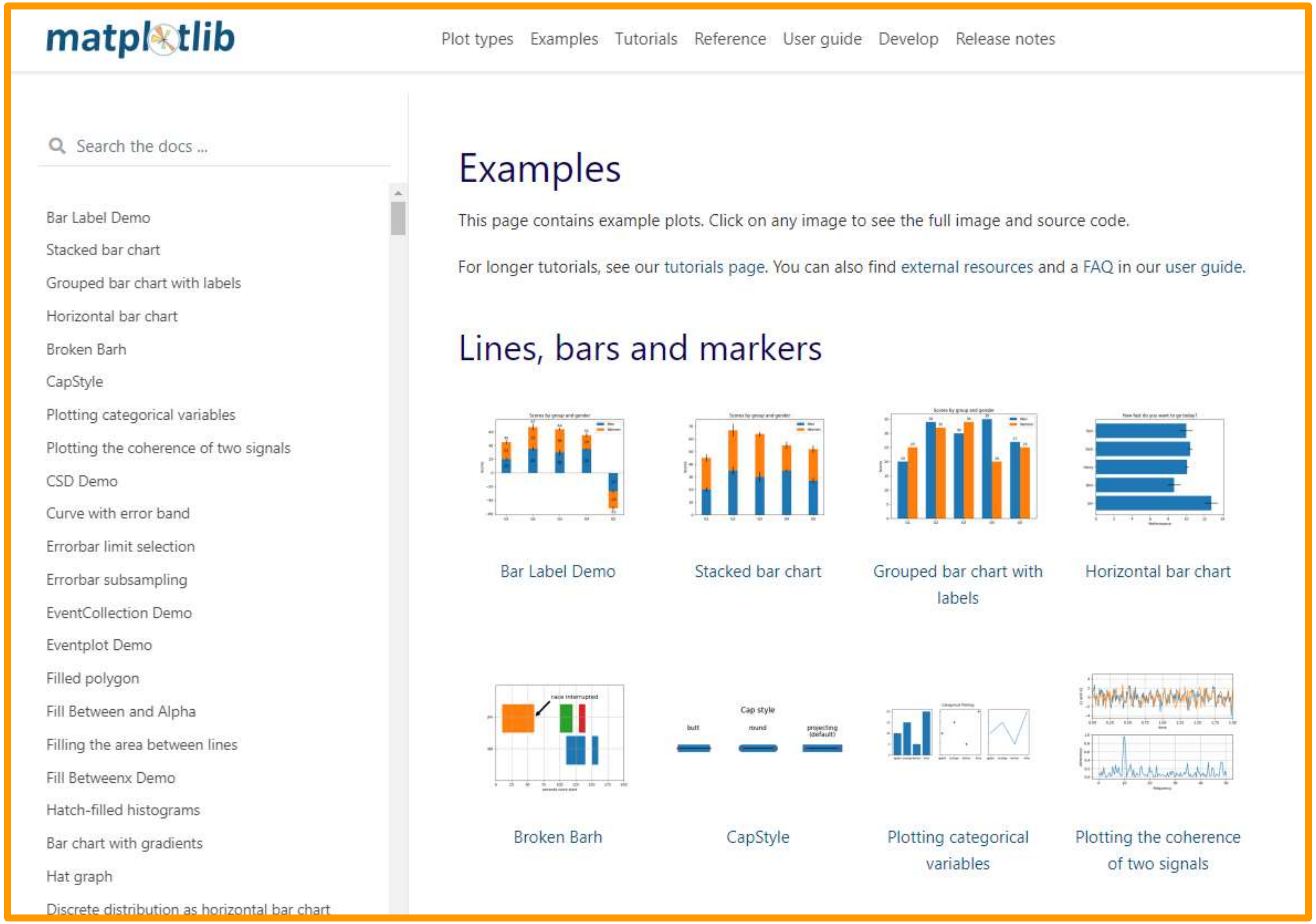
MATPLOTLIB

- Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python.
- Matplotlib is the godfather of data visualization libraries!
- Matplotlib was originally written by John D. Hunter. After John’s death, Michael Droettboom was nominated as matplotlib's lead developer in 2012.
- Matplotlib can be used to create (1) publication quality plots, (2) Customize figure style, (3) Embed in JupyterLabs and Graphical User Interfaces.
- Matplotlib works great with Pandas dataFrames. The plot method on Pandas Series and DataFrames is just a simple wrapper around plt.plot():

- Link to Library: https://matplotlib.org/

MATPLOTLIB GALLERY

Check this out: https://matplotlib.org/stable/gallery/index



Bar Label Demo
Stacked bar chart
Grouped bar chart with labels
Horizontal bar chart
Broken Barh
CapStyle
Plotting categorical variables
Plotting the coherence of two signals
CSD Demo
Curve with error band
Errorbar limit selection
Errorbar subsampling
EventCollection Demo
Eventplot Demo
Filled polygon
Fill Between and Alpha
Filling the area between lines
Fill Betweenx Demo
Hatch-filled histograms
Bar chart with gradients
Hat graph
Discrete distribution as horizontal bar chart

Examples

This page contains example plots. Click on any image to see the full image and source code.

For longer tutorials, see our tutorials page. You can also find external resources and a FAQ in our user guide.

