Introduction To Artificial Intelligence and Python

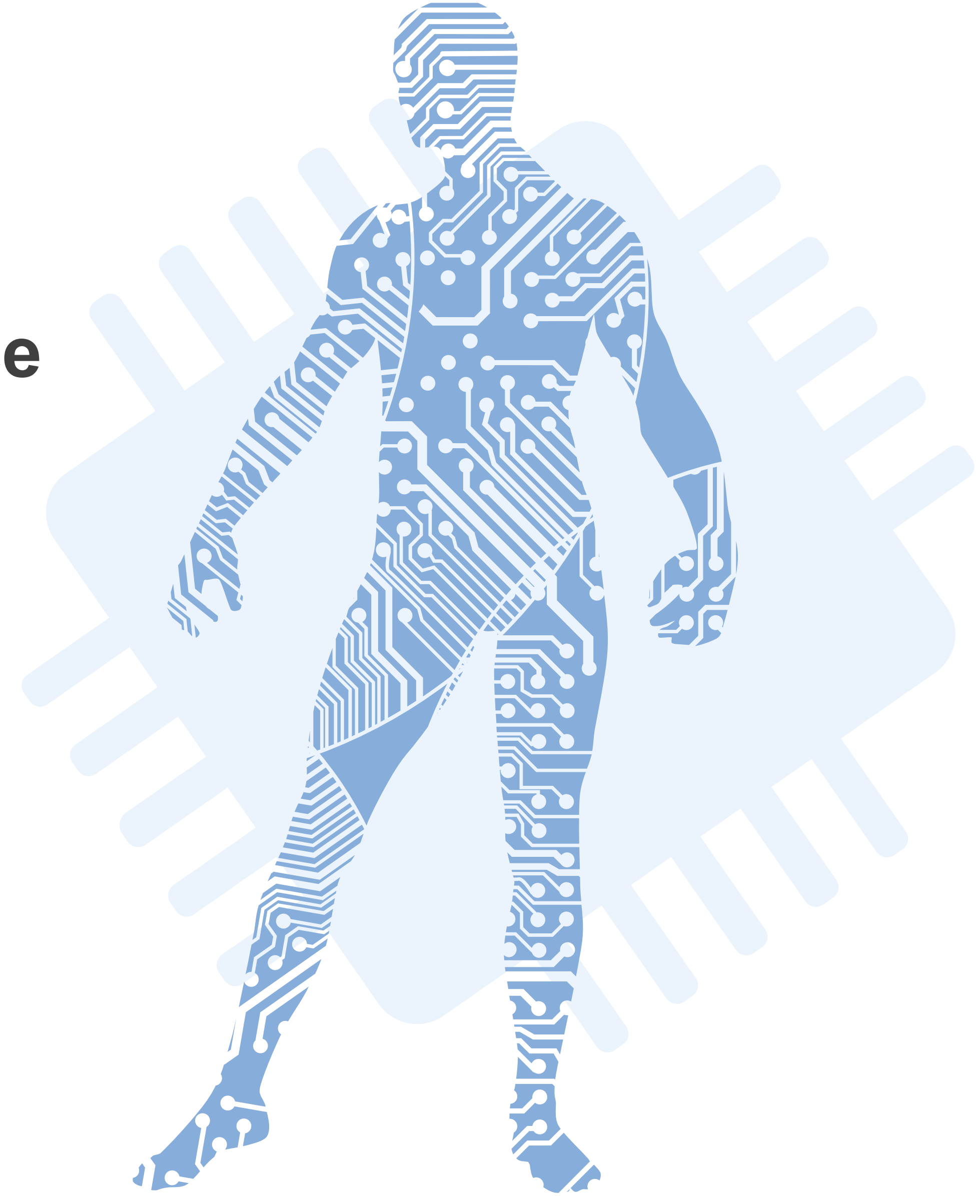
Presented by Nour Droubi & Alaa Maarouf

Agenda Of The Day

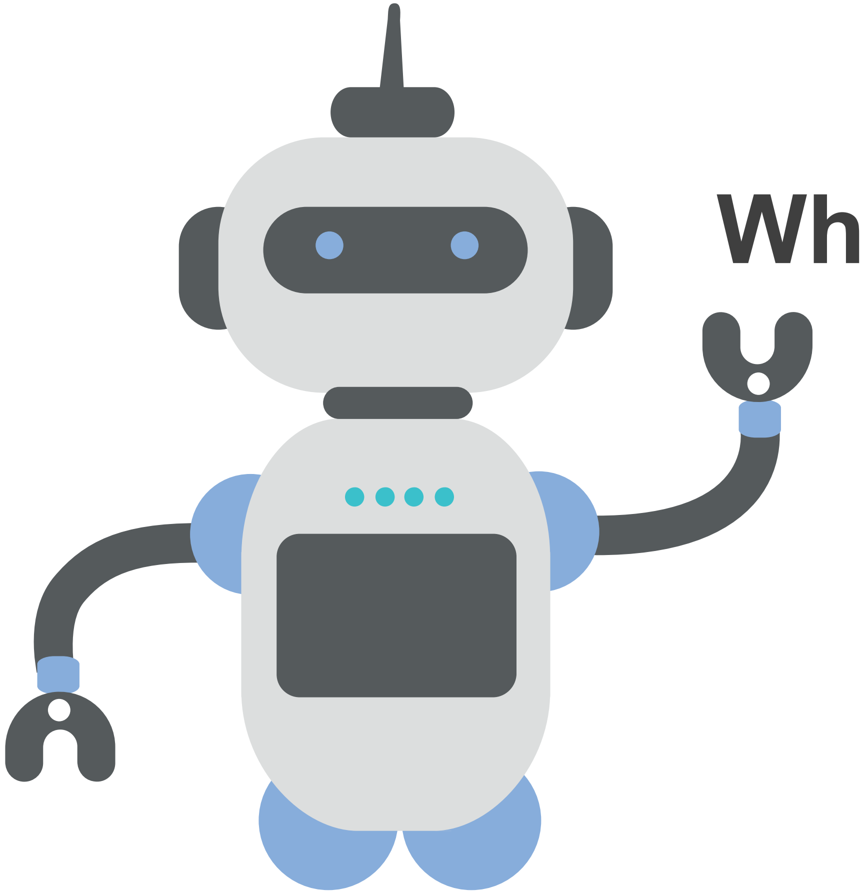
01 **Intro to Artificial Intelligence**

02 **Intro to Python**

03 **Hands-on Exercises**

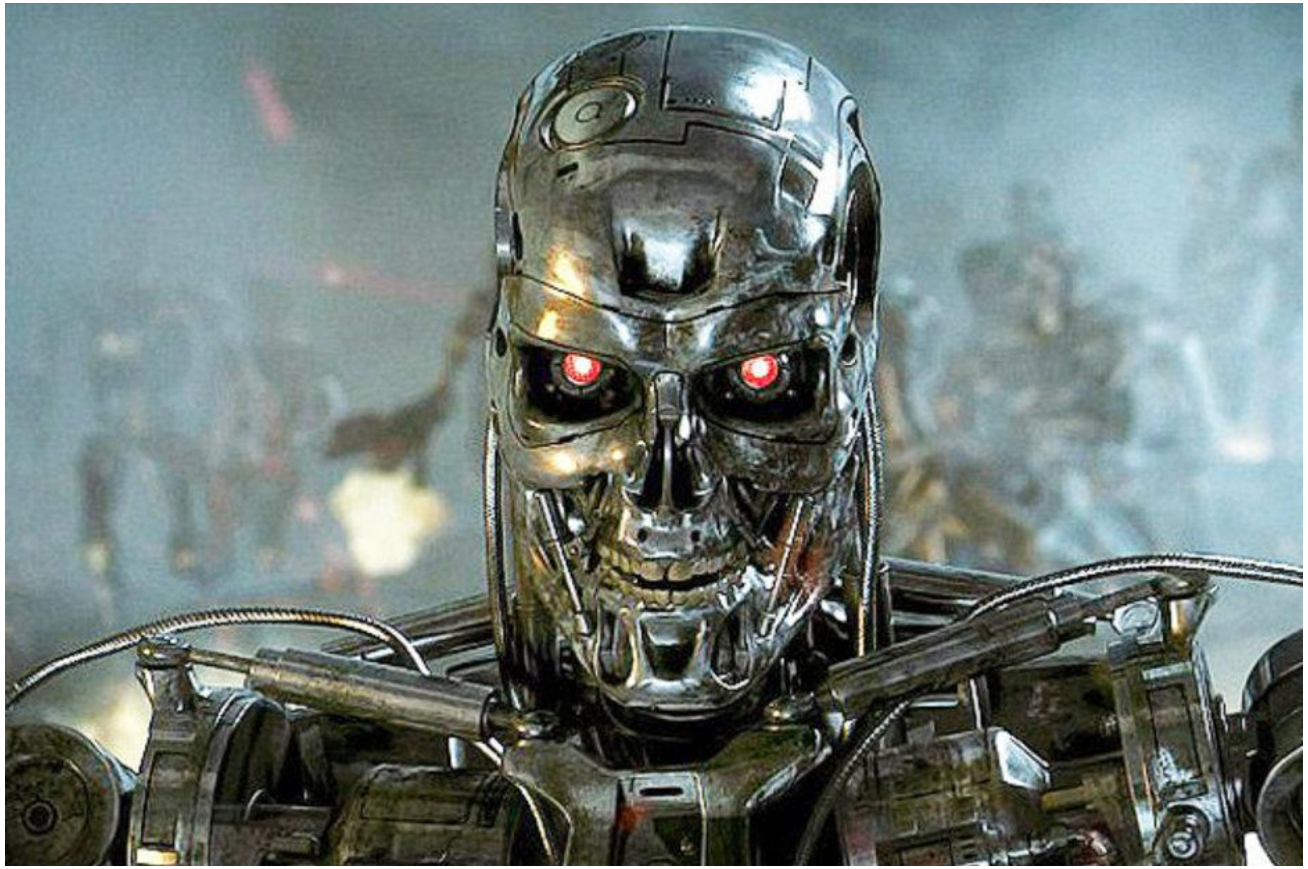


Let’s Talk Some AI



What is AI all about?

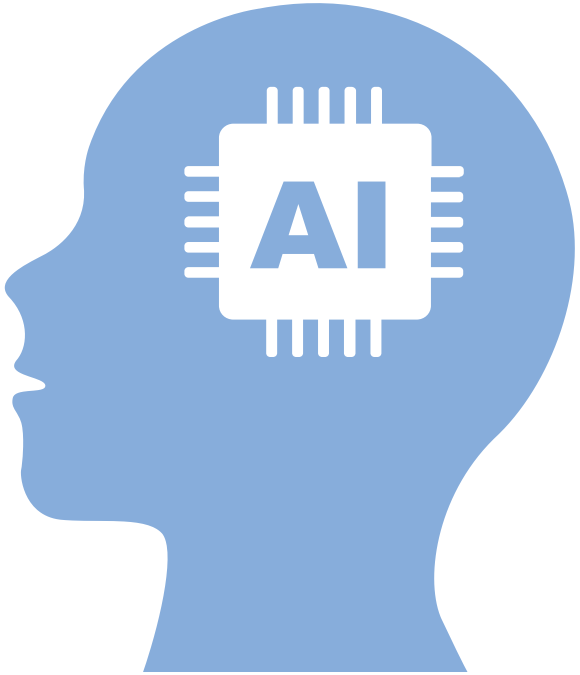
Is This AI?



Define Artificial Intelligence

AI allows machines to:

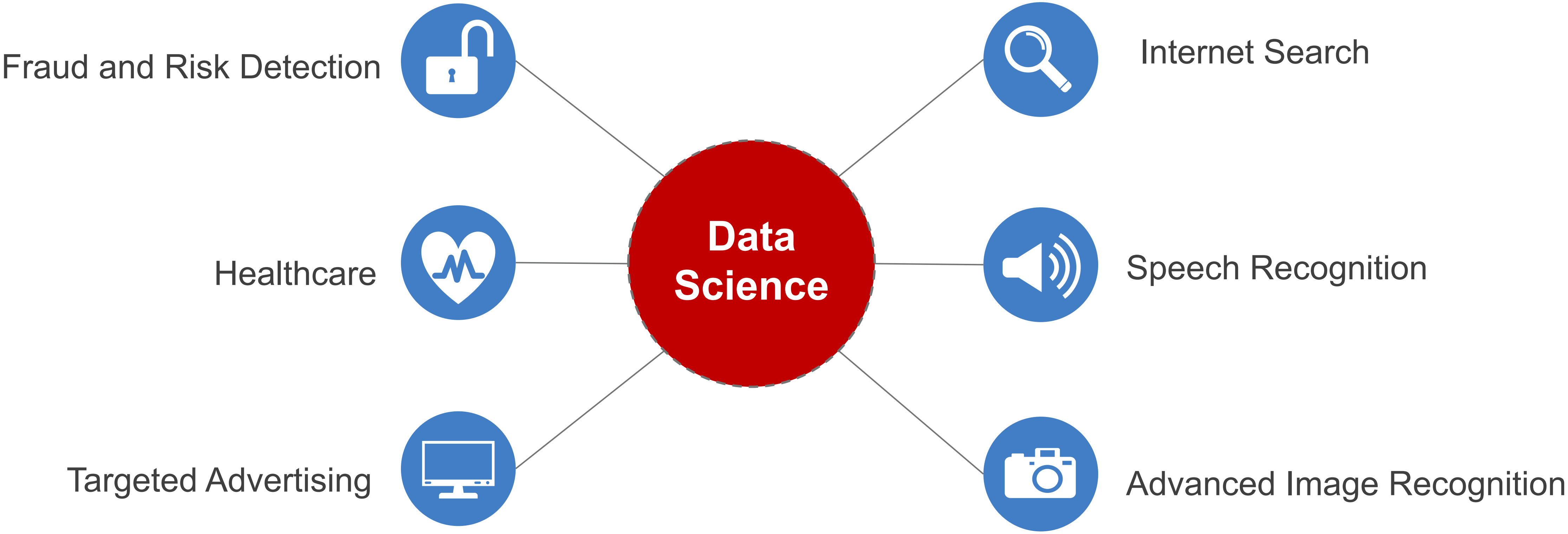
- process large amounts of data
- recognize hidden patterns in the data
- to perform human-like tasks
- learn from experience
- adjust to unseen input

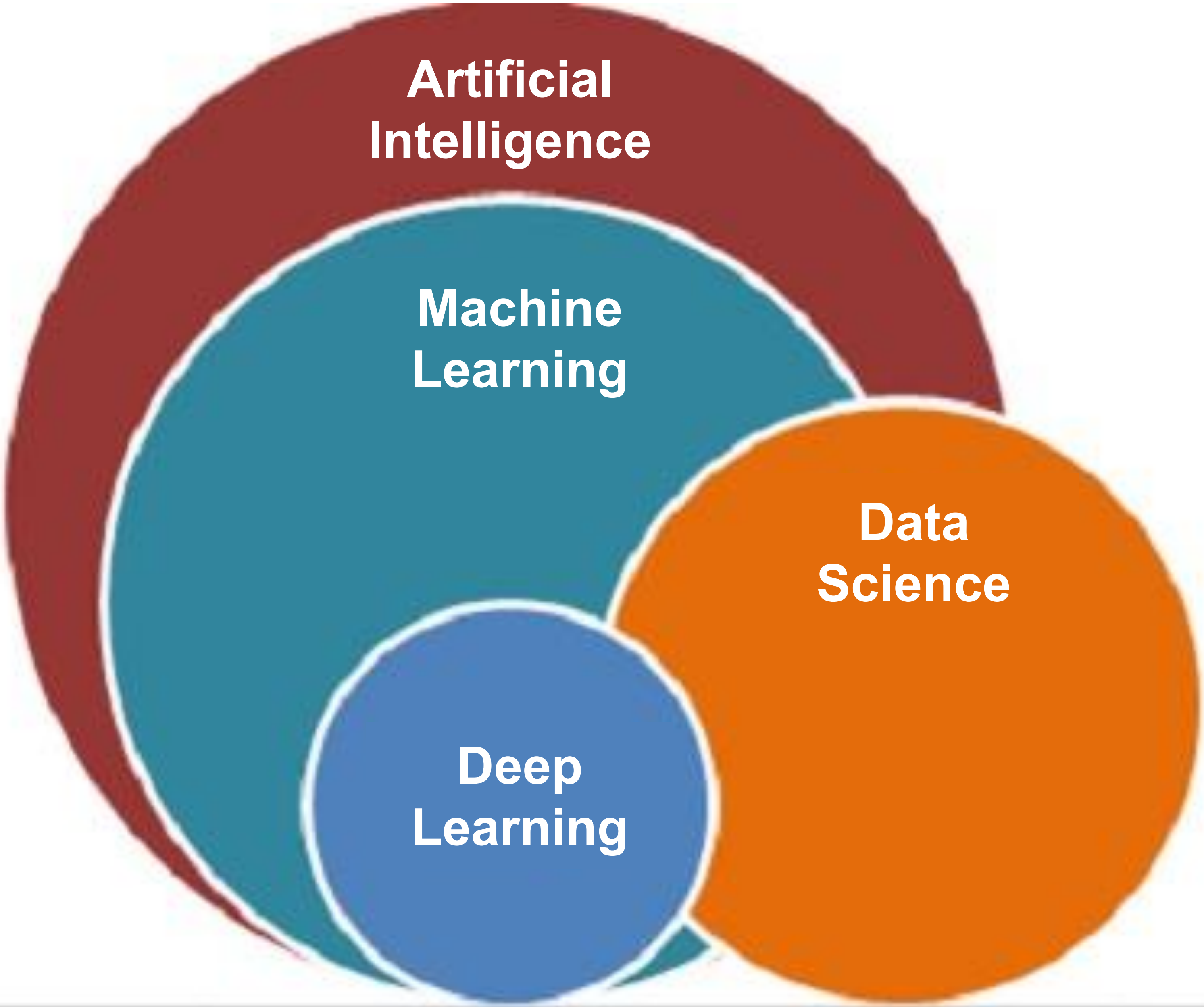


Define Artificial Intelligence

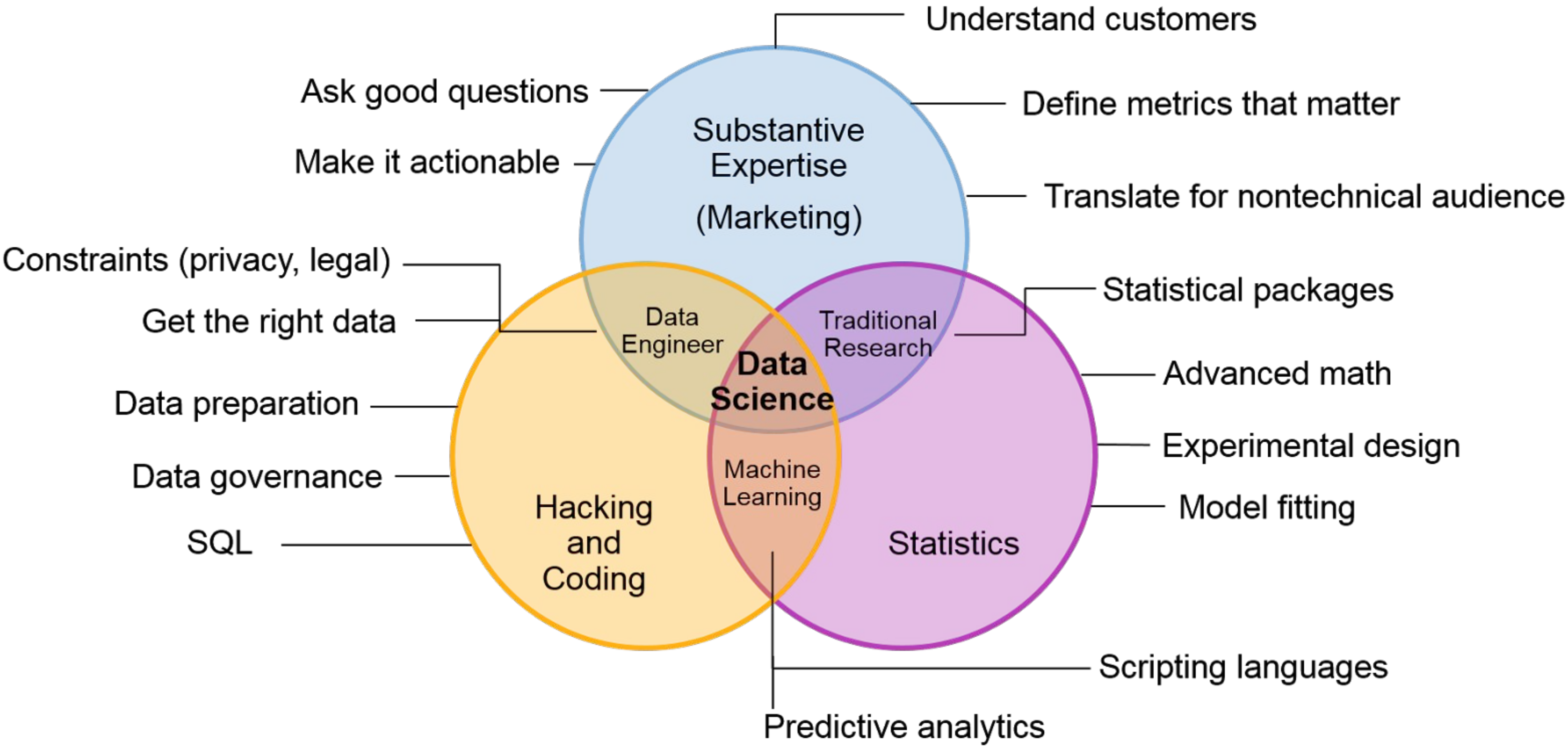
1. Narrow AI (DeepBlue, AlphaGo)
2. General AI
3. Superintelligent AI