Lines, bars and markers

Bar Label Demo

Stacked bar chart

Grouped bar chart with labels

Horizontal bar chart

Broken Barh

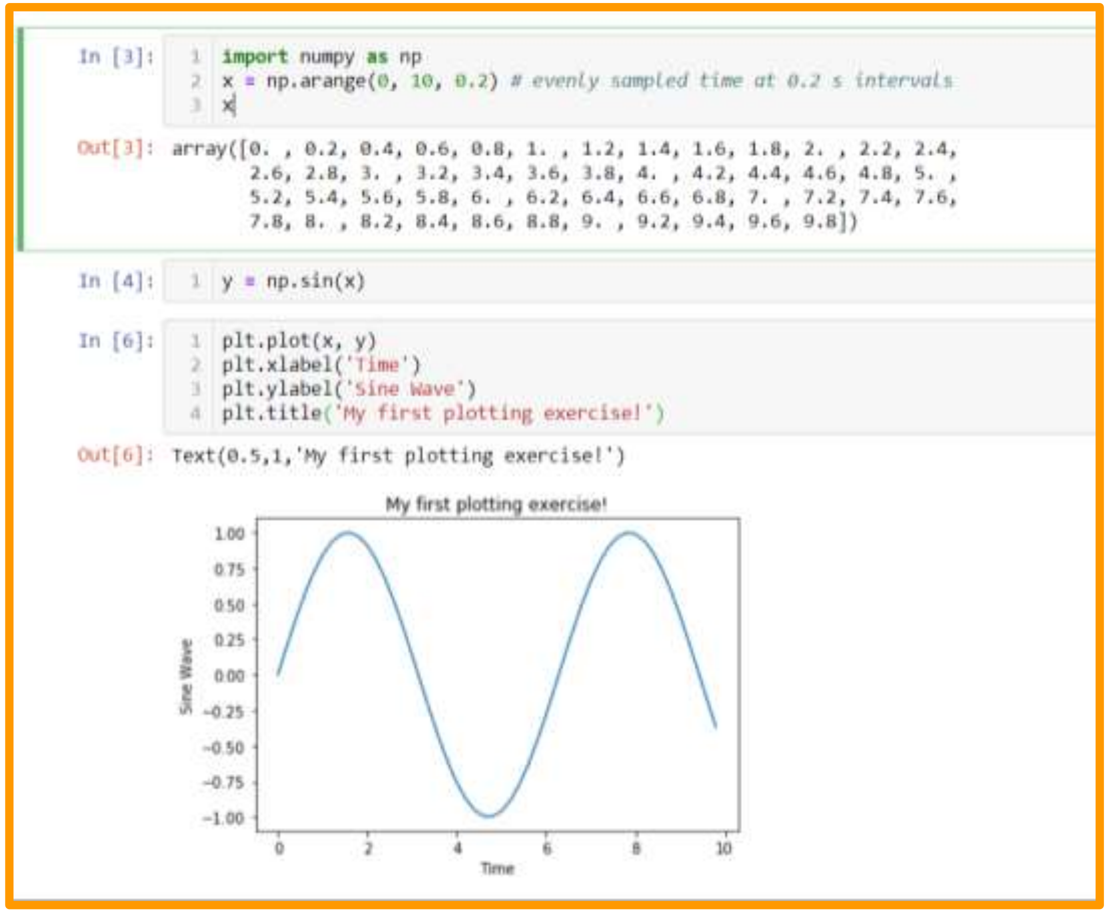
CapStyle

Plotting categorical variables

Plotting the coherence of two signals

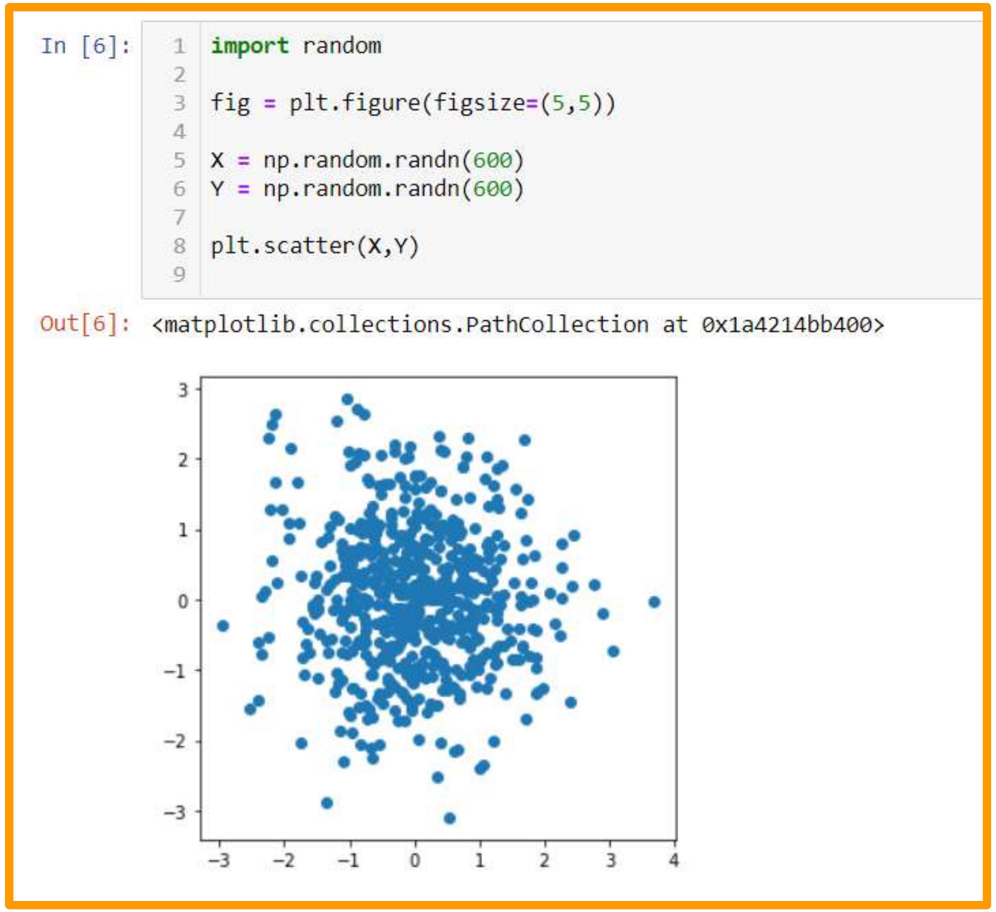
MATPLOTLIB SAMPLE CODE

LINE PLOT



```
In [3]:  import numpy as np
         x = np.arange(0, 10, 0.2) # evenly sampled time at 0.2 s intervals
         x

Out[3]: array([0. , 0.2, 0.4, 0.6, 0.8, 1. , 1.2, 1.4, 1.6, 1.8, 2. , 2.2, 2.4,
               2.6, 2.8, 3. , 3.2, 3.4, 3.6, 3.8, 4. , 4.2, 4.4, 4.6, 4.8, 5. ,
               5.2, 5.4, 5.6, 5.8, 6. , 6.2, 6.4, 6.6, 6.8, 7. , 7.2, 7.4, 7.6,
               7.8, 8. , 8.2, 8.4, 8.6, 8.8, 9. , 9.2, 9.4, 9.6, 9.8])

In [4]:  y = np.sin(x)

In [6]:  plt.plot(x, y)
         plt.xlabel('Time')
         plt.ylabel('Sine Wave')
         plt.title('My first plotting exercise!')

Out[6]: Text(0.5,1,'My first plotting exercise!')
```

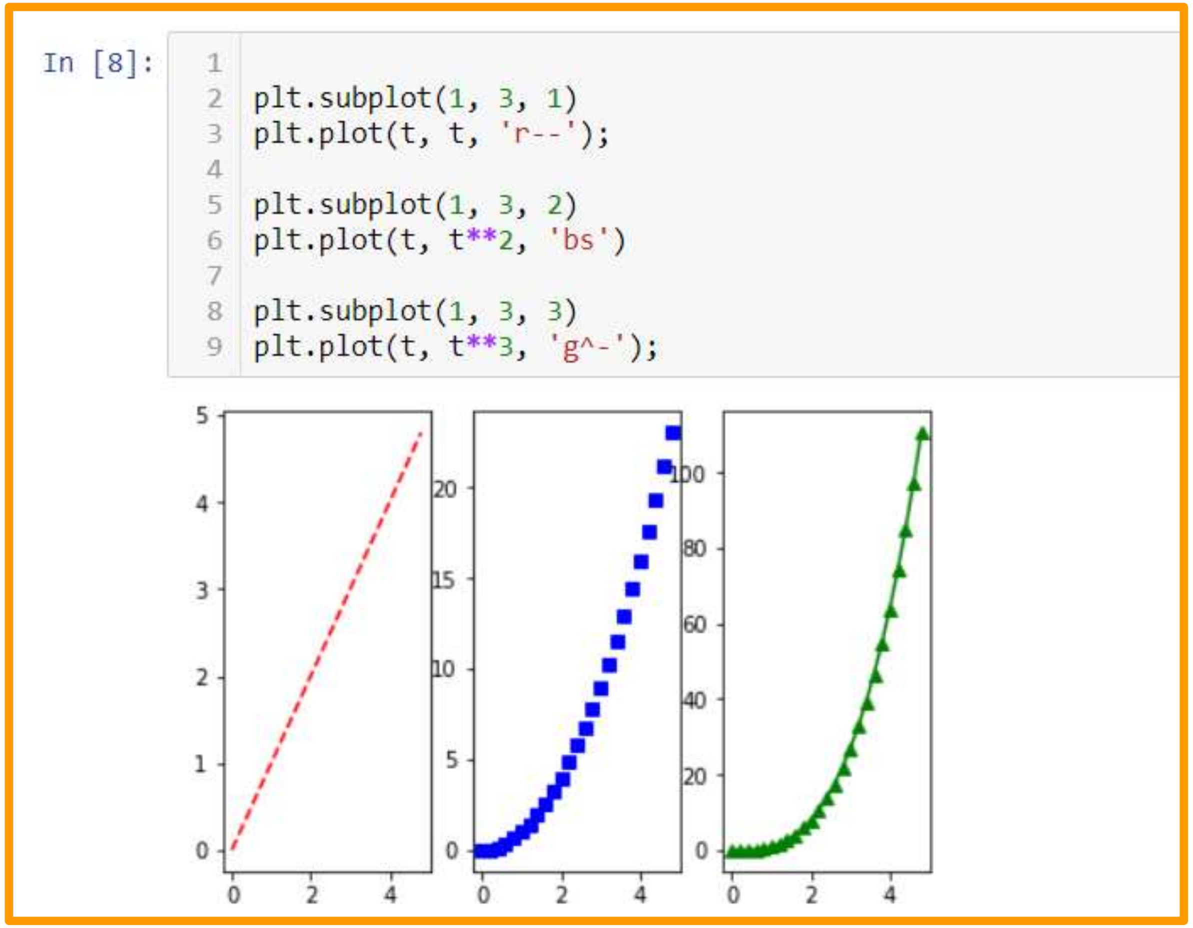
SCATTER PLOT



```
In [6]:  import random

         fig = plt.figure(figsize=(5,5))

         X = np.random.randn(600)
         Y = np.random.randn(600)

         plt.scatter(X,Y)

Out[6]: <matplotlib.collections.PathCollection at 0x1a4214bb400>
```

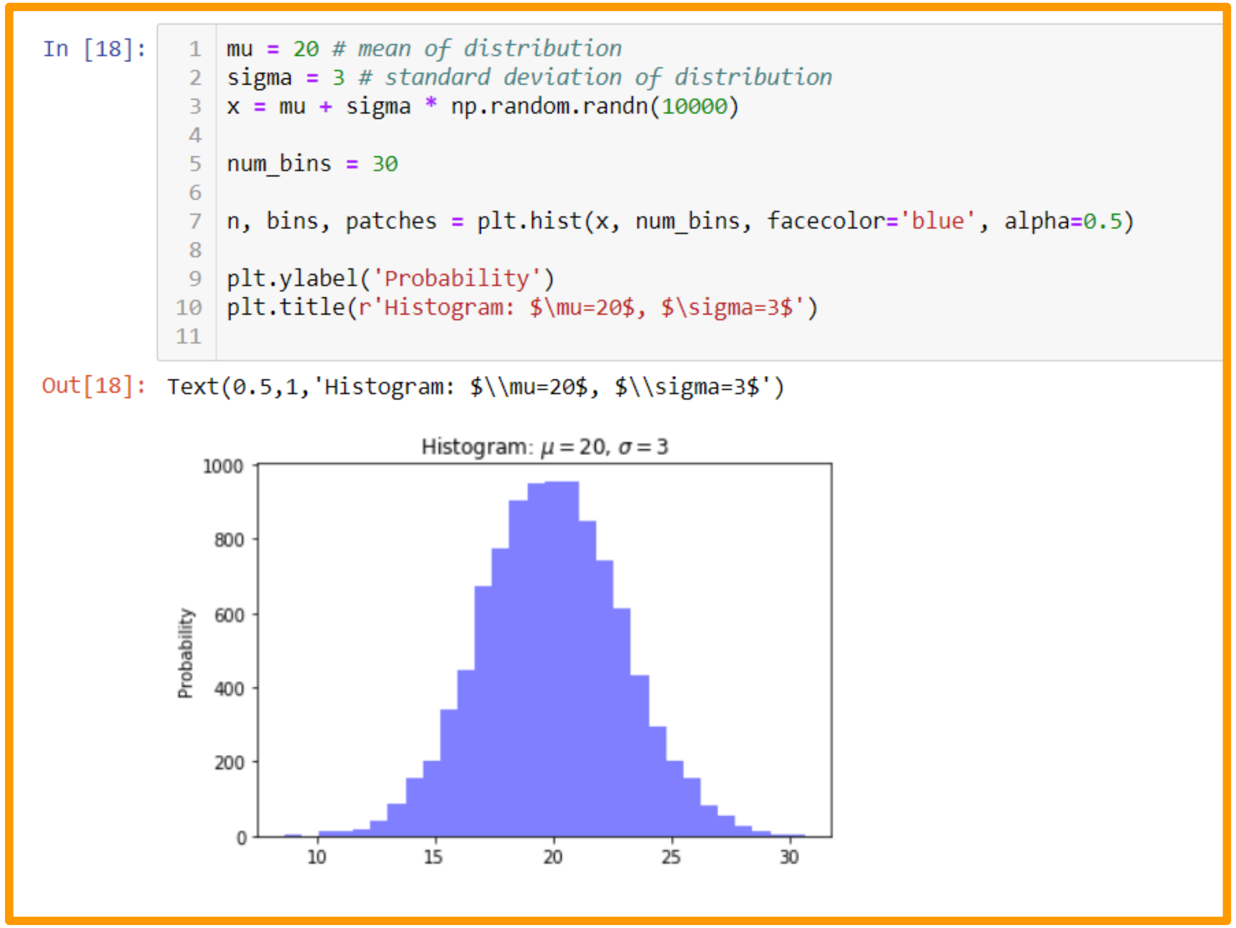
MATPLOTLIB SAMPLE CODE

SUBPLOT



```
In [8]:

plt.subplot(1, 3, 1)
plt.plot(t, t, 'r--');

plt.subplot(1, 3, 2)
plt.plot(t, t**2, 'bs')

plt.subplot(1, 3, 3)
plt.plot(t, t**3, 'g^-');
```

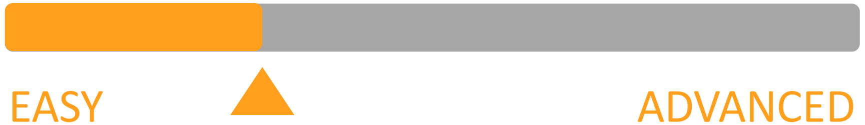
HISTOGRAMS



```
In [18]:
mu = 20 # mean of distribution
sigma = 3 # standard deviation of distribution
x = mu + sigma * np.random.randn(10000)

num_bins = 30

n, bins, patches = plt.hist(x, num_bins, facecolor='blue', alpha=0.5)

plt.ylabel('Probability')
plt.title(r'Histogram: $\mu=20$, $\sigma=3$')