Some Applications of AI



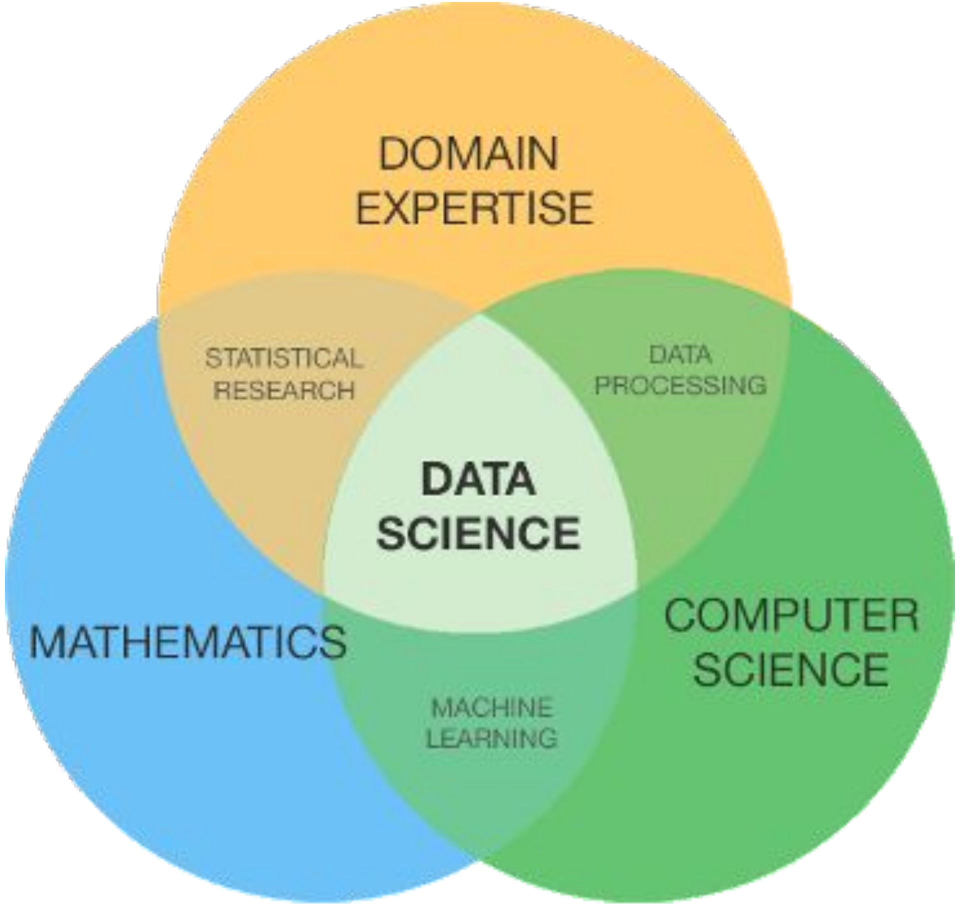


Source: https://www.aitrends.com/data-science/machine-learning-engineer-vs-data-scientist-who-does-what/

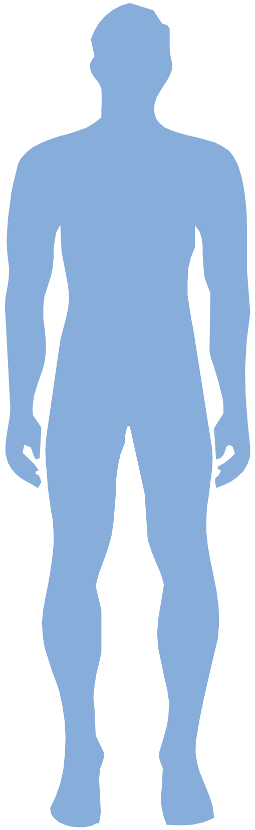


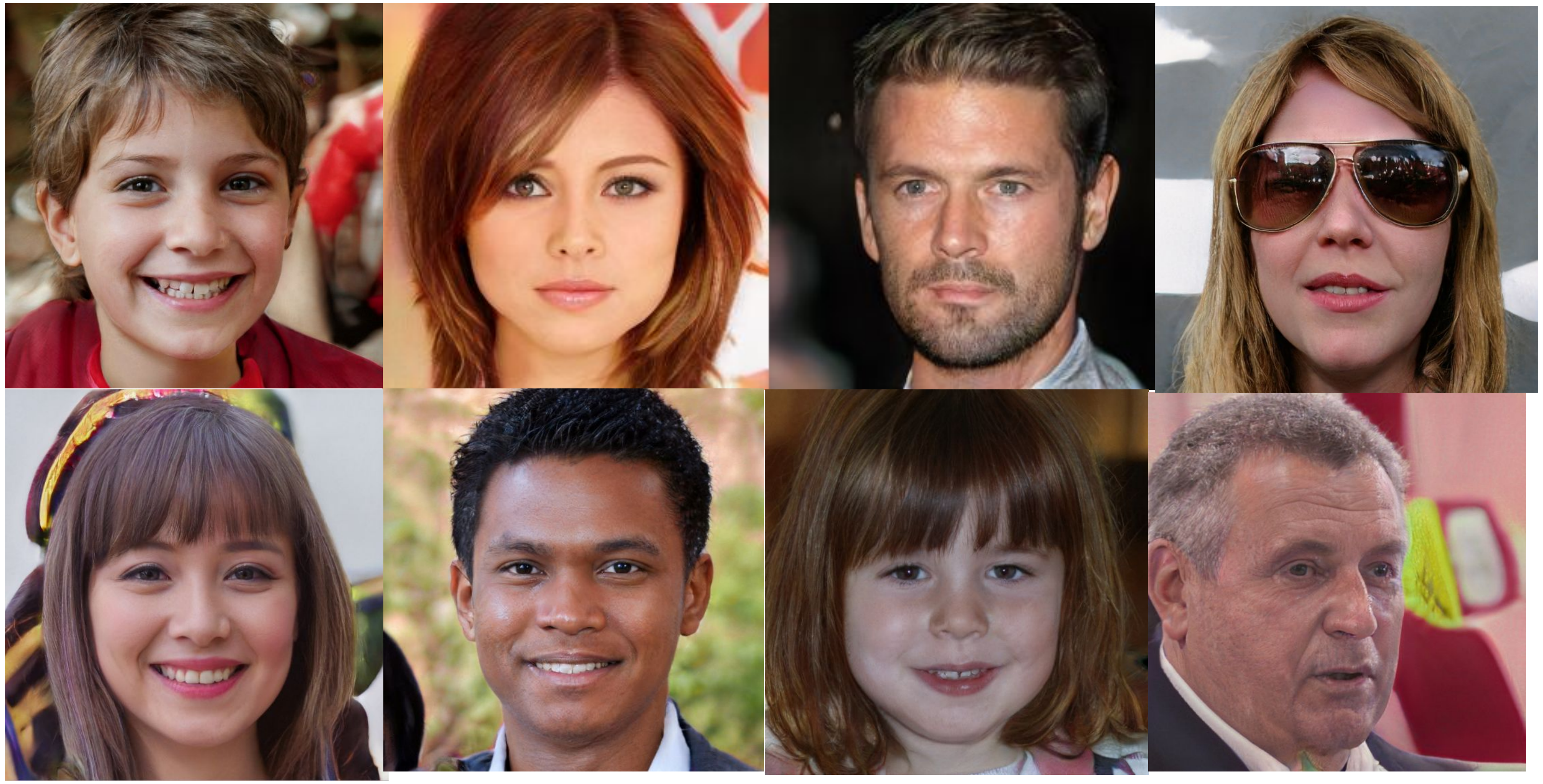
Source: http://www.prooffreader.com/2016/09/battle-of-data-science-venn-diagrams.html/

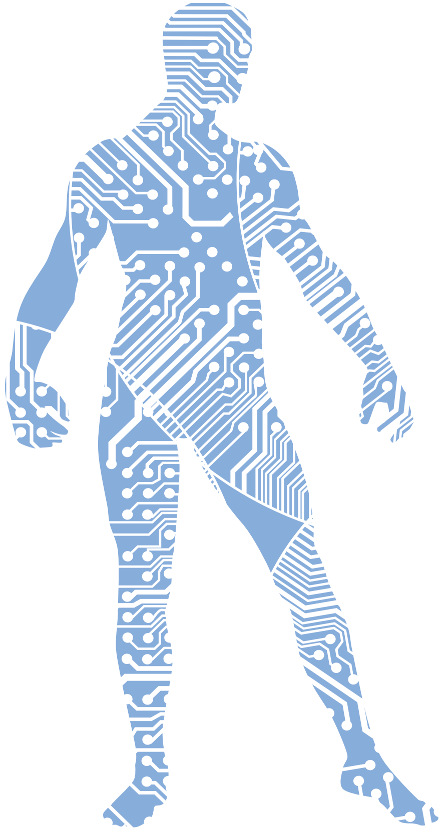
An Interdisciplinary Field



Which Ones Are Real?!