```

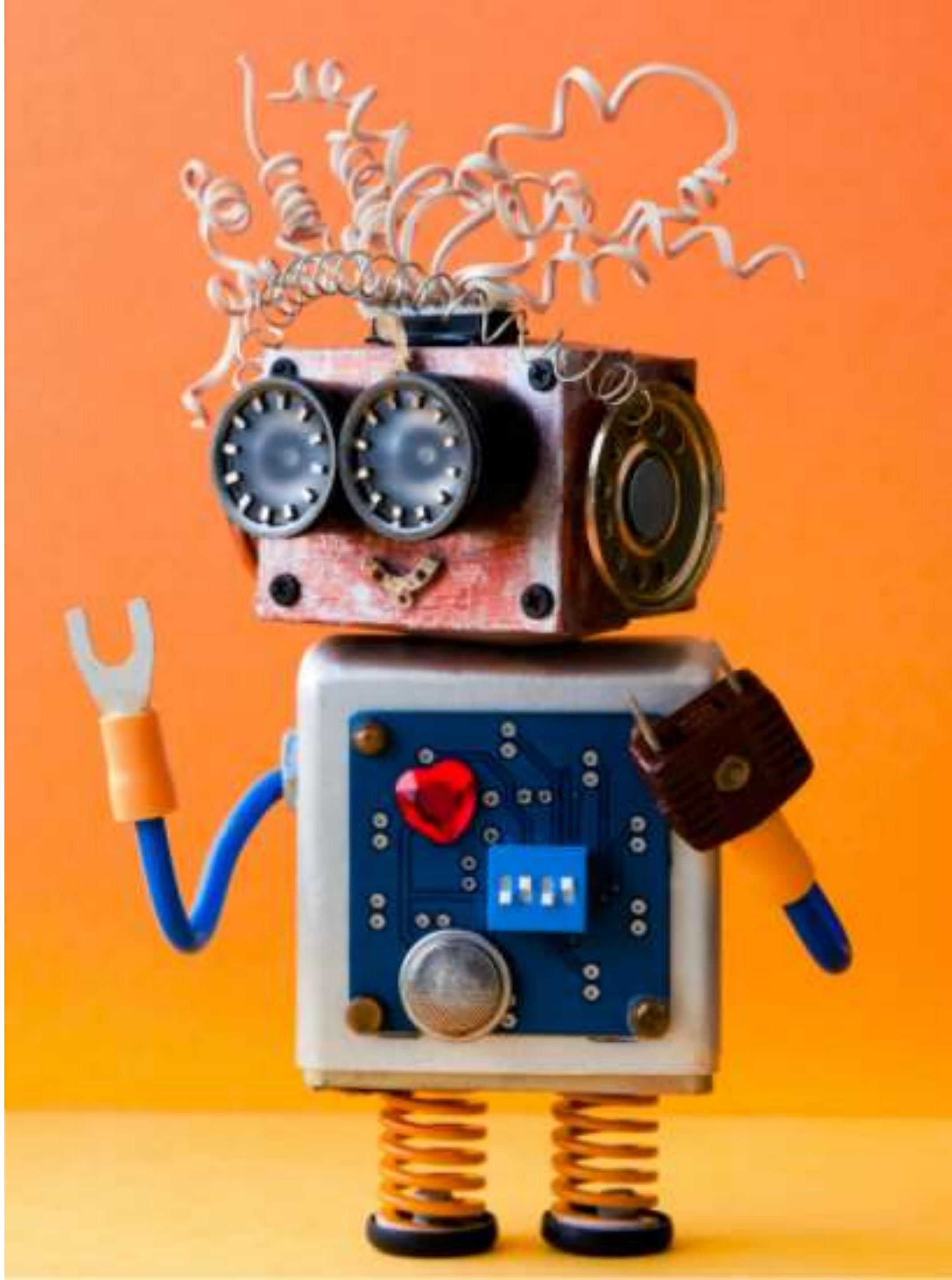
```
Out[18]: Text(0.5,1,'Histogram: $\\mu=20$, $\\sigma=3$')
```

SEABORN 101



EASY

ADVANCED

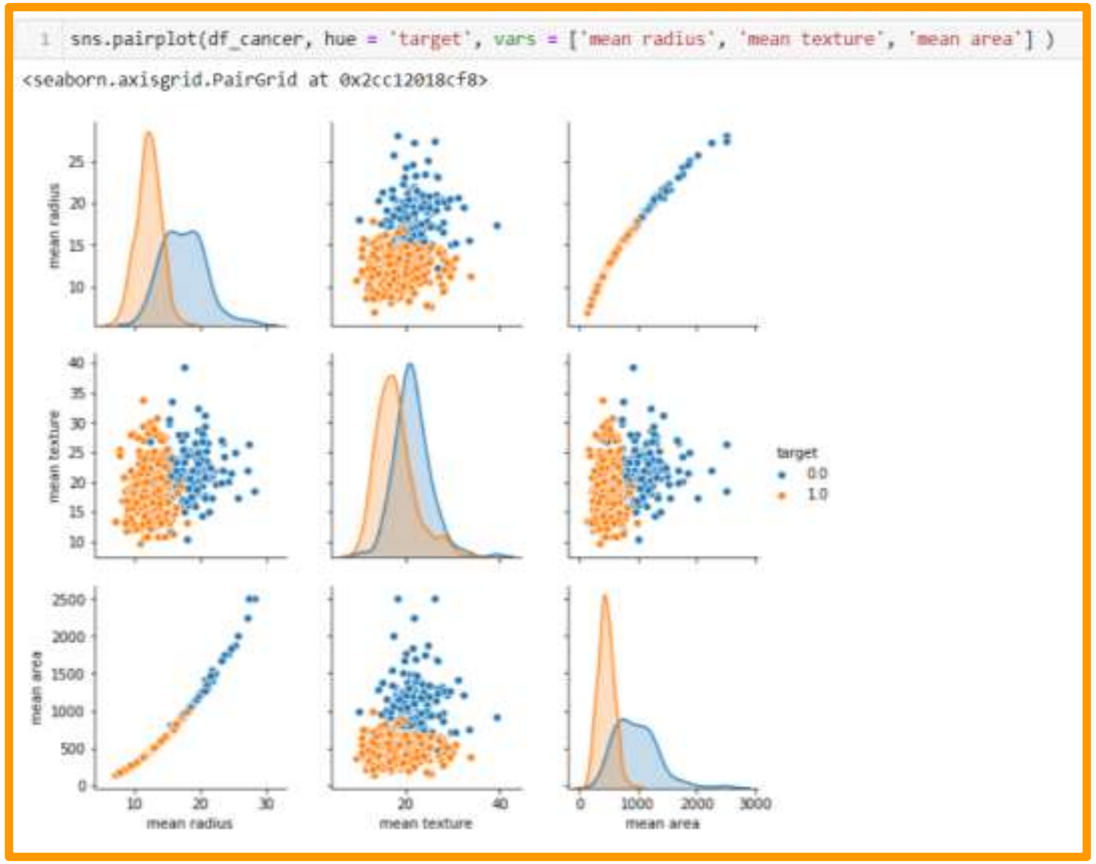


SEABORN

- Seaborn is a data visualization library that sits on top of matplotlib
- Seaborn offers enhanced features compared to matplotlib, it’s Matplotlib on steroids!
- Link to Seaborn: https://seaborn.pydata.org/examples/index.html