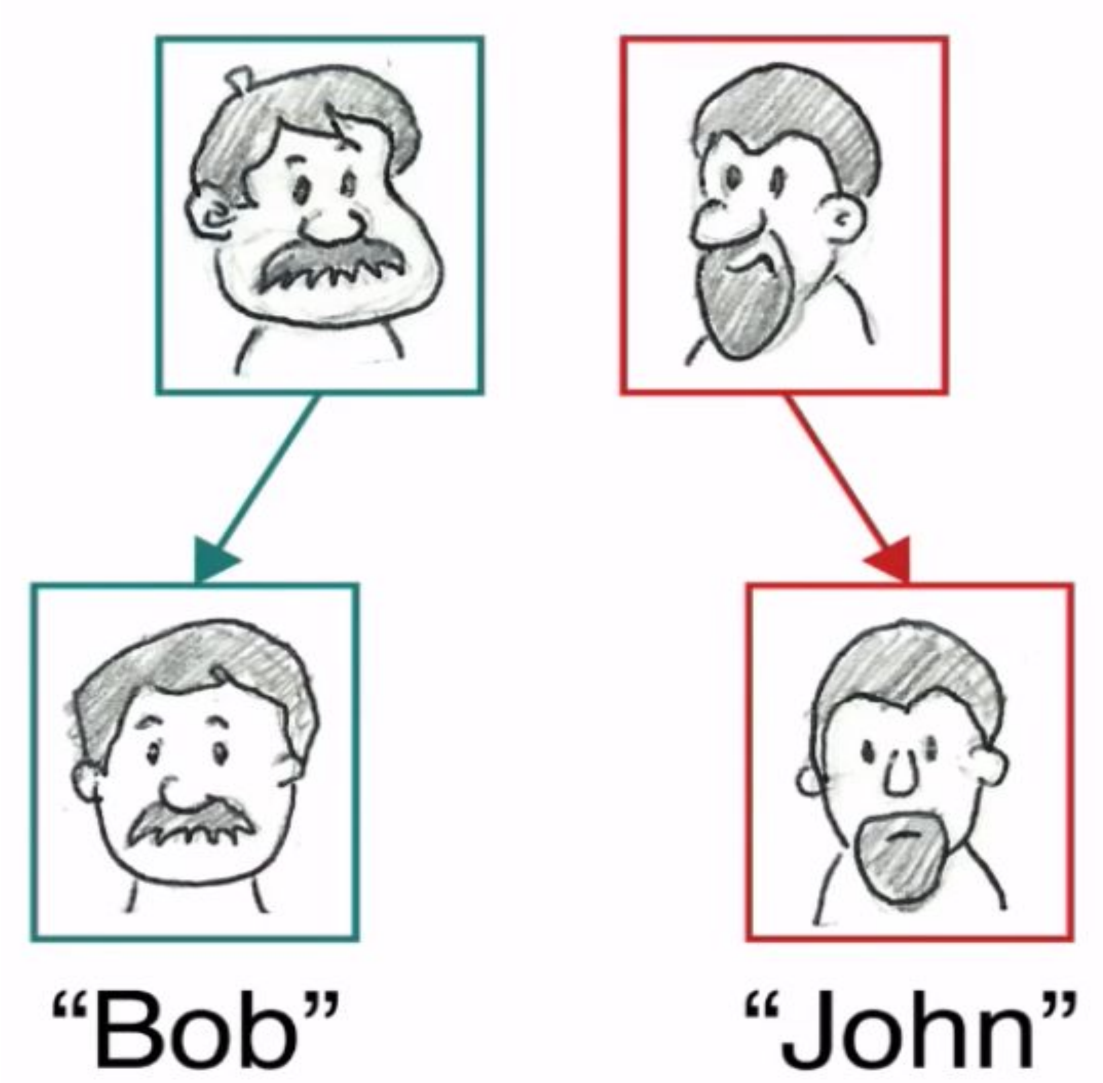


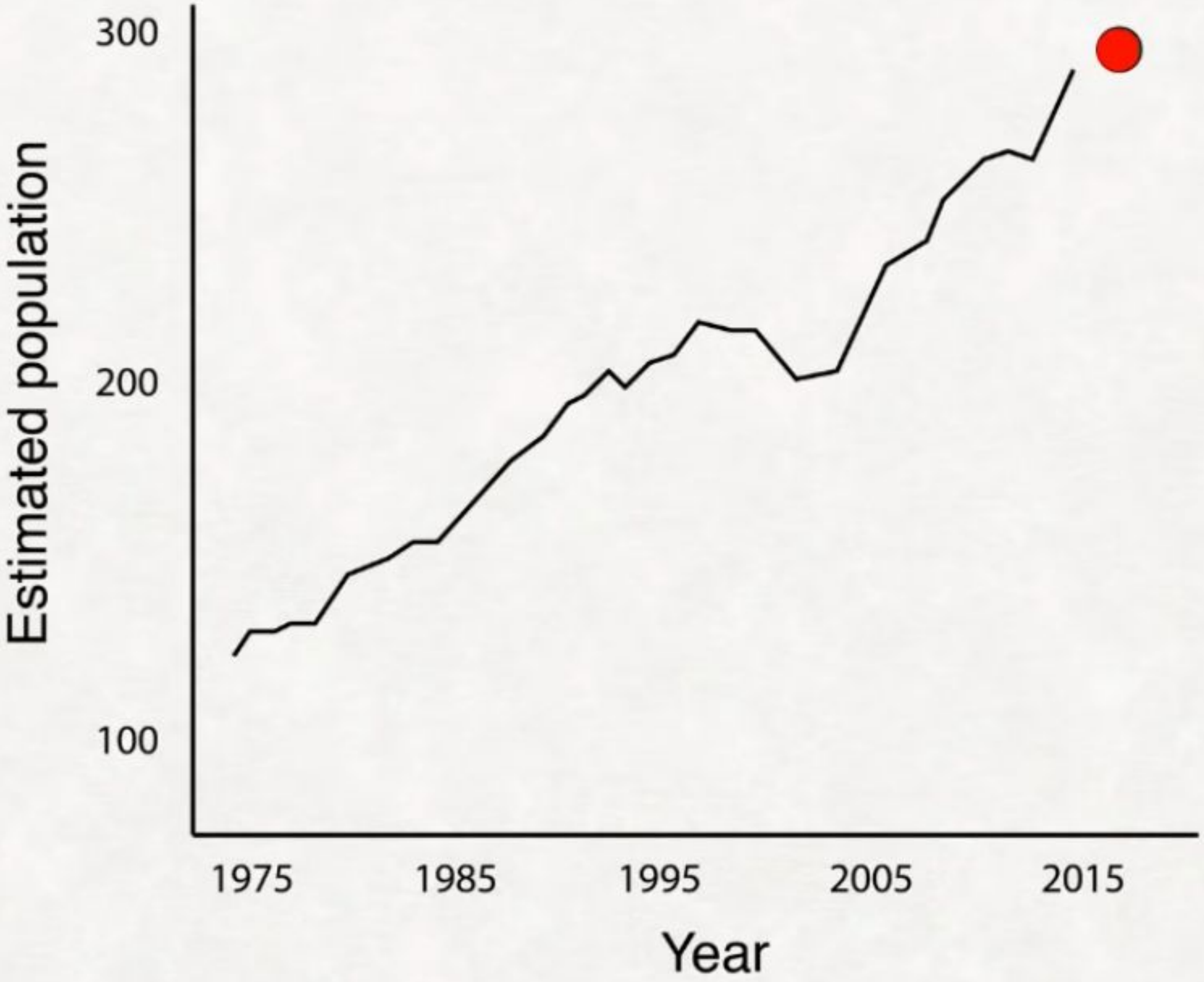




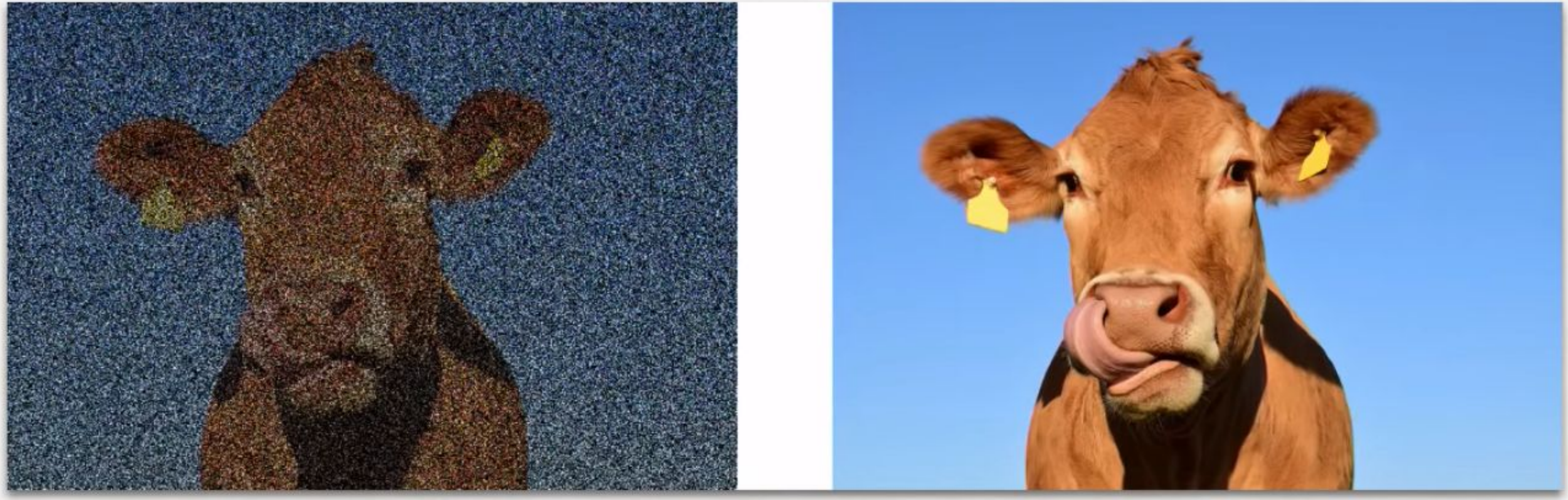
https://thispersondoesnotexist.com/

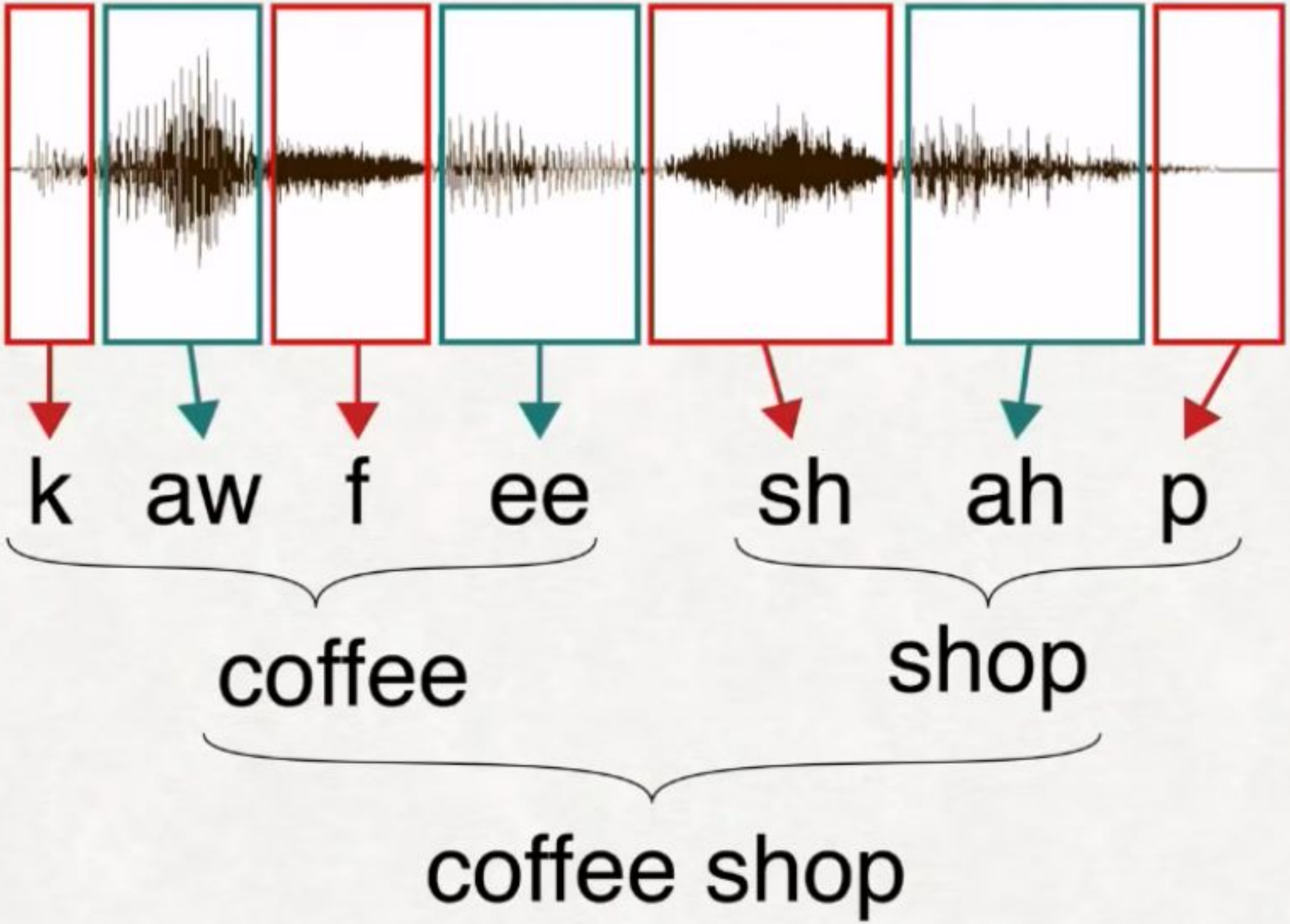
Face Recognition




300
200
100
Estimated population
1975
1985
1995
2005
2015
Year

Denoising Images

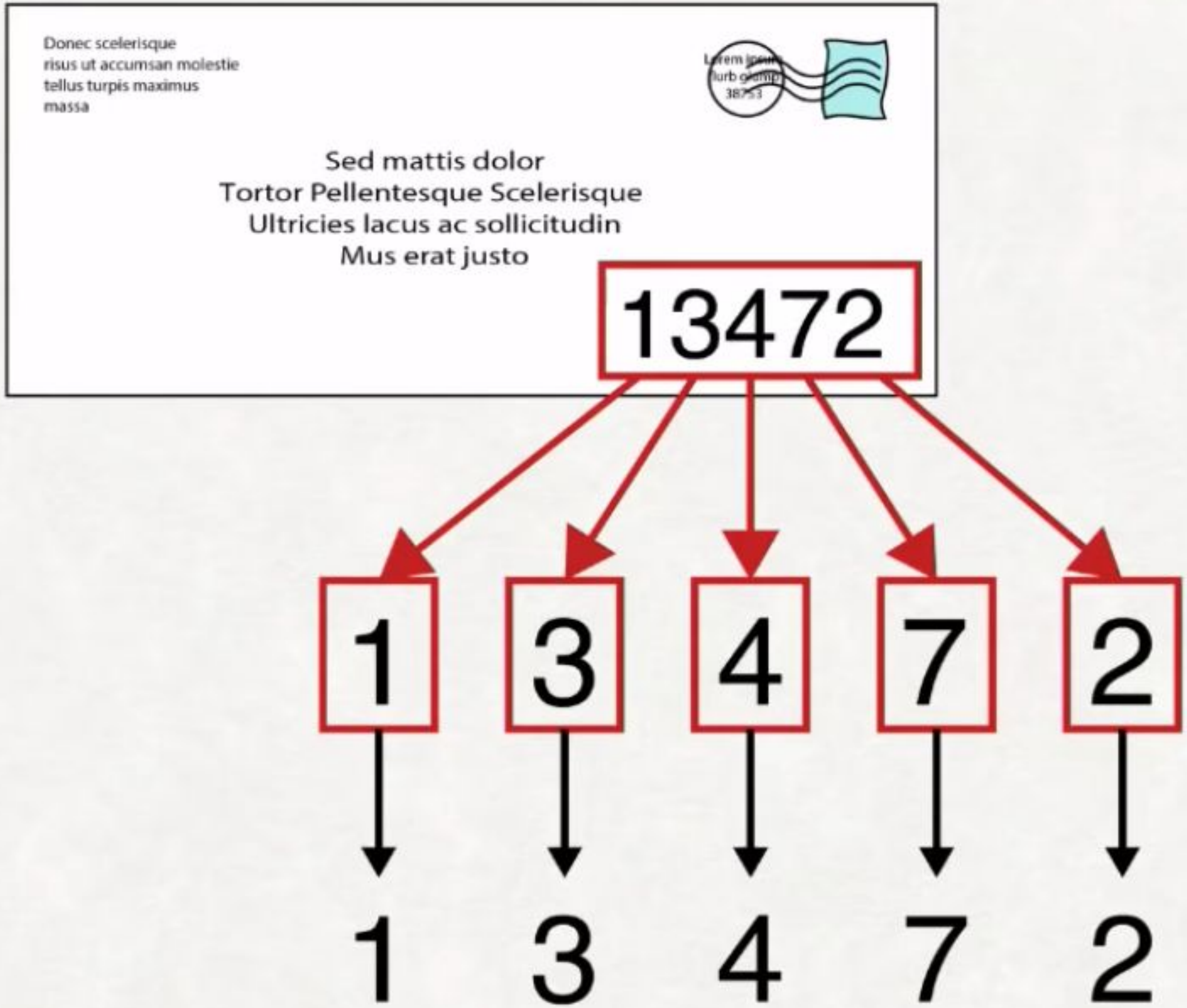




k aw f ee sh ah p

coffee shop

coffee shop



Donec scelerisque
risus ut accumsan molestie
tellus turpis maximus
massa

Sed mattis dolor
Tortor Pellentesque Scelerisque
Ultricies lacus ac sollicitudin
Mus erat justo

13472

1 3 4 7 2

1 3 4 7 2

Why AI?

- Increase amount of data available (advancement in IoT and Big Data)
- Increase usage of social media
- Huge amounts of new articles, videos,… published every day

→ Need for an automated way to process all this data and produce meaningful insights!

Why AI?

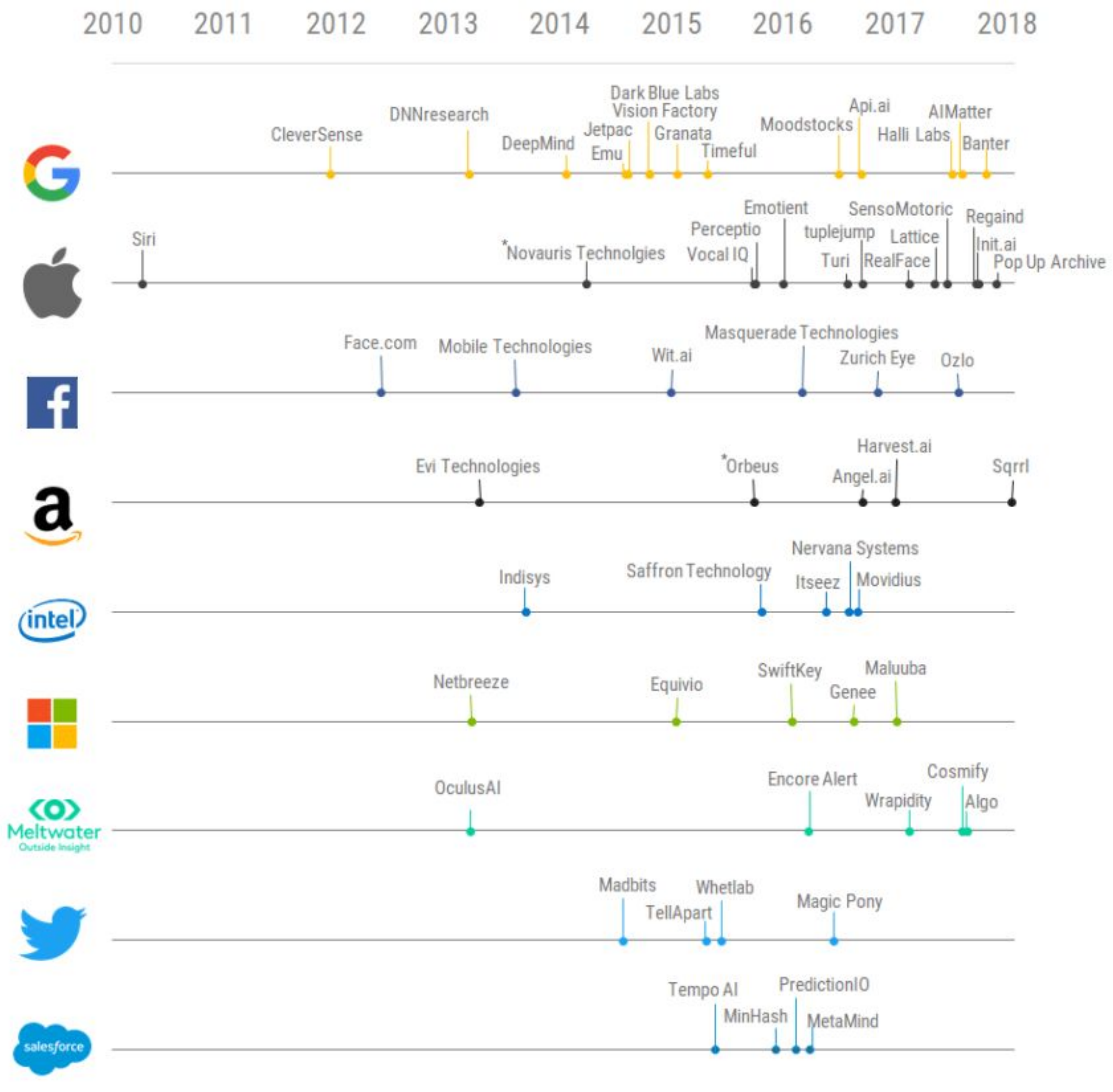
- Automating repetitive learning and discovery through data
- Adding intelligence to existing products
- Adapting through progressive learning algorithms (adapt to new data)
- Analyzing more and deeper data (computation power & Big Data)
- AI achieves very high accuracy
- Making the most out of data(best data wins)

The Race For AI:

Google, Intel, Apple In A Rush To Grab Artificial Intelligence Startups

Race To Acquire Top AI Startups Heats Up

Date of acquisition (only includes 1st exits of companies)



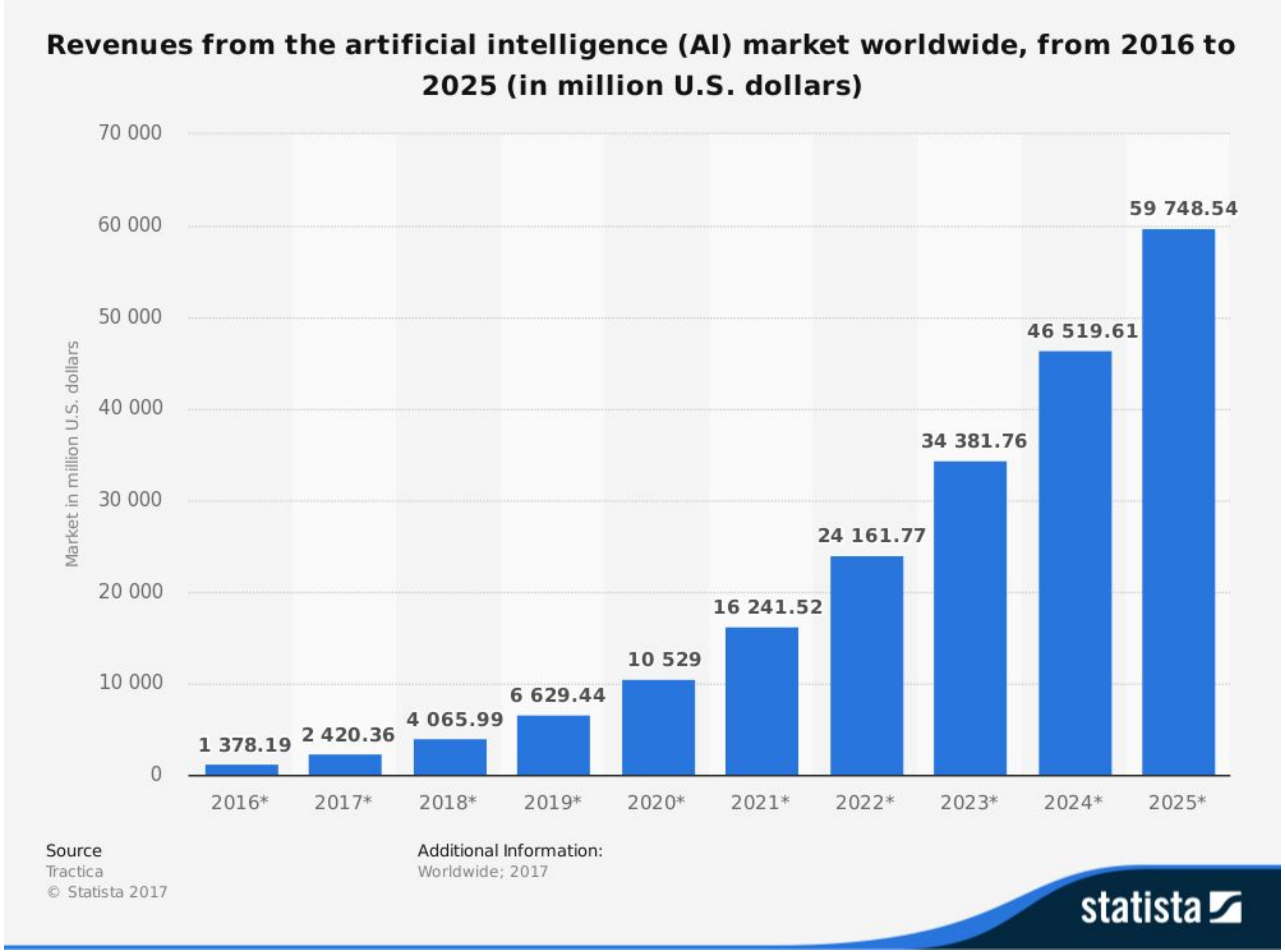
Source: cbinsights.com

*approximate dates of acquisition

CBINSIGHTS

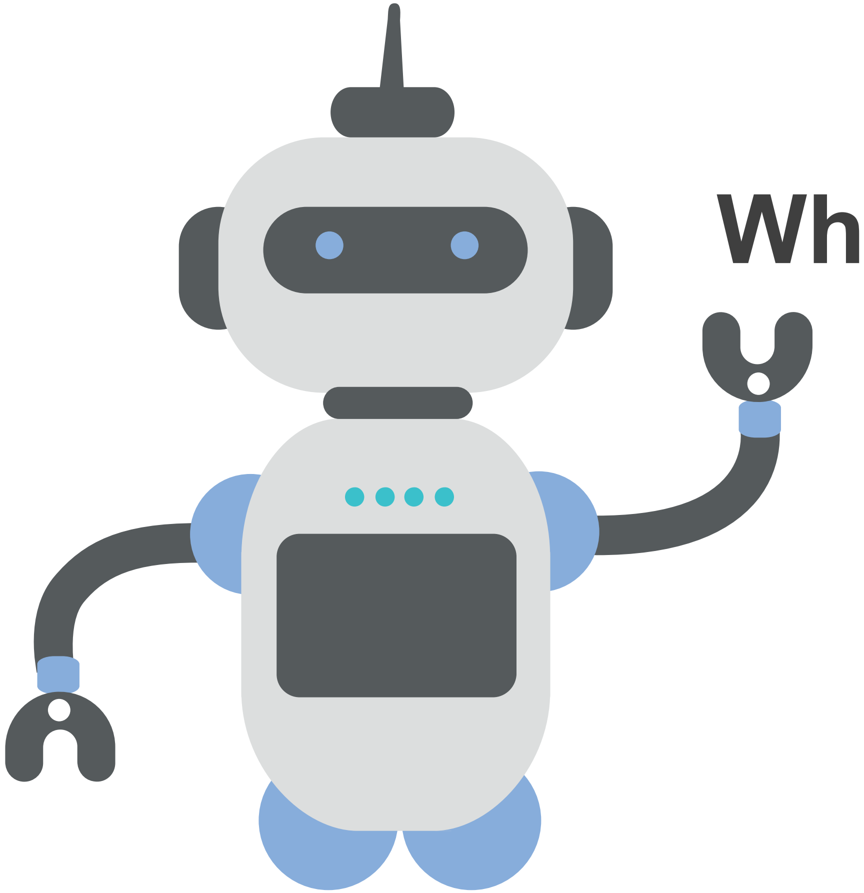
Source: https://www.cbinsights.com/research/top-acquirers-ai-startups-ma-timeline/

Artificial Intelligence market projected to grow 25-fold in eight years!



Source: https://www.moneycontrol.com/news/business/data-story-next-investment-opportunity-artificial-intelligence-market-projected-to-grow-25-fold-in-eight-years-2411419.html

Let's Define Some Terms!

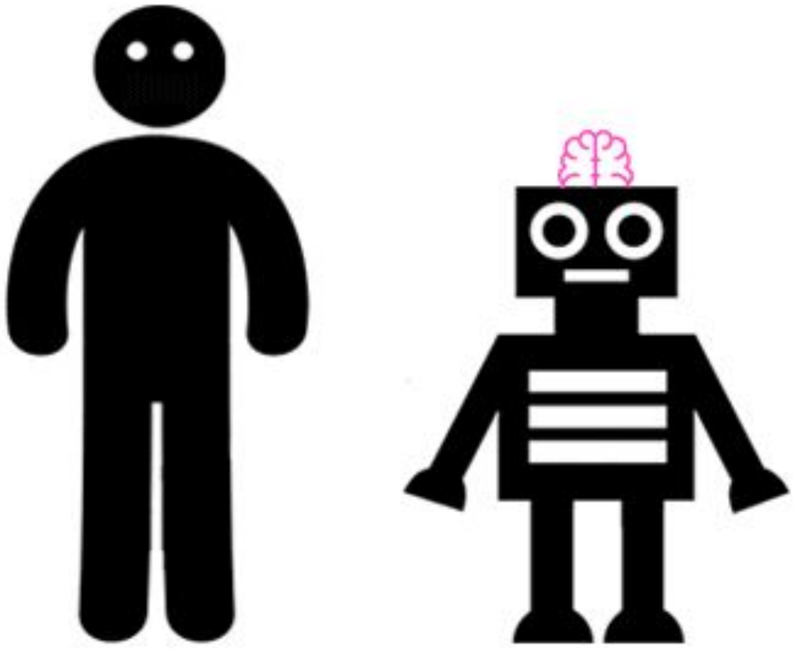


What is AI all about?

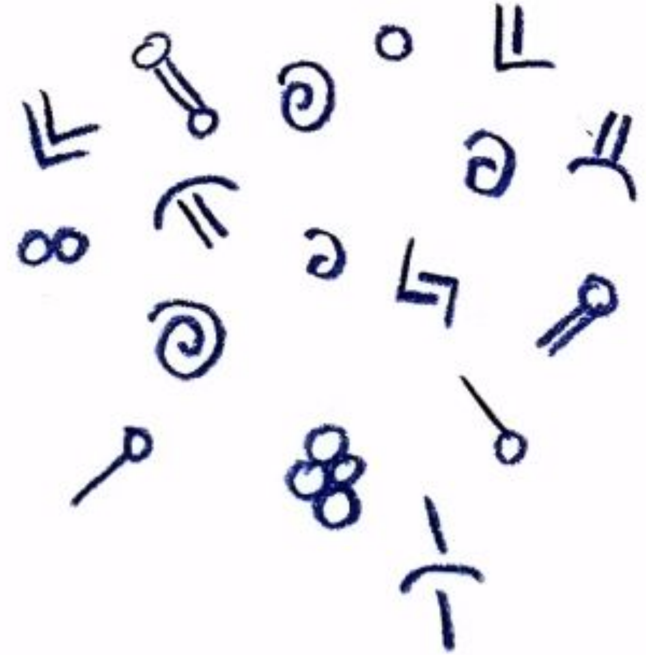
Learning Styles

1. Supervised Learning
2. Unsupervised Learning
3. Reinforcement Learning

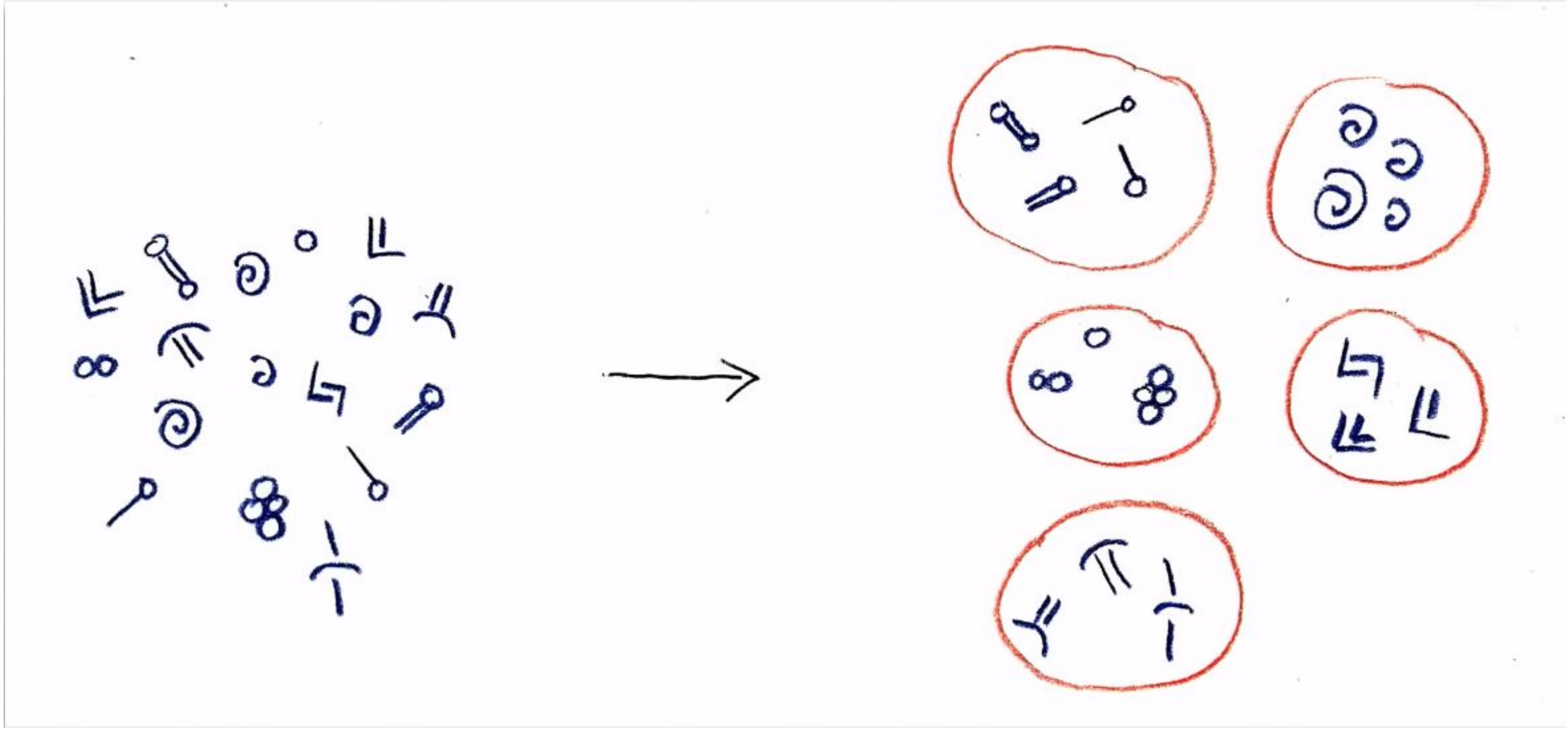
Supervised Learning



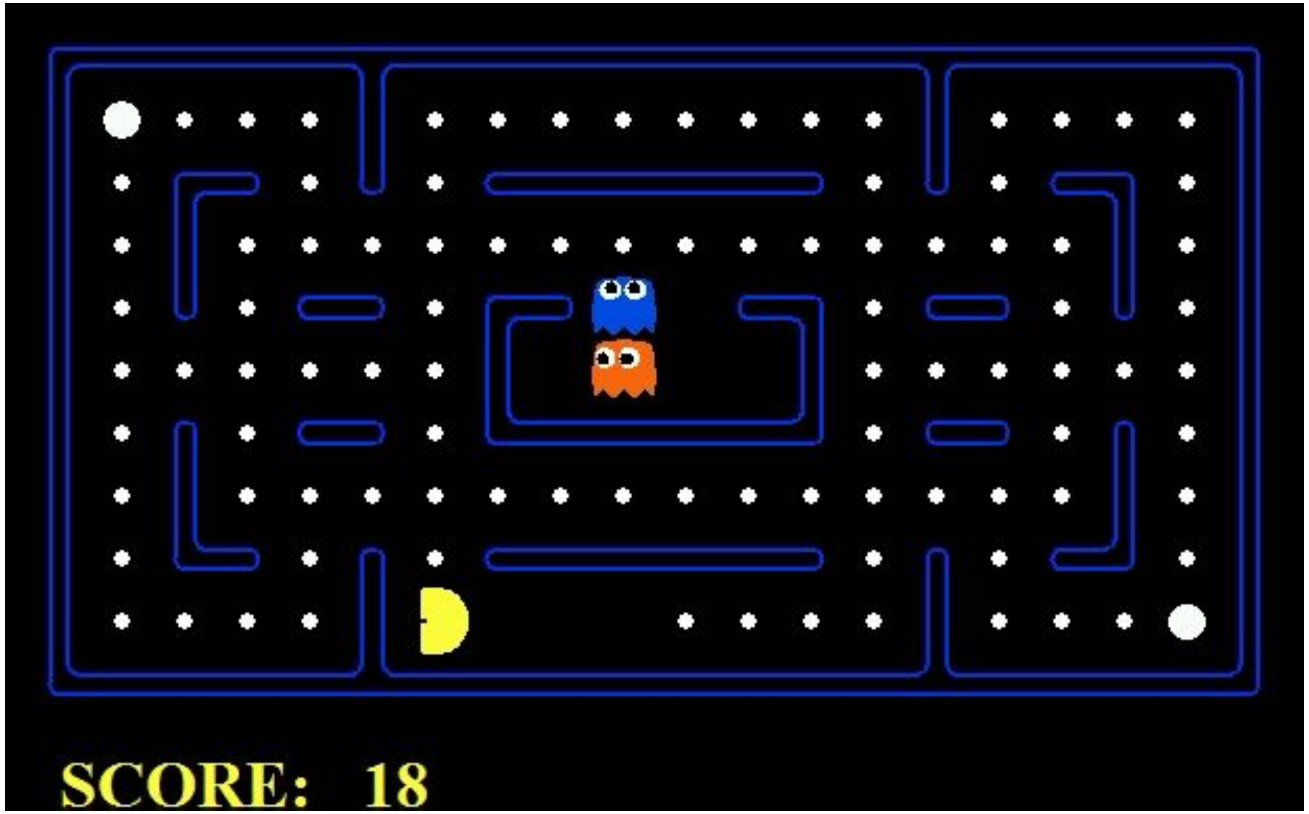
Unsupervised Learning

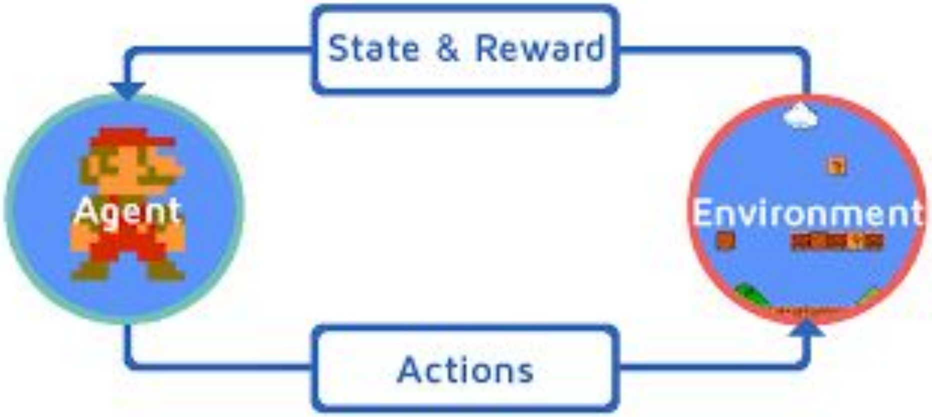


Unsupervised Learning



Reinforcement Learning




State & Reward

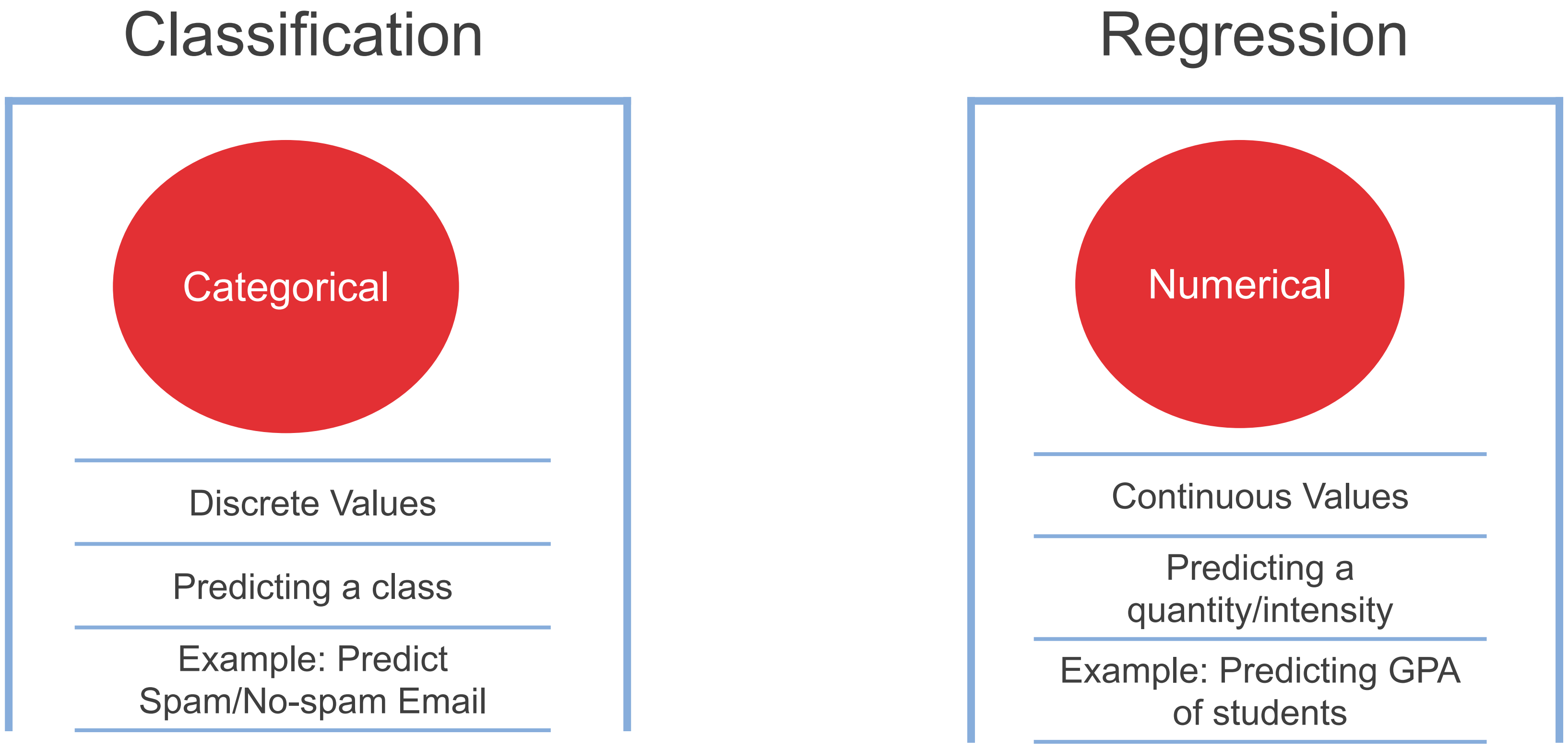
Agent

Environment

Actions

Source:
https://medium.freecodecamp.org/a-brief-introduction-to-reinforcement-learning-7799af5840db