SEABORN EXAMPLES

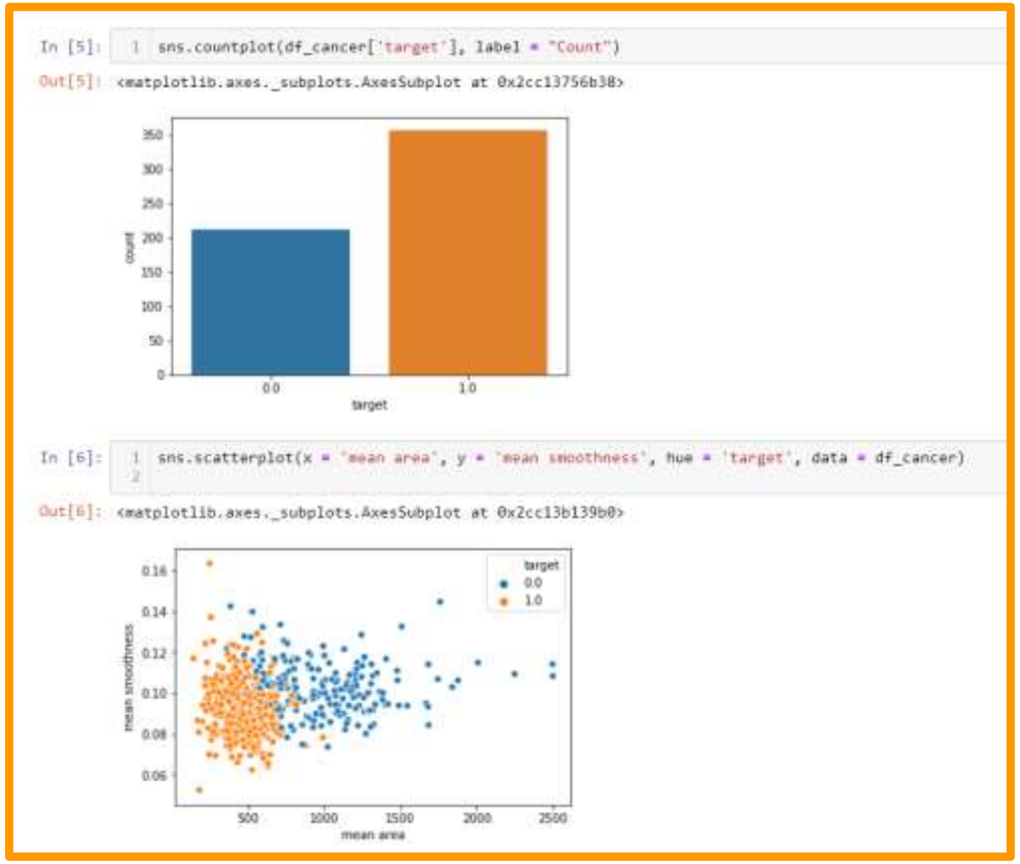
PAIRPLOT



```
sns.pairplot(df_cancer, hue = 'target', vars = ['mean radius', 'mean texture', 'mean area'] )
```

```
<seaborn.axisgrid.PairGrid at 0x2cc12018cf8>
```

COUNTPLOT AND SCATTERPLOT



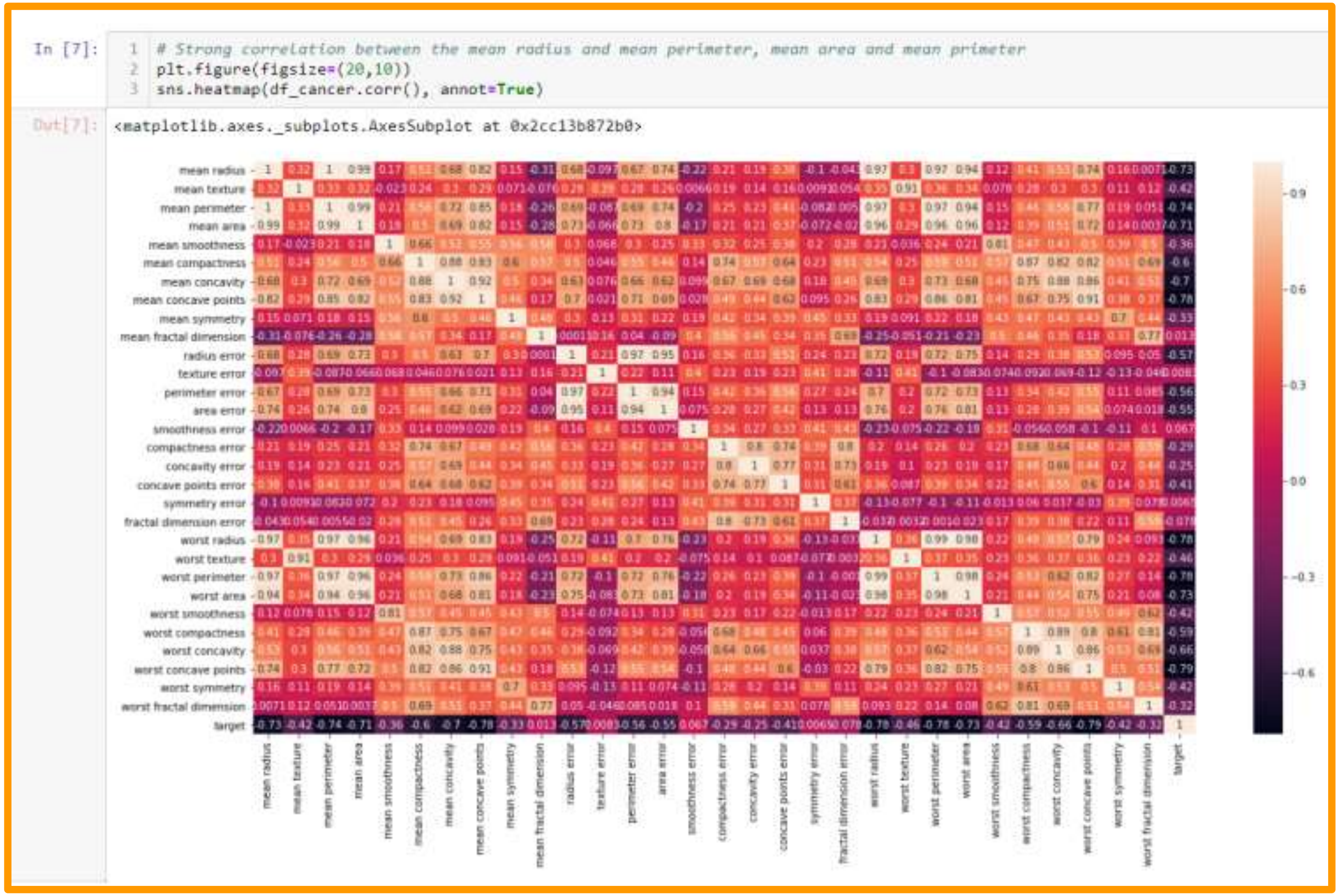
```
In [5]: sns.countplot(df_cancer['target'], label = "Count")
Out[5]: <matplotlib.axes._subplots.AxesSubplot at 0x2cc13756b38>
```

```
In [6]: sns.scatterplot(x = 'mean area', y = 'mean smoothness', hue = 'target', data = df_cancer)

Out[6]: <matplotlib.axes._subplots.AxesSubplot at 0x2cc13b139b0>
```

SEABORN EXAMPLES

- Heatmaps are used to represents values as colours.