Classification Versus Regression



Classification

Categorical

Discrete Values

Predicting a class

Example: Predict Spam/No-spam Email

Regression

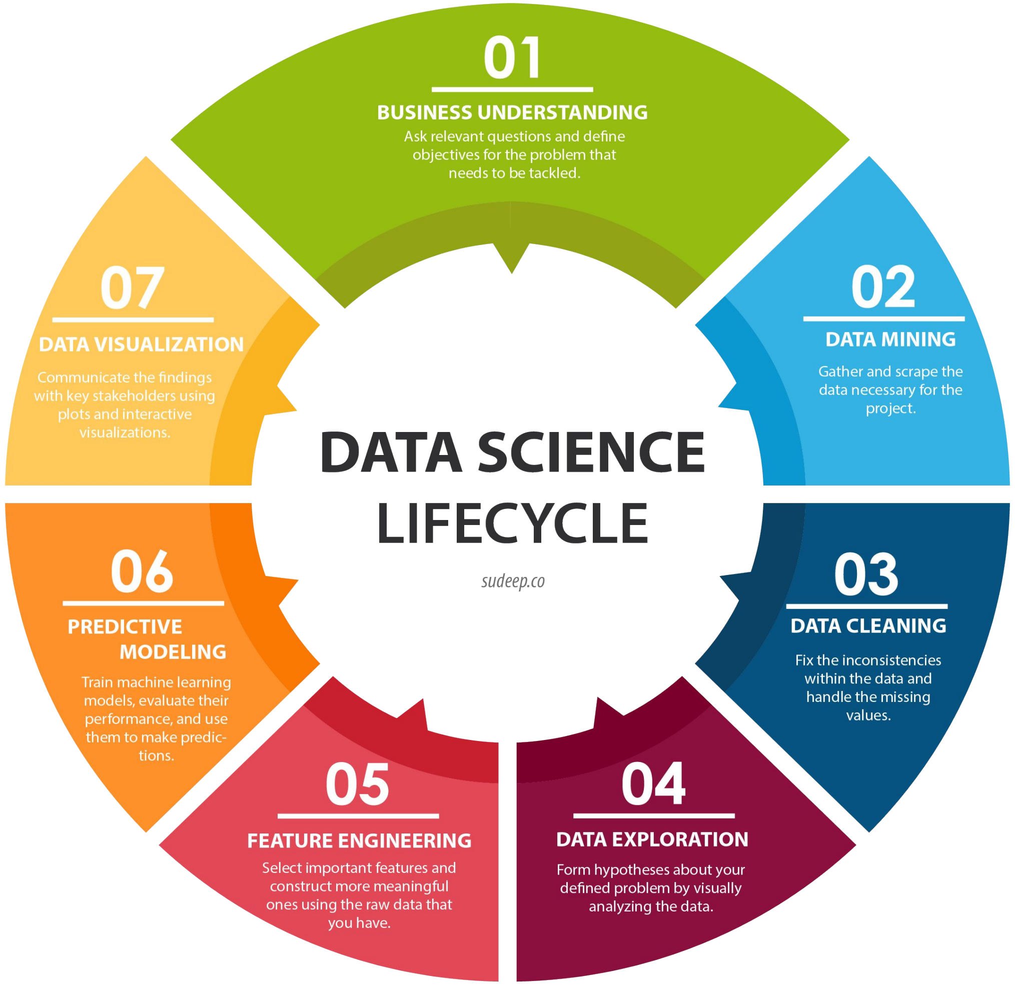
Numerical

Continuous Values

Predicting a quantity/intensity

Example: Predicting GPA of students

Data Science Life Cycle



DATA SCIENCE LIFECYCLE

sudeep.co

01 BUSINESS UNDERSTANDING
Ask relevant questions and define objectives for the problem that needs to be tackled.

02 DATA MINING
Gather and scrape the data necessary for the project.

03 DATA CLEANING
Fix the inconsistencies within the data and handle the missing values.

04 DATA EXPLORATION
Form hypotheses about your defined problem by visually analyzing the data.

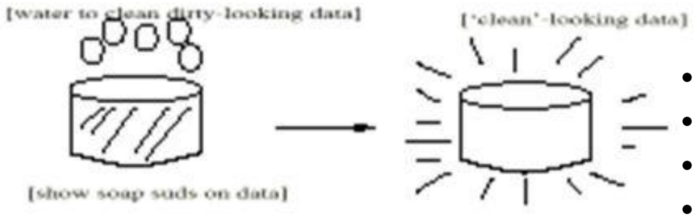
05 FEATURE ENGINEERING
Select important features and construct more meaningful ones using the raw data that you have.

06 PREDICTIVE MODELING
Train machine learning models, evaluate their performance, and use them to make predictions.

07 DATA VISUALIZATION
Communicate the findings with key stakeholders using plots and interactive visualizations.

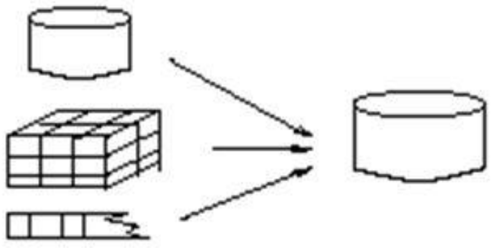
Data Preprocessing

Data cleaning



- Duplicates
- Missing Values
- Outliers
- Incorrect values

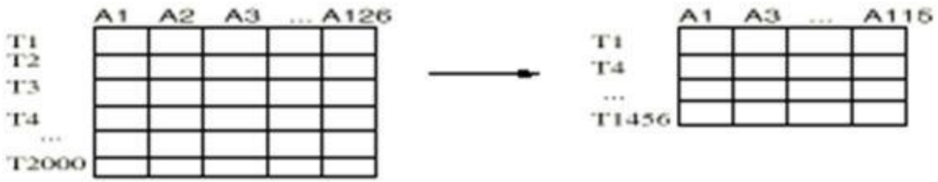
Data Integration



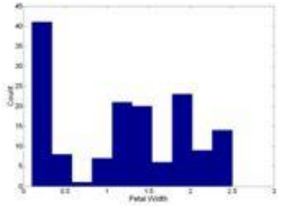
Data transformation

-2, 32, 100, 59, 48 → -0.02, 0.32, 1.00, 0.59, 0.48

Data reduction

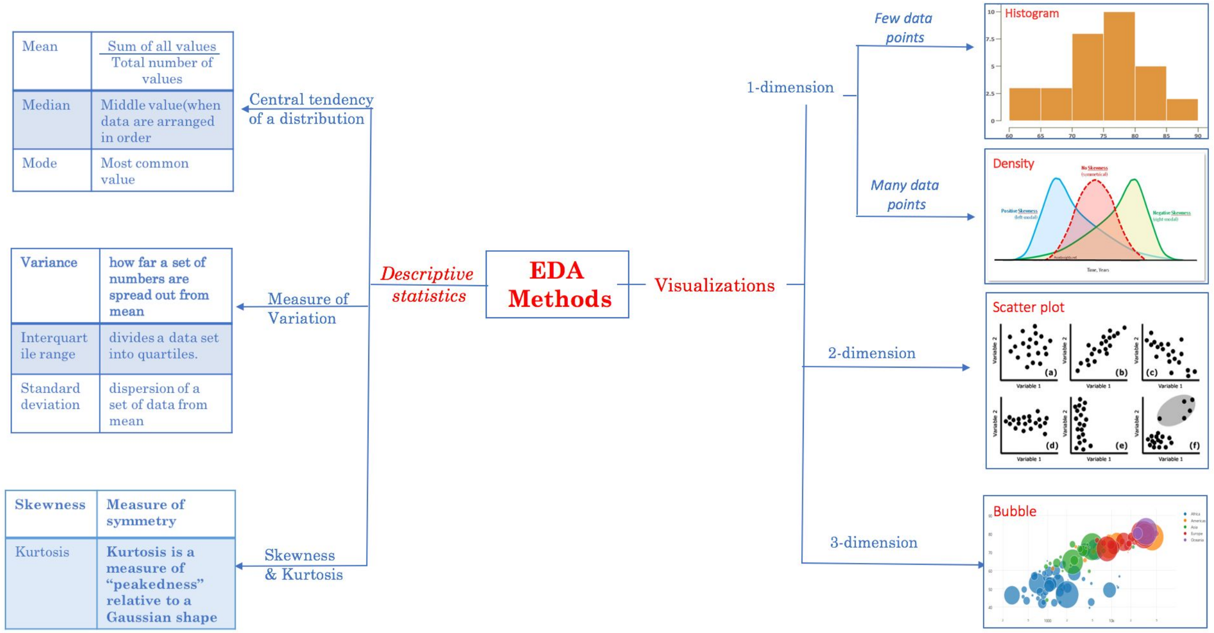


Data Exploratory Analysis



Exploratory Data Analysis

- plots
- graphs
- summary statistics



Mean	Sum of all values / Total number of values
Median	Middle value(when data are arranged in order
Mode	Most common value

Variance	**how far a set of numbers are spread out from mean**
Interquartile range	divides a data set into quartiles.
Standard deviation	dispersion of a set of data from mean

Skewness	**Measure of symmetry**
Kurtosis	**Kurtosis is a measure of "peakedness" relative to a Gaussian shape**

Source: http://www.jeannjoroge.com/significance-of-exploratory-data-anaysis/

Feature Extraction

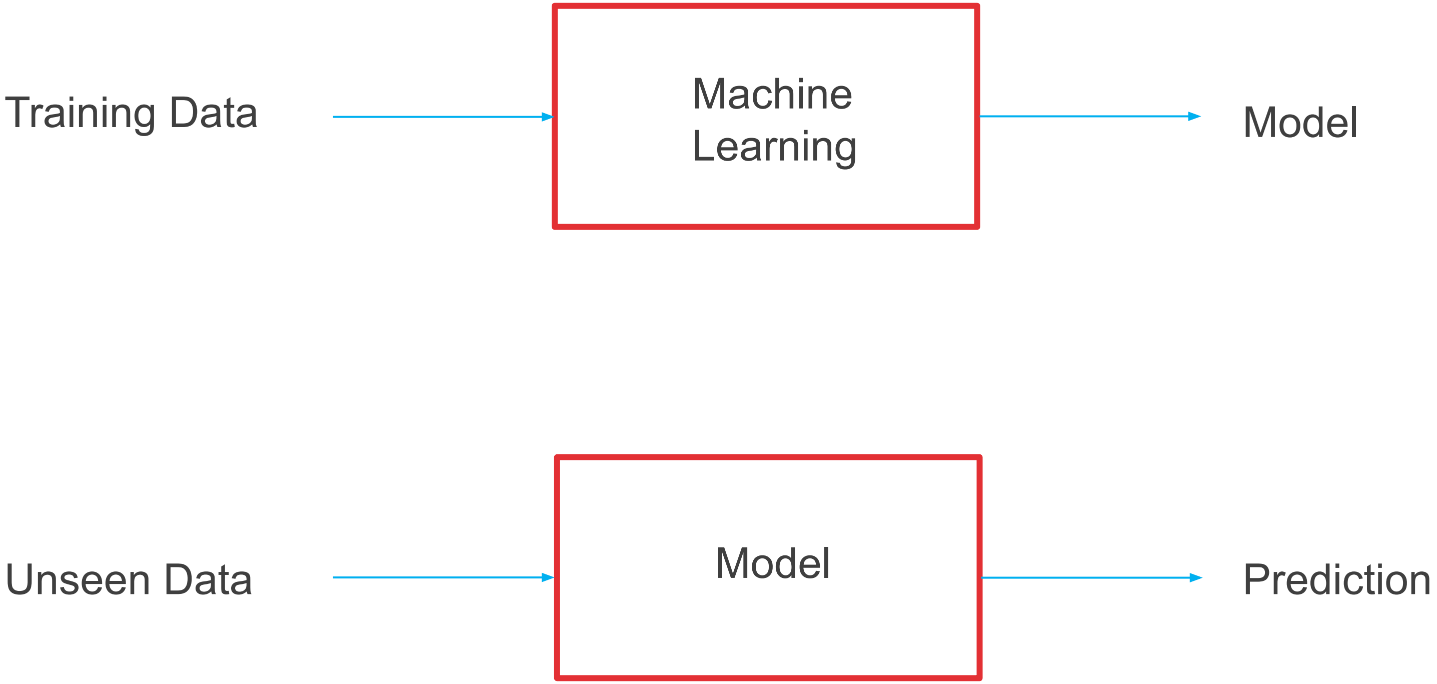
- Extracting important/relevant characteristics that are enclosed within the data

- Transformation of input data into a set of distinctive properties (features) that can well represent the data

- Needs domain-specific expertise to select the features

- Example: Relevant features for a certain disease can be blood pressure and body temperature but not person’s height

Predictive Modeling



Training Data

Machine Learning

Model

Unseen Data

Model

Prediction

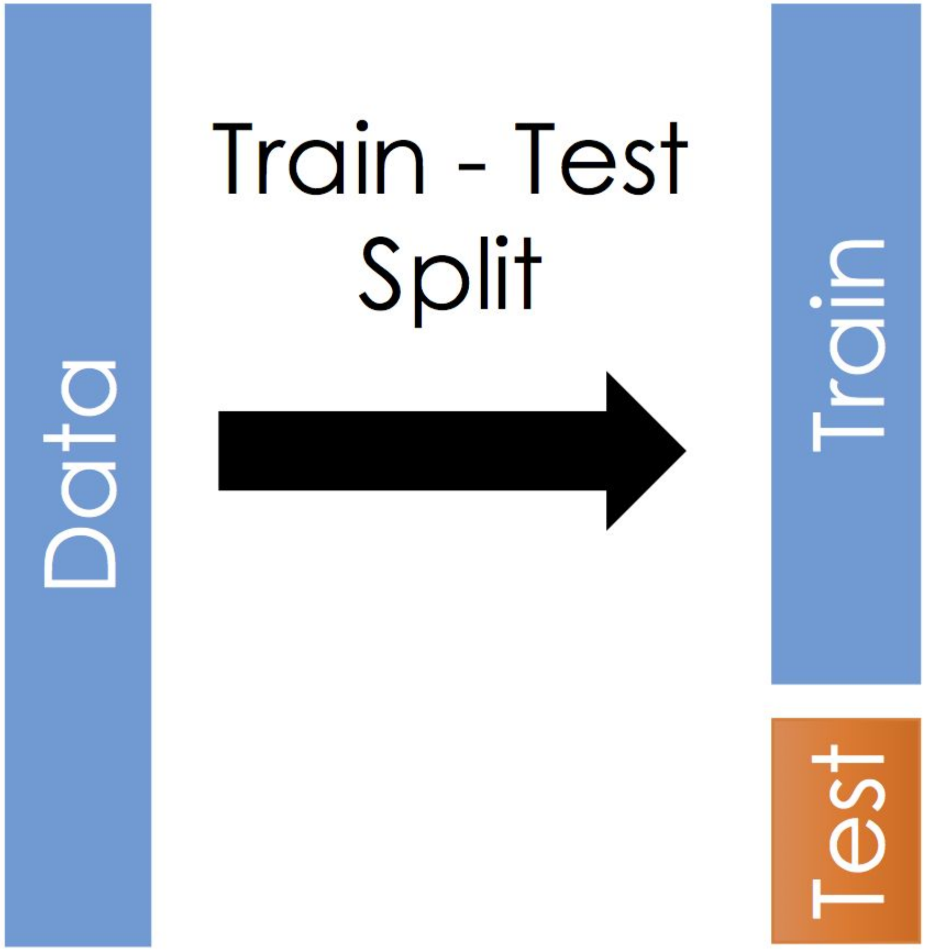
Split Your Dataset

We need the network to be able to generalize.

Split the training dataset:

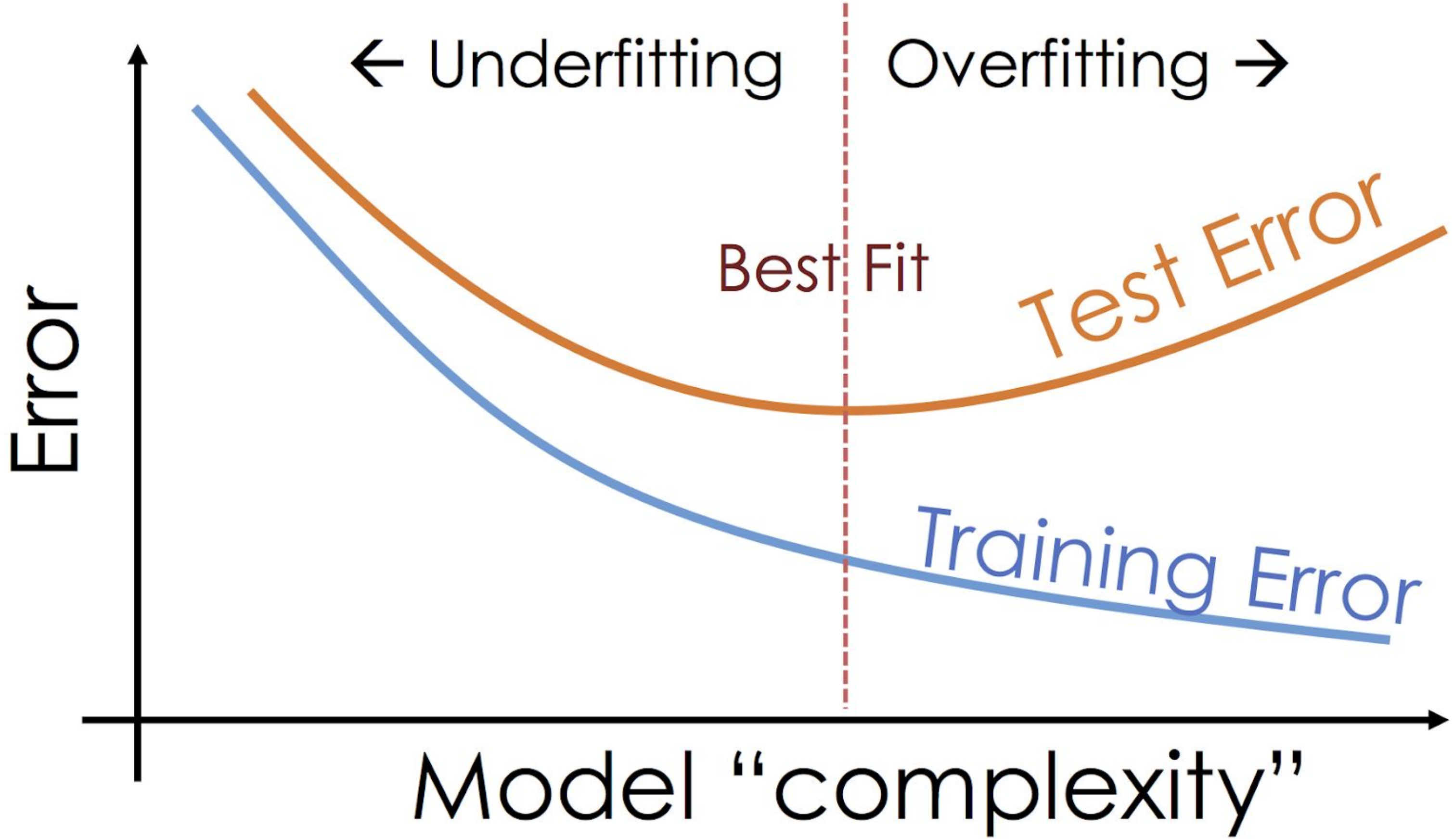
-Train set

- Test set



Evaluating Your Model

Underfitting
Overfitting
Good fitting



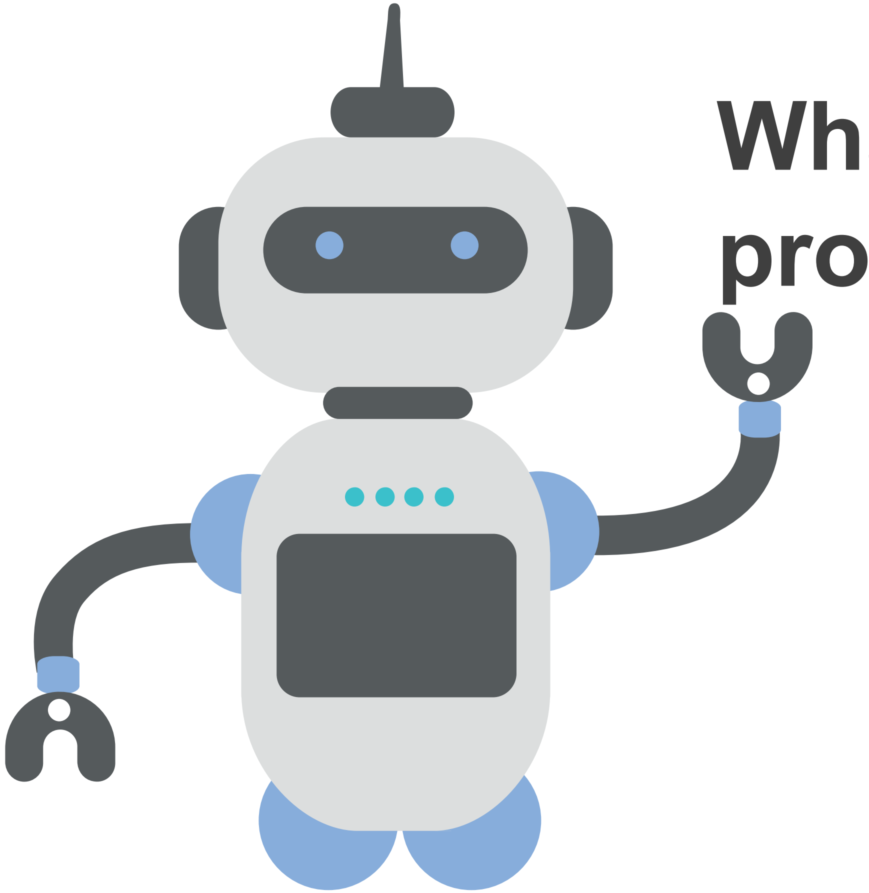
Some Challenges of AI

- Feature-based models need domain-specific expertise
- Data often needs a lot of cleaning and preprocessing (missing data, inconsistent data, etc)
- Handling data from different sources and of various types
- Data is not always free
- Generalization and scalability of machine learning models
- Interpretability of decisions taken by these models
- Data privacy concerns (like in the financial and medical sectors)

“AI is likely to be either the best or worst thing to happen to humanity.”