CORRELATIONS AND HEATMAPS



In [7]:

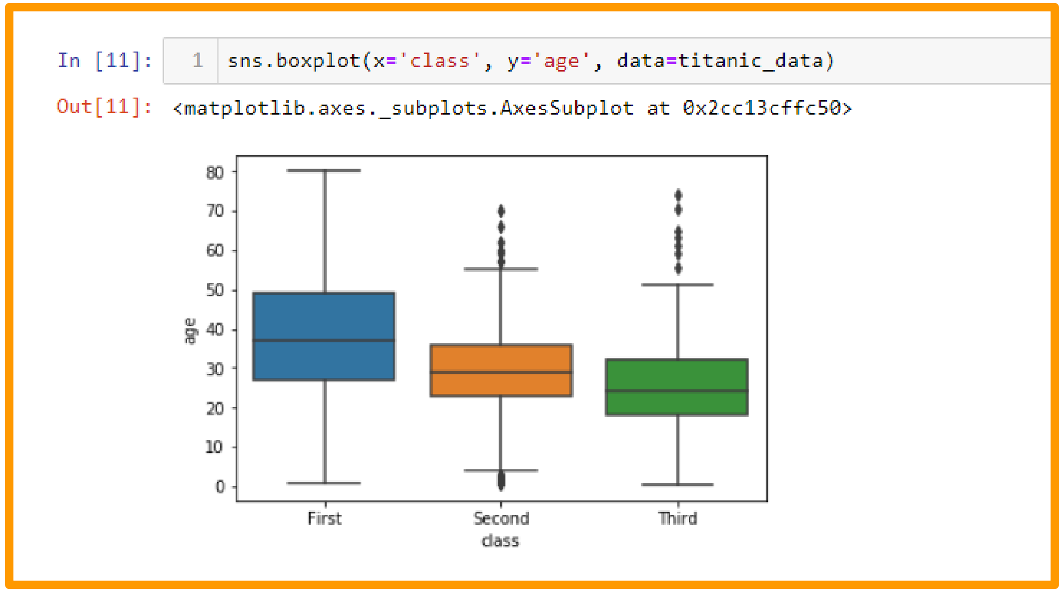
```
# Strong correlation between the mean radius and mean perimeter, mean area and mean primeter
plt.figure(figsize=(20,10))
sns.heatmap(df_cancer.corr(), annot=True)
```

Out[7]:

```
<matplotlib.axes._subplots.AxesSubplot at 0x2cc13b872b0>
```

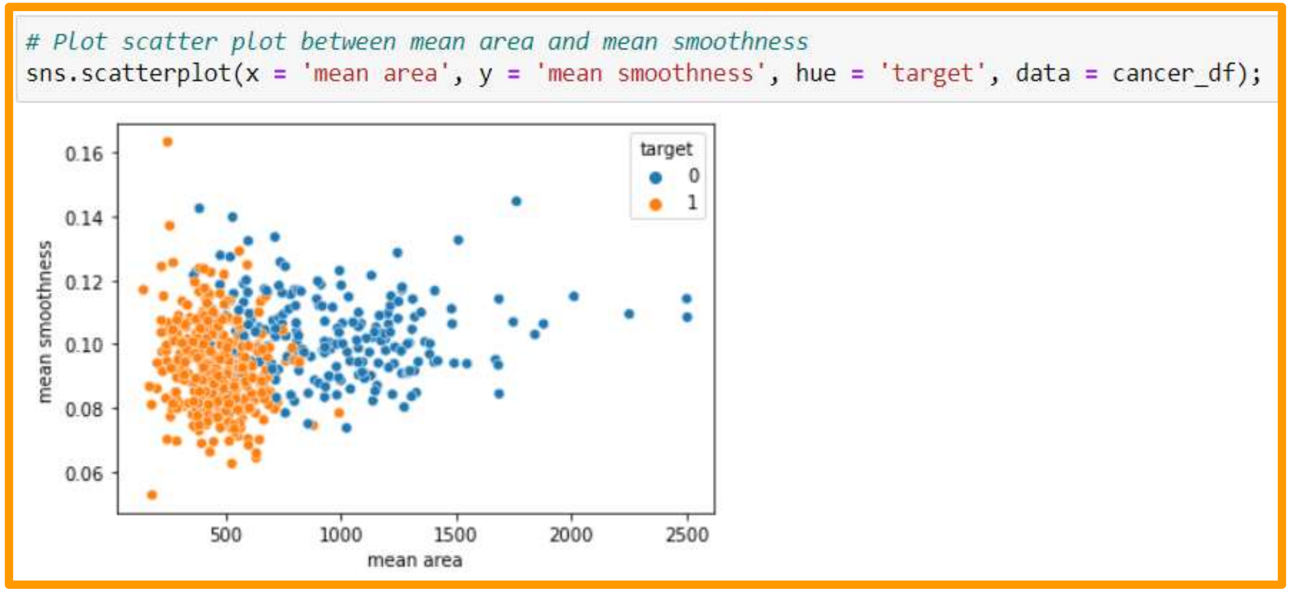
SEABORN EXAMPLES

BOXPLOT



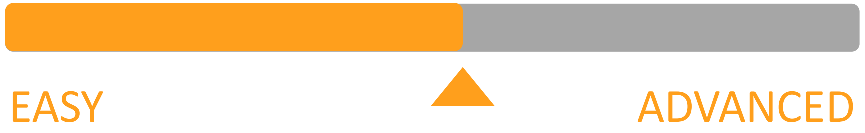
```
In [11]: sns.boxplot(x='class', y='age', data=titanic_data)
Out[11]: <matplotlib.axes._subplots.AxesSubplot at 0x2cc13cffc50>
```