-- Stephen Hawking

Computer Programming



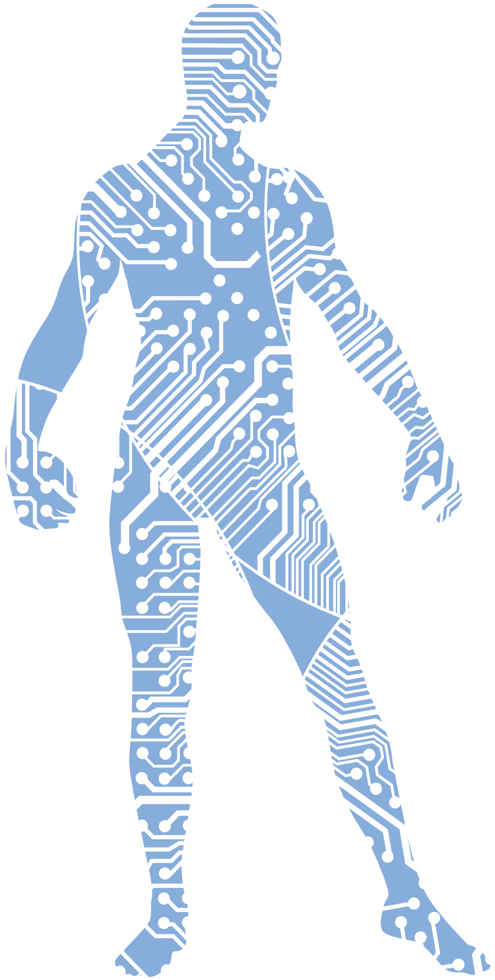
What is computer programming?

Computer Programming

- The process of designing and building a computer program that executes a specific task
- Done through a set of commands that the computer understands

Examples include:

- creating video games
- building websites
- creating animated videos



Programming Languages

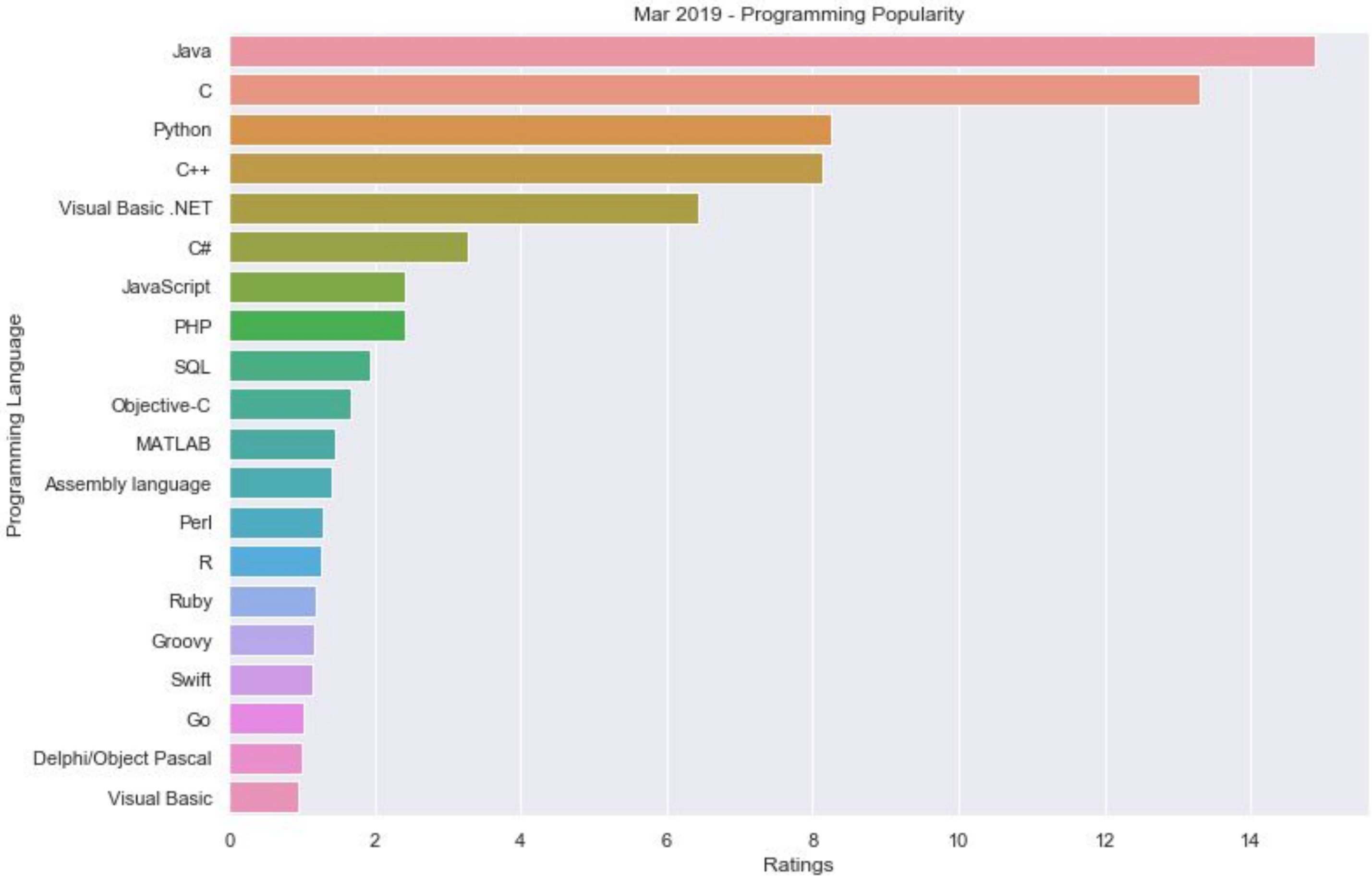
What are programming languages?

- Used to create computer programs
- A set of rules that instruct a computer of operations to perform
- Every programming language has its unique words, symbols, and grammatical rules

Programming Languages

Most popular programming languages:

- Java
- C
- Python
- C++
- …



Intro to Python

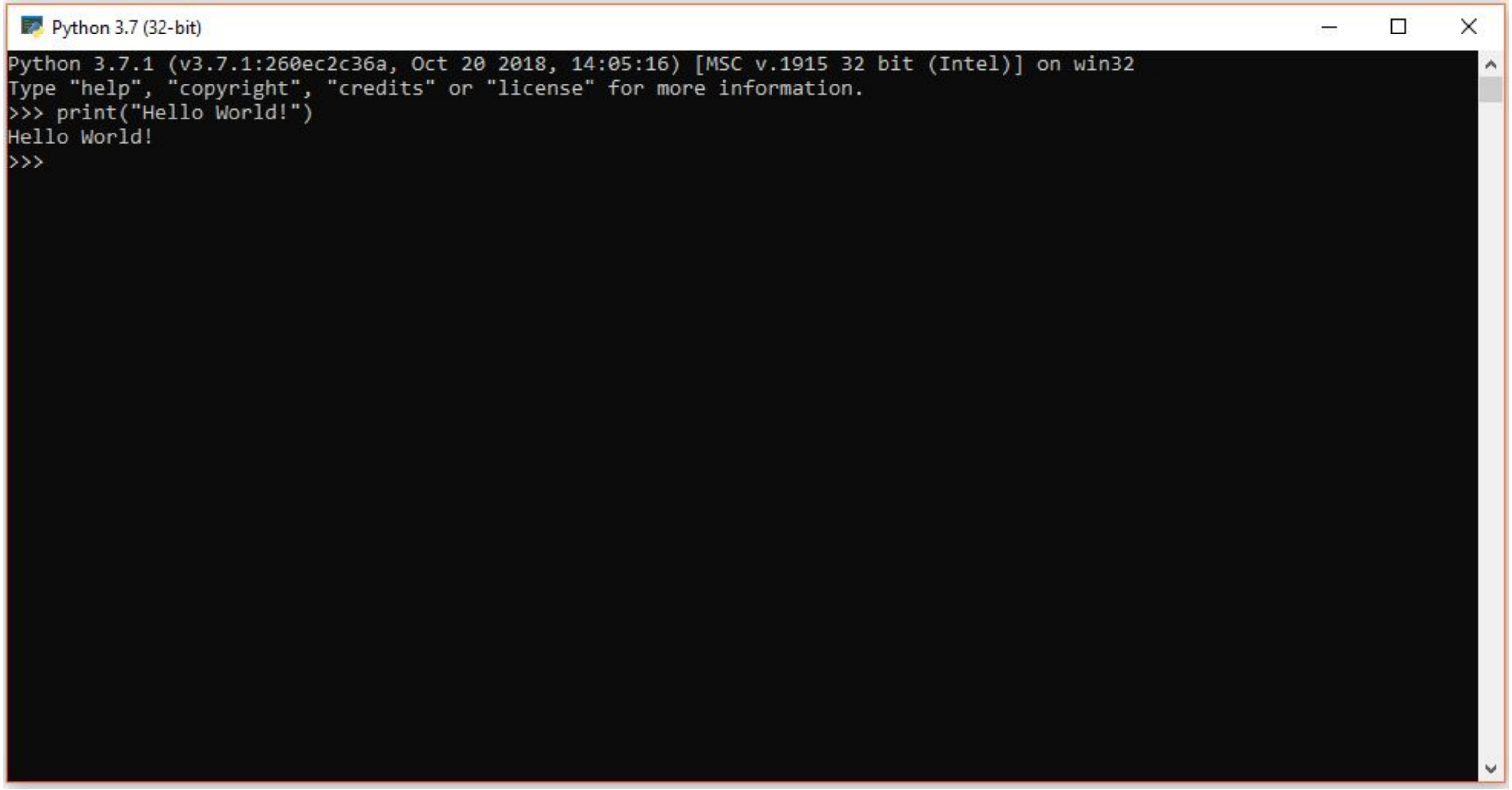
Python is:

- A high-level and a general-purpose programming language
- First released in 1991

Python Environment

Python Shell:

- Where the command and lines of code are entered and directly executed



```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:05:16) [MSC v.1915 32 bit (Intel)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> print("Hello World!")
Hello World!
>>>
```

Python Environment

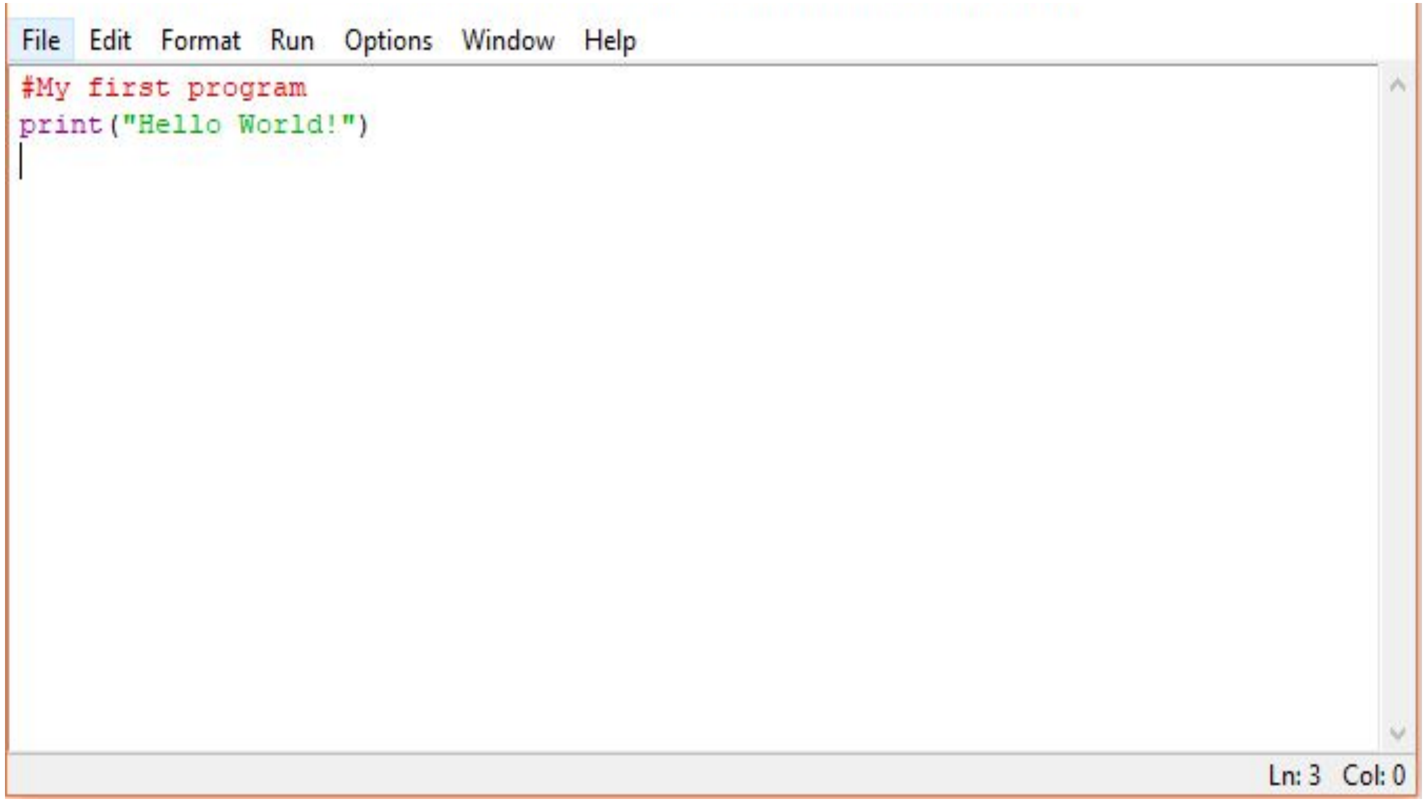
IDE (Integrated Development Environment):

- Similar to text editors except that they are specific to writing and editing code
- Entire code is written to be saved, used, and executed later

Python Environment

Example IDEs:

- Python IDLE:

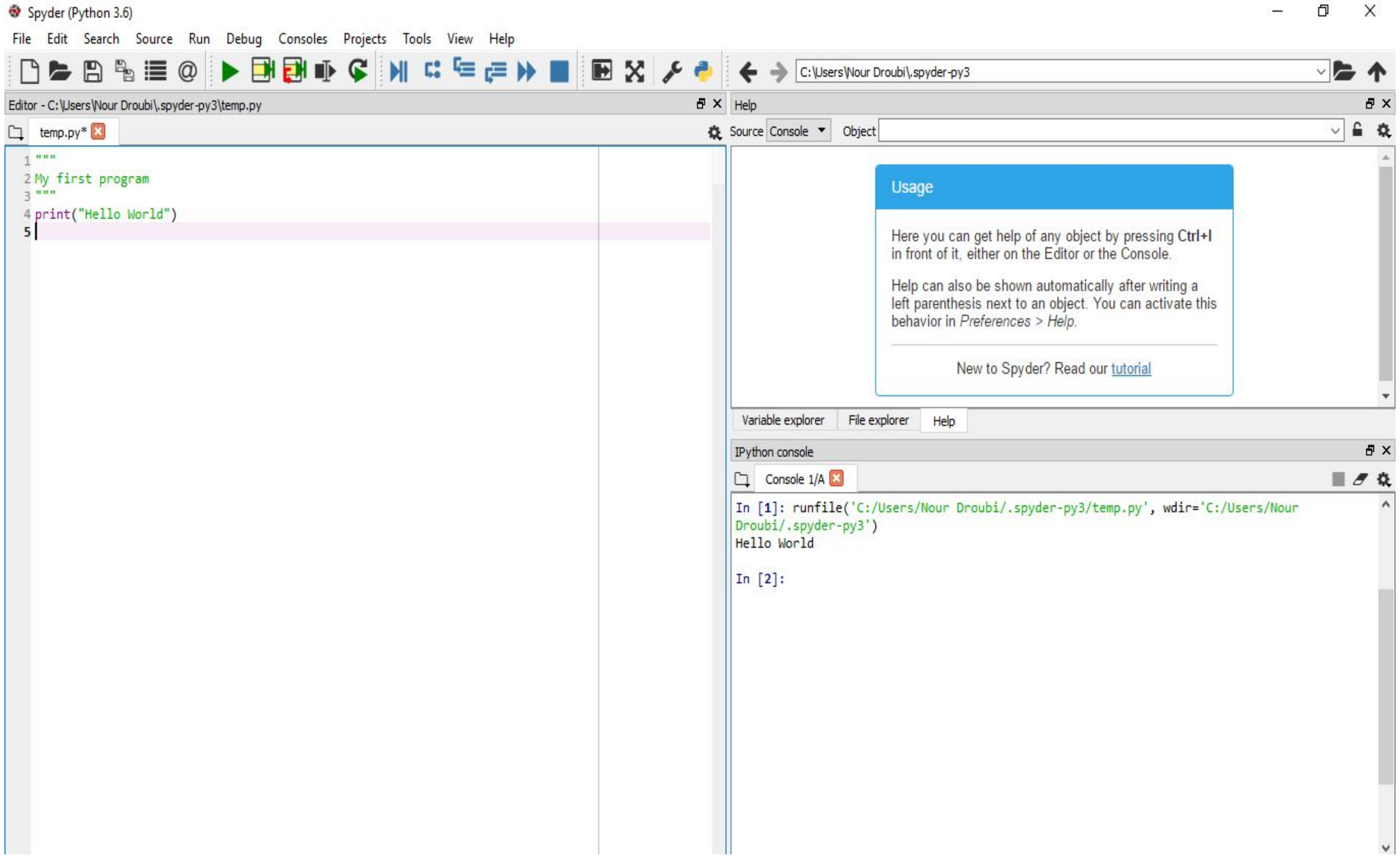


```
#My first program
print("Hello World!")
```

Python Environment

Example IDEs:

- Spyder:



Spyder (Python 3.6)
File Edit Search Source Run Debug Consoles Projects Tools View Help
C:\Users\Nour Droubi\.spyder-py3
Editor - C:\Users\Nour Droubi\.spyder-py3\temp.py
temp.py*
Help
Source Console
Object
Variable explorer File explorer Help
IPython console
Console 1/A

```python
"""
My first program
"""
print("Hello World")

```

Usage

Here you can get help of any object by pressing Ctrl+I in front of it, either on the Editor or the Console.