SCATTERPLOT



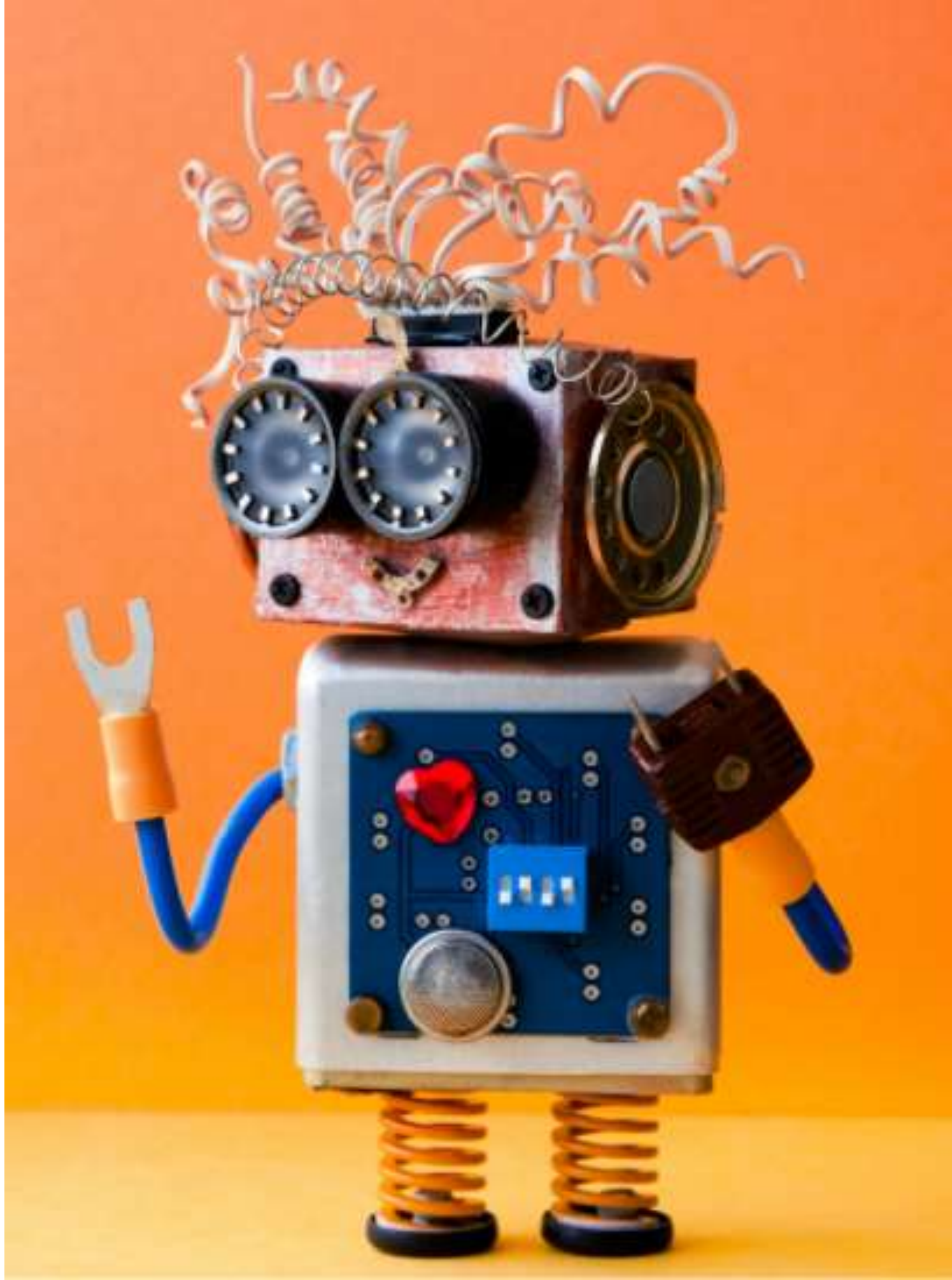
```
# Plot scatter plot between mean area and mean smoothness
sns.scatterplot(x = 'mean area', y = 'mean smoothness', hue = 'target', data = cancer_df);
```

PROJECT DEMO

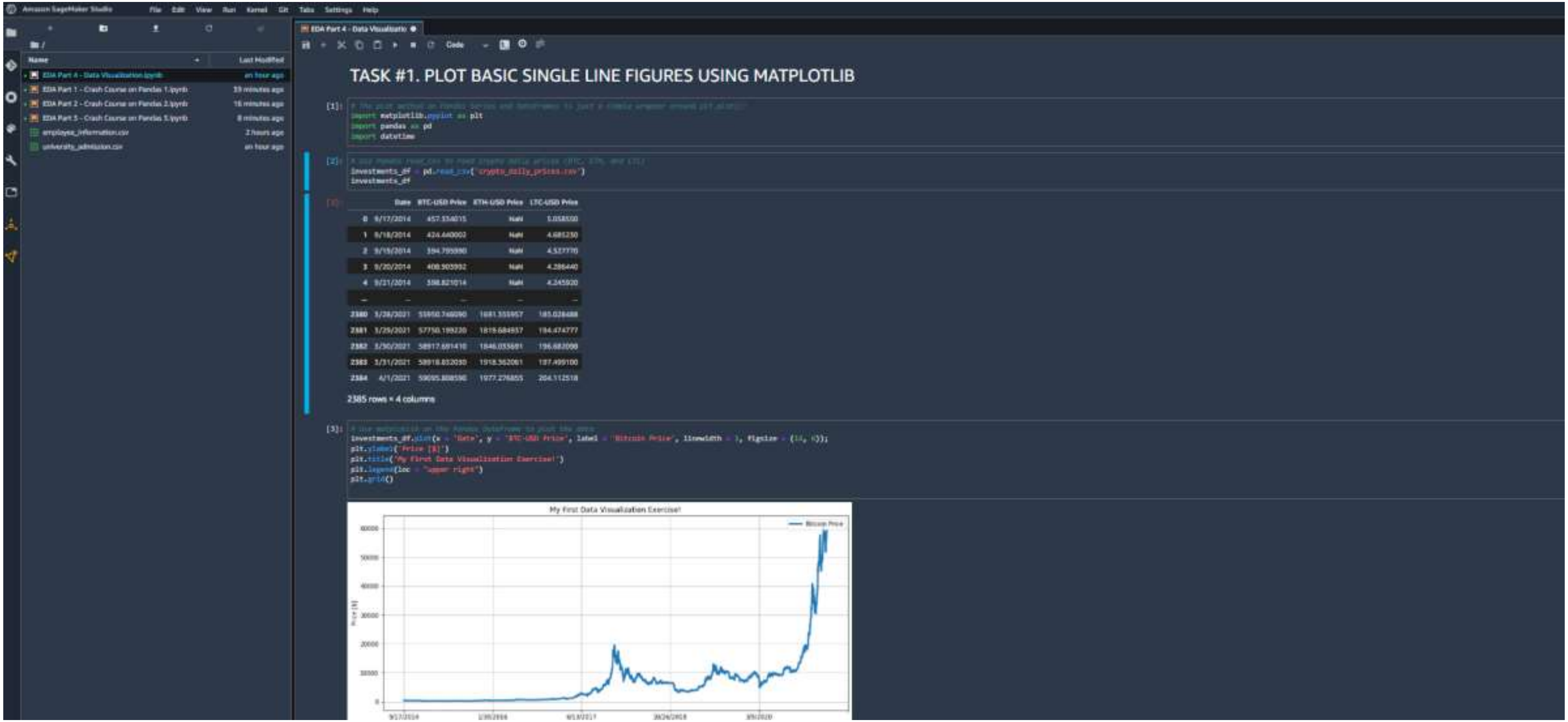


EASY

ADVANCED



PROJECT DEMO



TASK #1. PLOT BASIC SINGLE LINE FIGURES USING MATPLOTLIB

```
[1]: # The plot method on Pandas Series and DataFrames is just a simple wrapper around plt.plot()
     import matplotlib.pyplot as plt
     import pandas as pd
     import datetime
```

```
[2]: # Use Pandas read_csv to read crypto daily prices (BTC, ETH, and LTC)
     investments_df = pd.read_csv('crypto_daily_prices.csv')
     investments_df
```

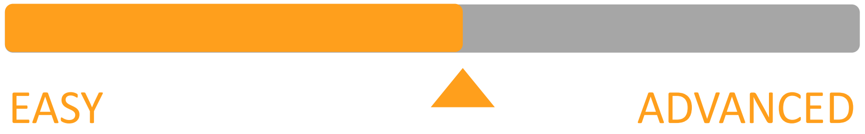
[2]:

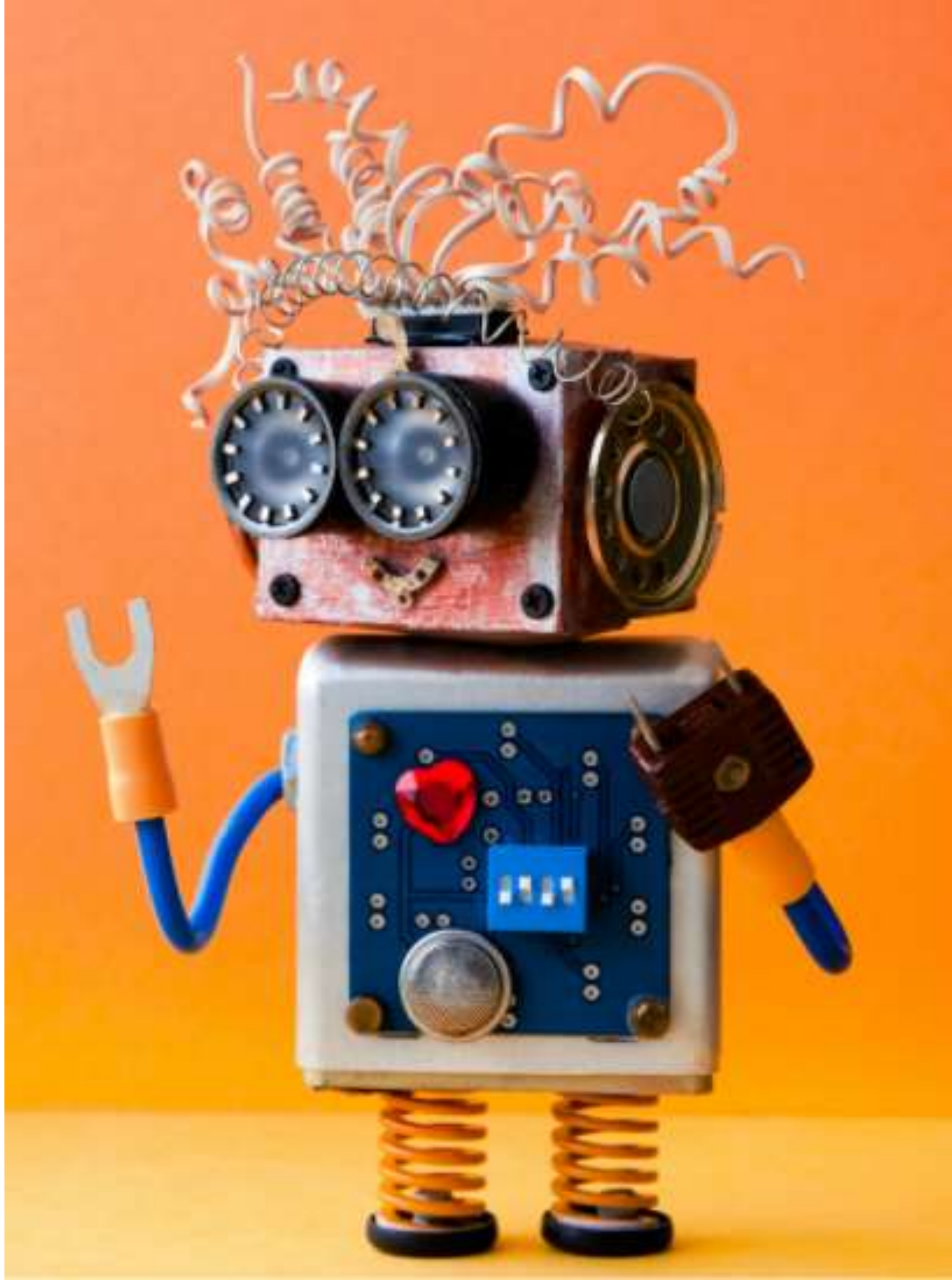
	Date	BTC-USD Price	ETH-USD Price	LTC-USD Price
0	9/17/2014	457.334015	NaN	5.058500
1	9/18/2014	424.440002	NaN	4.685230
2	9/19/2014	394.795990	NaN	4.327770
3	9/20/2014	408.903992	NaN	4.286440
4	9/21/2014	398.821014	NaN	4.245920
...	...	...	...	...
2380	3/28/2021	55950.746090	1691.355957	185.028488
2381	3/29/2021	57750.199220	1819.684937	194.474777
2382	3/30/2021	58917.691410	1846.033691	196.682098
2383	3/31/2021	58918.832030	1918.362061	197.499100
2384	4/1/2021	59095.808590	1977.276855	204.112518

2385 rows × 4 columns

```
[3]: # Use matplotlib on the Pandas DataFrame to plot the data
     investments_df.plot(x = 'Date', y = 'BTC-USD Price', label = 'Bitcoin Price', linewidth = 3, figsize = (16, 8));
     plt.ylabel('Price [$]')
     plt.title('My First Data Visualization Exercise!')
     plt.legend(loc = "upper right")
     plt.grid()
```

FINAL END-OF-DAY CAPSTONE PROJECT





CAPSTONE PROJECT TASKS

- In this project, we will visualize stock prices using Seaborn and Matplotlib. 3 Stocks are considered including Facebook (FB), Twitter (TWTR) and Netflix (NFLX).
- Using the *stock_daily_prices.csv* and *stocks_daily_returns.csv* dataset included in the course/workshop package, please do the following:
  1. Import both datasets using Pandas.
  2. Using Matplotlib, plot lineplots that display all 3 stocks daily prices on one single figure.
  3. Using Matplotlib, plot 3 stocks daily prices on multiple subplots.
  4. Using Matplotlib, plot the 3 plots on subplots next to each other (all figures in one row).
  5. Using Matplotlib, plot the scatterplot between Facebook and Twitter daily returns.
  6. Using Seaborn, plot similar scatterplot between Facebook and Twitter daily returns.
  7. Assume that you now expanded your portfolio to include additional stocks such as Amazon (AMZN) and Google (GOOG). You decided to become bullish on Twitter and you allocated 60% of your assets in it. You also decided to equally divide the rest of your assets in other stocks (AMZN, FB, GOOG, NFLX). Using Matplotlib, plot a pie chart that shows these allocations. Use 'explode’ attribute to increase the separation between TWTR and the rest of the portfolio.
  8. Using Matplotlib, plot the histogram for FB returns using 40 bins with red color. Display the mean and Standard deviation on top of the figure.
  9. Using Seaborn, plot a heatmap that shows the correlations between stocks daily returns.
  10. Plot a 3D plot showing all daily returns from FB, TWTR and NFLX [External Research is required].