Help can also be shown automatically after writing a left parenthesis next to an object. You can activate this behavior in *Preferences > Help*.

New to Spyder? Read our tutorial

```
In [1]: runfile('C:/Users/Nour Droubi/.spyder-py3/temp.py', wdir='C:/Users/Nour Droubi/.spyder-py3')
Hello World

In [2]:
```

Python Environment

Cloud Alternatives:

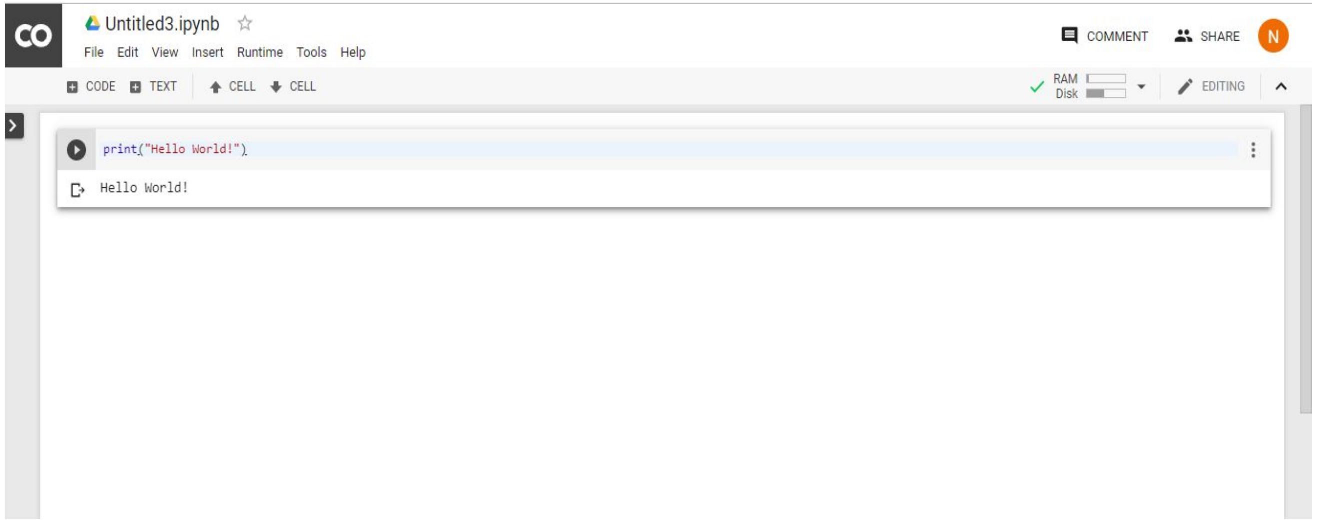
- Run completely online on the cloud
- Require no prior setup

Python Environment

Examples:

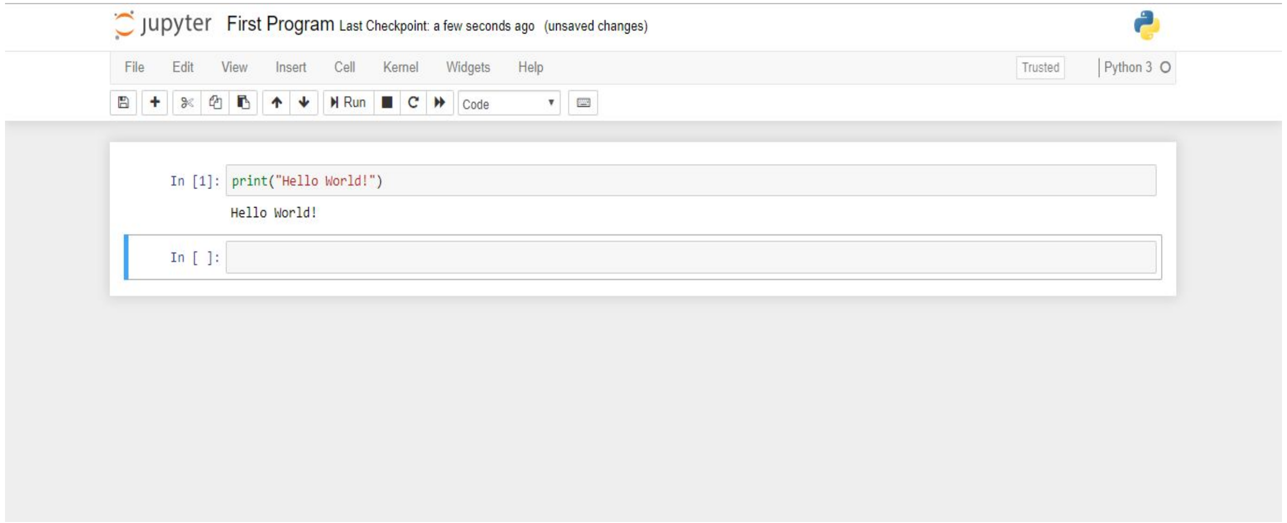
- Google's Colab

- Jupyter Notebook



```
print("Hello World!")
```

```
Hello World!
```



jupyter
First Program Last Checkpoint: a few seconds ago (unsaved changes)
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code

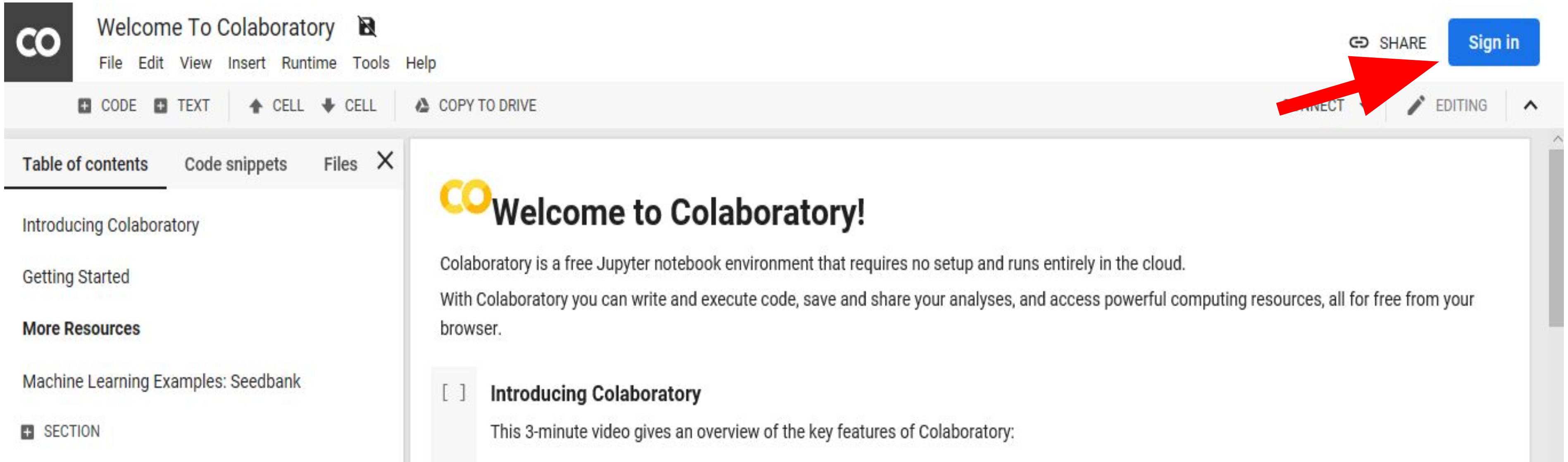
```
In [1]: print("Hello World!")

        Hello World!

In [ ]:
```

Using Google Colab

Steps:

1- Click on the link of the Jupyter Notebook given

2- Sign in with your Gmail account



Welcome to Colaboratory!

Colaboratory is a free Jupyter notebook environment that requires no setup and runs entirely in the cloud.

With Colaboratory you can write and execute code, save and share your analyses, and access powerful computing resources, all for free from your browser.

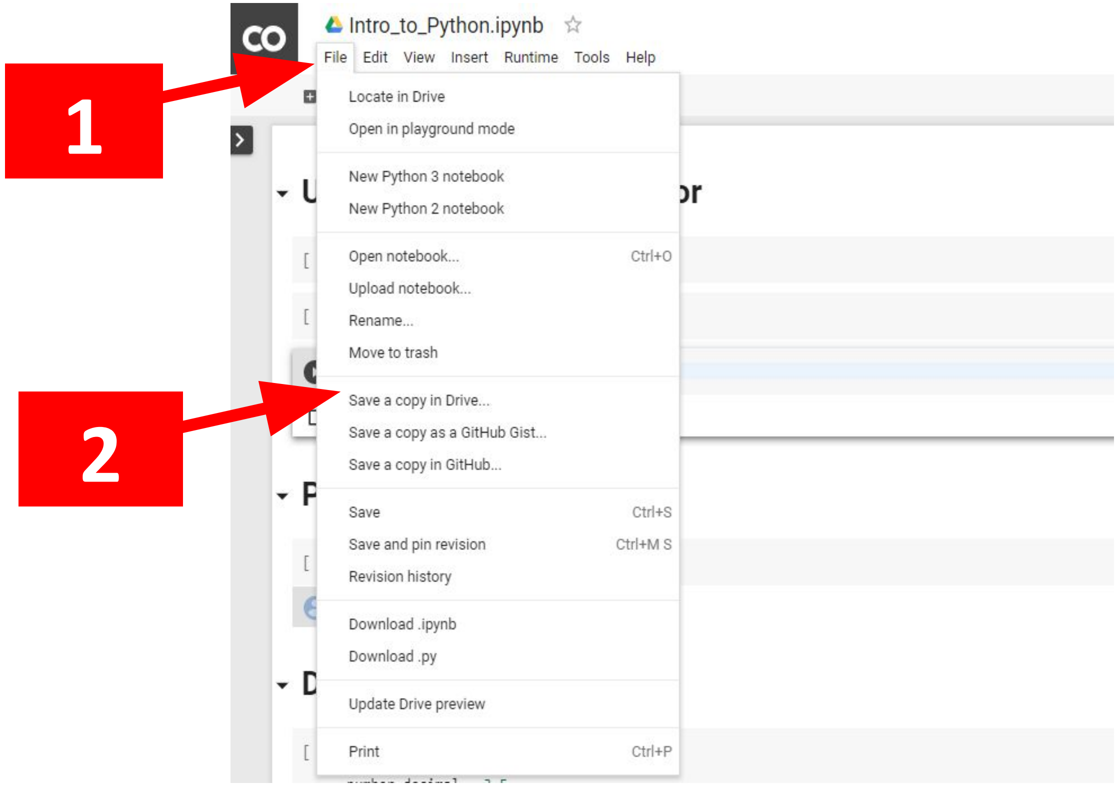
Introducing Colaboratory

This 3-minute video gives an overview of the key features of Colaboratory:

Using Google Colab

Steps:

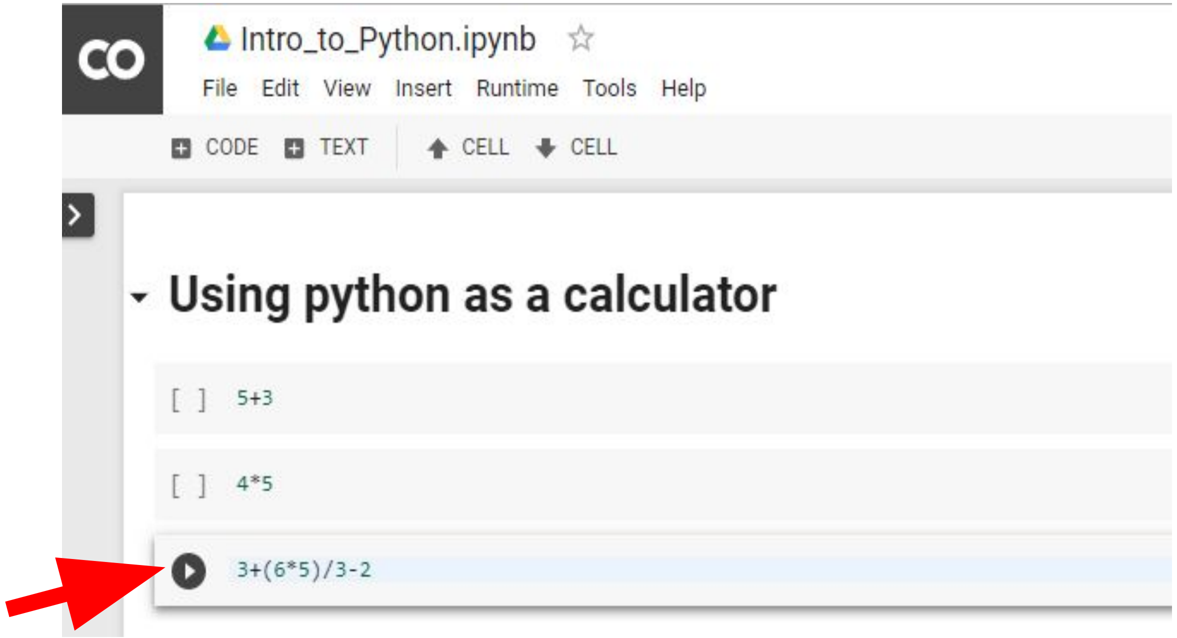
3- File -> Save a copy in Drive



Using Google Colab

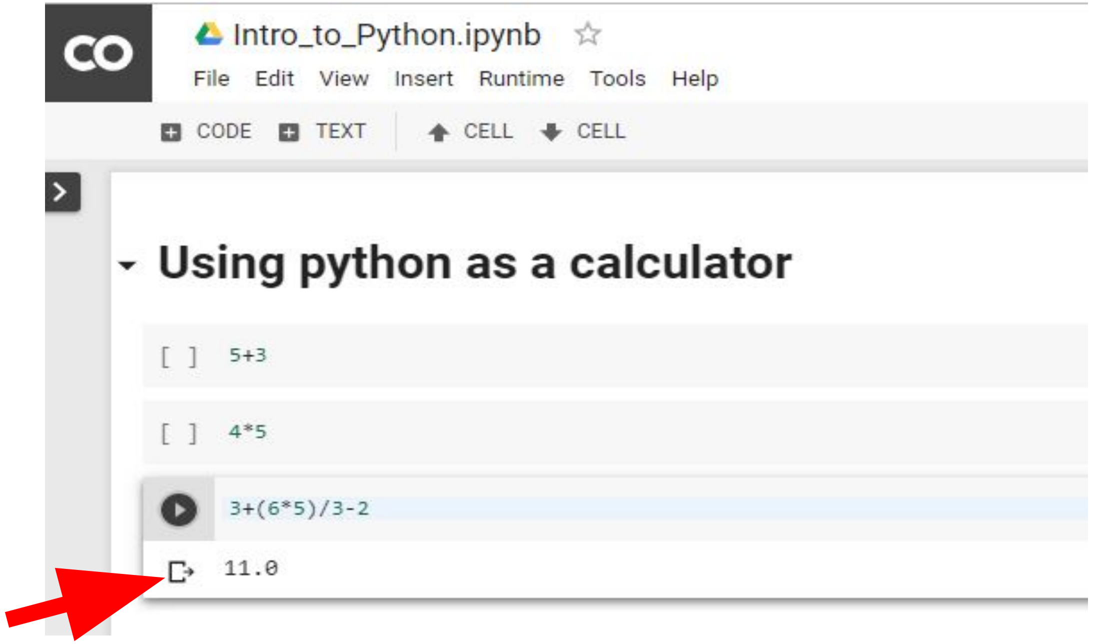
Steps:

3- Run the code



Using python as a calculator

```
[ ]  5+3
[ ]  4*5
     3+(6*5)/3-2
```



Using python as a calculator

```
[ ]  5+3
[ ]  4*5
     3+(6*5)/3-2
     11.0
```

Python Environment

Demos:

- Using python as a calculator

```
>>> 5+3
8
>>> 4*5
20
>>> 3+(6*5)/3-2
11.0
```

Python Environment

Demos:

- Using the print function:

```python
print("Hello World!")
```

```
Hello World!
```

Language Basics

Basic Data types:

- Numbers (signed, unsigned, integers, floating point numbers)
- Characters (a,b,A)
- Boolean (0 or 1, True or False)

Complex Data types:

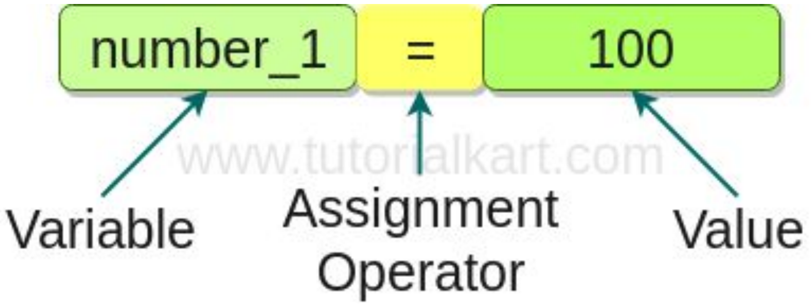
- Strings (combination of characters)

Language Basics

Variables:

Reserved memory locations that store values that can be used again

A variable is created by giving it a name and assigning it to a value using the equals sign (=)



Naming rules:

- First character can’t be a digit
- Can be of any length
- Can contain uppercase and lowercase letters, digits, and underscore (_)

Language Basics

Variables:

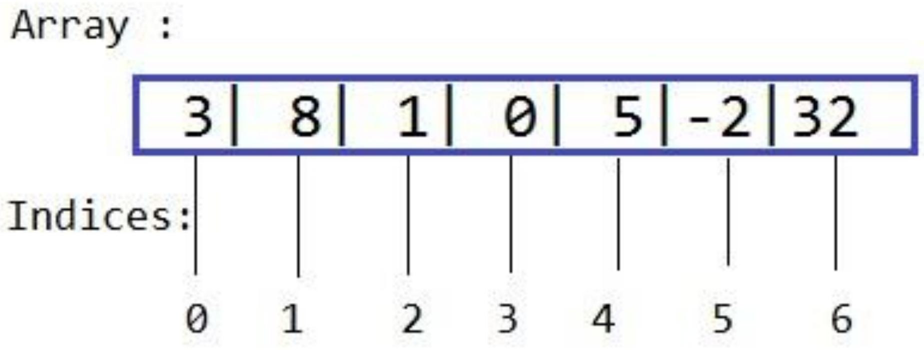
```
string = "This is a string"
number_integer = 4
number_decimal = 3.5

print(string)
print(number_integer)
print(number_decimal)
```

```
This is a string
4
3.5
```

Language Basics

Arrays:

- A type of variable that holds many pieces of information
- Elements are accessed using their index
- First element -> index = 0
- Second element -> index = 1
- …
- Last element can also be accessed through index = -1



Array :

3	8	1	0	5	-2	32

Indices:

0	1	2	3	4	5	6

Language Basics

Arrays:

```
array = [1, 4, "hello", 4.5]

print( array[0] )
print( array[1] )
print( array[2] )
print( array[3] )
```

```
1
4
hello
4.5
```

Language Basics

Arrays:
Slicing

```
array = [1, 4, "hello", 4.5]

print( array[ : 3 ] )
print( array[ 2 : ] )
print( array[ 1 : 3 ] )
print( array[ : ] )
```

```
[1, 4, 'hello’]
['hello', 4.5]
[4, 'hello’]
[1, 4, 'hello', 4.5]
```

Language Basics

Operators:

- Addition (+)
- Subtraction (-)
- Multiplication (*)
- Division (/)
- Assignment operators (=)
- Relational operators:
  - greater than: >
  - greater than or equal: >=
  - less than: <
  - less than or equal: <=
  - equal-to: ==
  - not-equal-to: !=
- Logical operators:
  - And: and
  - Or: or
  - Not: not
  - Boolean: True or False

Language Basics

Comments:

- Lines of codes that start with the symbol #
- Not executed by the program
- Usually used to explain your code to other programmers

```
#This is a comment
```

Language Basics

If/else statements:

- A programming conditional statement

If (condition) → do this
Else → do that

If the condition is satisfied → certain lines of code are executed
Otherwise → other lines of code are executed

Language Basics

If/else statements:

```
x = 0
if (x==0):
    print("True")
else:
    print("False")
```

```
True
```

Language Basics

If/else statements:

```
x = 3
if (x==0):
    print("True")
else:
    print("False")
```

```
False
```

Language Basics

While loops:

A set of statements are executed as long as the condition is True

while (condition) → do this

```
i = 1
while ( i < 6 ):
    print( i )
    i = i + 1
```

```
1
2
3
4
5
```

Language Basics

For loops:

- Used to iterate over a sequence
- A set of statements are executed for every item in the sequence

```
numbers = [1 , 2 , 3]

for x in numbers:
   print(x)
```

```
1
2
3
```

Language Basics

For loops with the range( ) function:

range (starting value, ending value, increment value)

- The range function returns a sequence of numbers
- Starts from 0 by default
- Increments by 1 by default
- Ends at the specified number

```
for x in range(3):
   print(x)
```

```
0
1
2
```

Language Basics

Functions:

- A block of code that only runs when it is called
- Defined using the “def” keyword

```
def my_function():
    print(“Hello World!”)
```

?

Language Basics

Functions:

The code will not run unless the function is called by its name

```
def my_function():
    print(“Hello World!”)

my_function()
```

```
Hello World!
```

Language Basics

Functions:

Function naming rules:

- Start with a letter or underscore only (_)
- Should be lowercase
- Can contain numbers except for the first character
- Shouldn’t be the same as a Python Keyword
- Can have any length but preferred to be short

Language Basics

Functions with parameters:

Take parameters as input that can be used inside the function

```
def my_function(number):
     print (number + 2)

my_function(5)
```

```
7
```

Language Basics

Functions with the return function:

Allows the function to return the result

```
def my_function(number):
    return (number + 2)

my_function(5)
```

?

Language Basics

Functions with the return function:

A print function is required to see the result

```
def my_function(number):
     return (number + 2)

print( my_function(5) )
```

```
7
```

Language Basics

Indentation:

Indentation is very important
The program will not run if the indentation is not correct

```
for x in range(3):
    print(x)
```

✓

```
for x in range(3):
print(x)
```

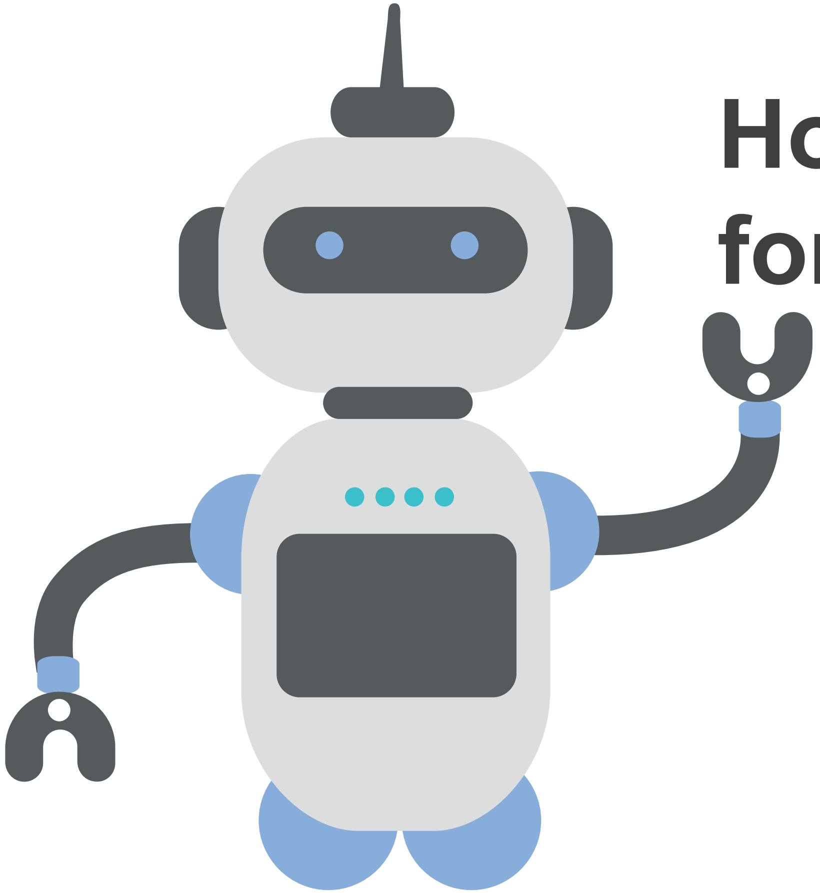
✗

```
IndentationError:
expected an indented
block
```

Language Basics

Types of errors:

- **NameError:** attempt to access an undeclared variable
- **SyntaxError:** python interpreter syntax error
- **IndexError:** request for an out-of-range index for sequence
- **ZeroDivisionError:** division by any numeric zero
- **AttributeError:** attempt to access an unknown object attribute



How can python be used for machine learning?

Python Libraries

To use a library, it should be imported in the code:

```
import library_name
```

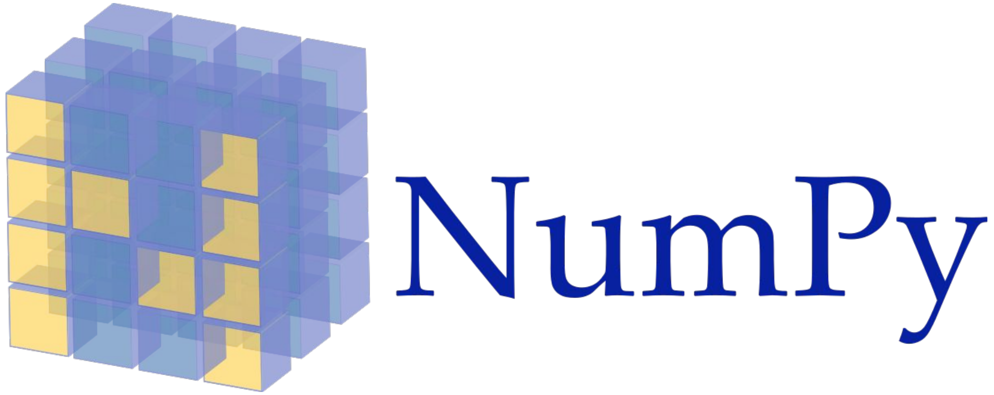
Built-in functions of the library can be accessed using the dot operand

!!

```
library_name.function_name
```

NumPy Library

NumPy Library:



- General-purpose array-processing package
- For Data Analysis
- Provides efficient and high-performance multi-dimensional array object and tools

NumPy Library

Creating NumPy Arrays:

```
import numpy as np
#One-dimensional NumPy array
list1 = [1 , 2 , 3]
numpy_array= np.array( list1 )
print("One-dimensional array: ", numpy_array)

#Two-dimensional NumPy array
list2 = [ [1 , 2 , 3], [4, 5, 6] ]
numpy_array2= np.array( list2 )
print("Two-dimensional array: ", numpy_array2)
```

```
One-dimensional
array:  [1 2 3]
Two-dimensional
array:  [[1 2 3]
         [4 5 6]]
```

NumPy Library

Shape of NumPy Arrays:

```
import numpy as np
#One-dimensional NumPy array
list1 = [1 , 2 , 3]
numpy_array= np.array( list1 )
print("Shape of array 1: ", numpy_array.shape)

#Two-dimensional NumPy array
list2 = [ [1 , 2 , 3], [4, 5, 6] ]
numpy_array2= np.array( list2 )
print("Shape of array 2: ", numpy_array2.shape)
```

```
Shape of array 1:
(3,)
Shape of array 2:
(2, 3)
```

NumPy Library

Reshape function:

Change the dimensions of an array without changing its data

```
import numpy as np
list1 = [1 , 2 , 3 , 4 , 5 , 6]
numpy_array= np.array( list1 )
print( np.reshape ( numpy_array , (3,2) ) )
```

```
[[1 2]
 [3 4]
 [5 6]]
```

SciPy Library

SciPy Library:

- Depends on NumPy
- Has a variety of high-level science and engineering modules together such as:
  - Signal processing tools
  - Optimization tools
  - Linear algebra routines

Pandas Library

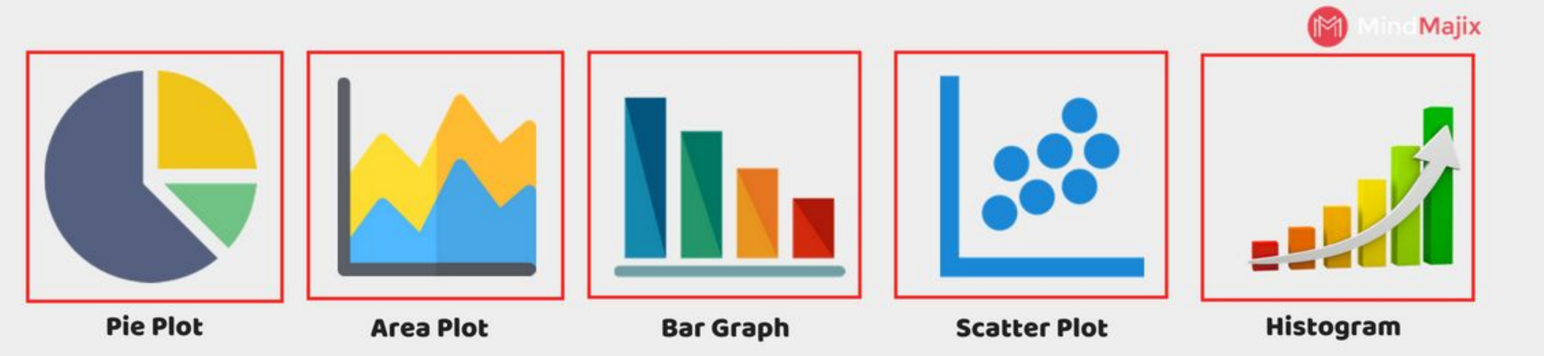
Pandas Library:

- Relational data tool built on top of NumPy
- Used for data analysis
- Main functions:
  - Reading and writing data
  - Data alignment
  - Reshaping
  - Slicing, subsetting
  - Merging and joining of datasets

Matplotlib Library

Matplotlib Library:

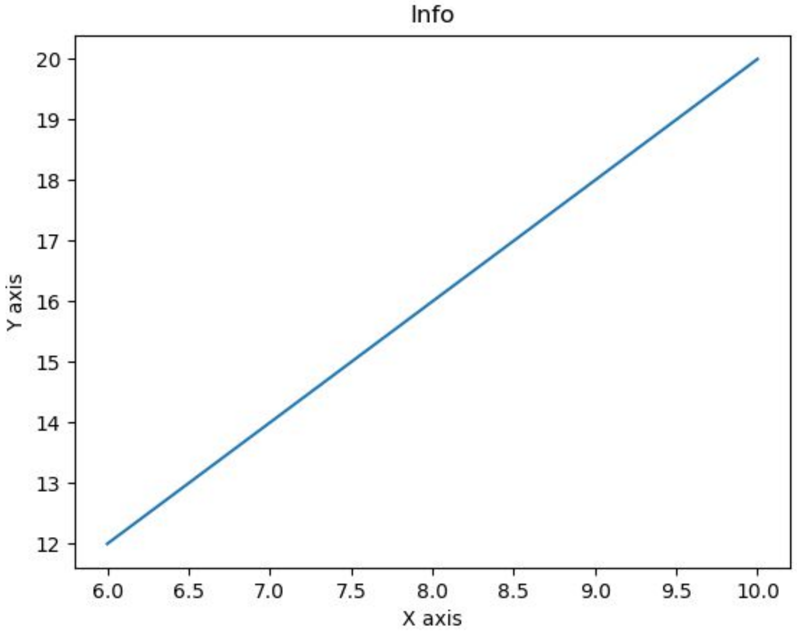
- A 2D and 3D graphic library used to plot scientific figures
- Types of plots:
  - Pie Plot
  - Area Plot
  - Bar graphs
  - Scatter plot
  - Histograms



Matplotlib Library

Plotting a straight line:

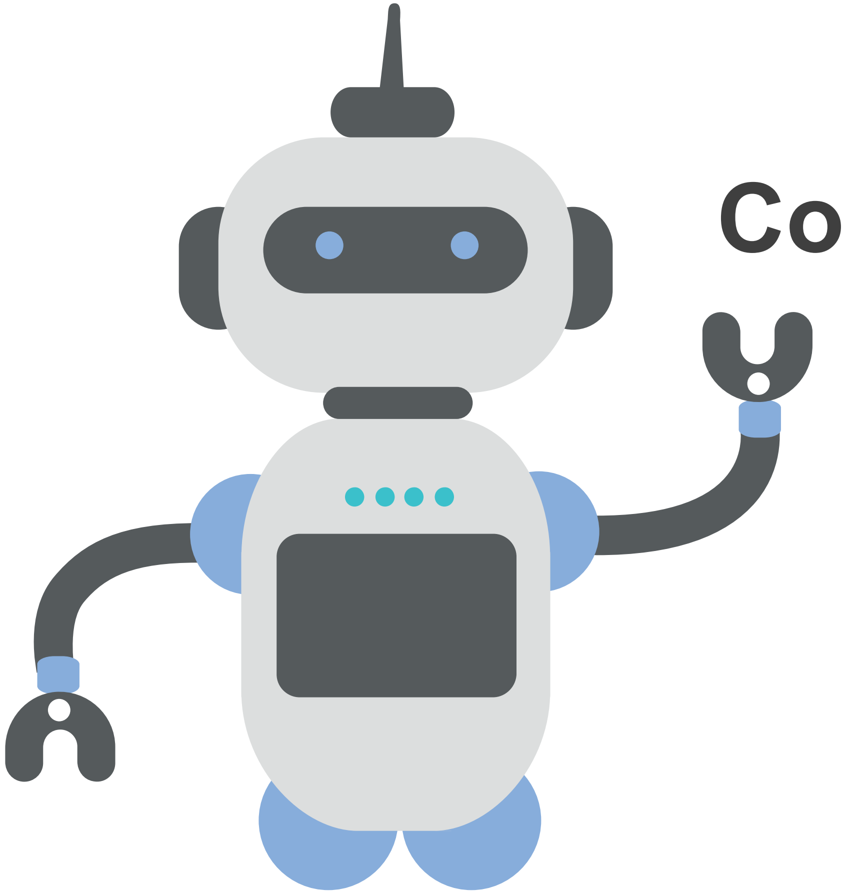
```
from matplotlib import pyplot as plt
x=[6,8,10]
y=[12,16,20]
plt.plot(x,y)
plt.title("Info")
plt.ylabel("Y axis")
plt.xlabel("X axis")
plt.show()
```



Scikit-learn Library

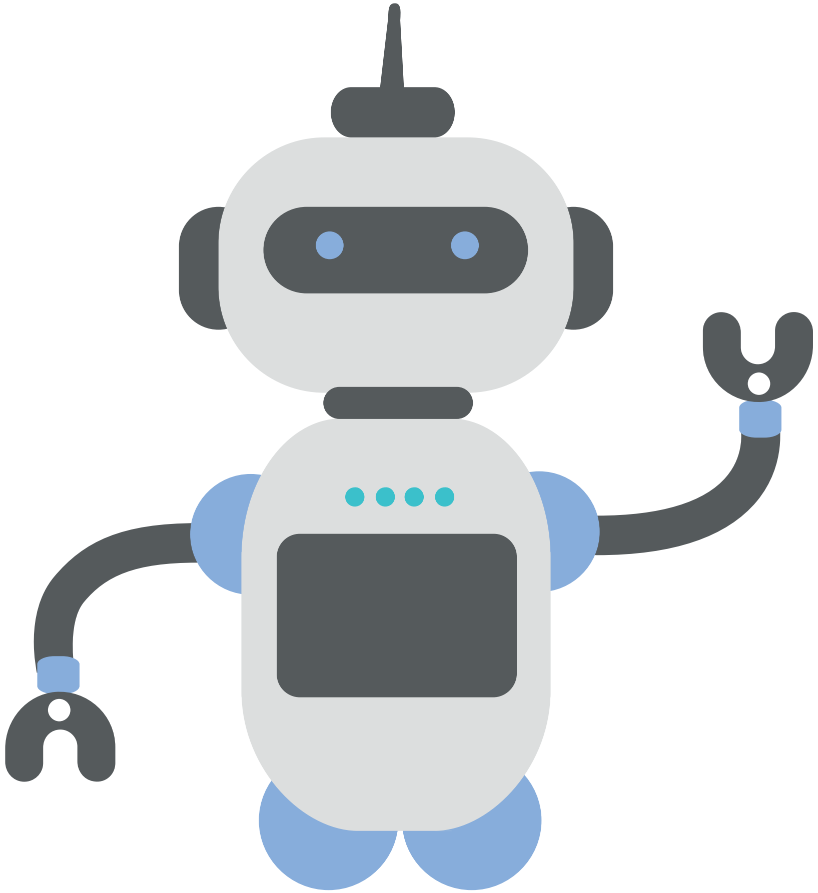
Scikit-learn Library:

- A simple and efficient library for data mining and data analysis
- Main uses:
  - Classification
  - Regression
  - Clustering
  - Preprocessing
  - Model selection



Coding Time!

AI showcase



Demo Time!

- https://teachablemachine.withgoogle.com/
- https://quickdraw.withgoogle.com/
- https://research.google.com/semantris/
- http://fontmap.ideo.com/
- https://thing-translator.appspot.com/

Thank You

See you again TOMORROW!