



# CSC 462: Machine Learning

Introduction to Machine Learning (ML)

Dr. Sultan Alfarhood

Fall 2024

# Which face is real?



[www.whichfaceisreal.com](http://www.whichfaceisreal.com)

# Introduction

# AI will...



\* According to a survey by University of Oxford and Yale University

# Autonomous Trucks



# Autonomous Cars



# Self-checkout Stores



# What about now?



\* According to a survey by University of Oxford and Yale University

# ChatGPT's Opinion

AB

Will artificial intelligence overtake humans in the future?



As an AI language model, I can say that the development of artificial intelligence has the potential to greatly enhance and augment human capabilities in many fields. However, it is unlikely that AI will completely overtake humans in the foreseeable future.



While AI can excel in specific tasks and perform them more efficiently than humans, it still lacks the general intelligence and versatility that humans possess. AI also lacks the ability to experience emotions, intuition, and creativity in the way that humans do.

Moreover, the development of AI is heavily influenced by ethical considerations and societal concerns, which will limit its development in certain areas. The goal of AI research is to create machines that can work alongside humans and assist them in various tasks, rather than replace them entirely.

Therefore, while AI will continue to advance and become more sophisticated, it is unlikely that it will completely surpass human capabilities in the near future.

# AI in Saudi Arabia

**"41% of existing work activities in KSA are theoretically automatable today"**

\* According to McKinsey & Company

# What is AI?

- **Artificial intelligence (AI)** is the study of the agents that **observe** the world that surrounds them, form **plans** and **make decisions** to achieve their objectives
- In simple words, we can say that the goal of AI is to **perform tasks** that require **human intelligence**
- Many fields are under the umbrella of AI
  - Machine Learning (ML)
  - Computer Vision (CV)
  - Natural Language Processing (NLP)
  - Robotics

# AI Techniques

## Logical and rule-based approaches

- Represent systems using **logical rules**
- Rules are designed from top to bottom for the computer
- Systems take decision based on those rules

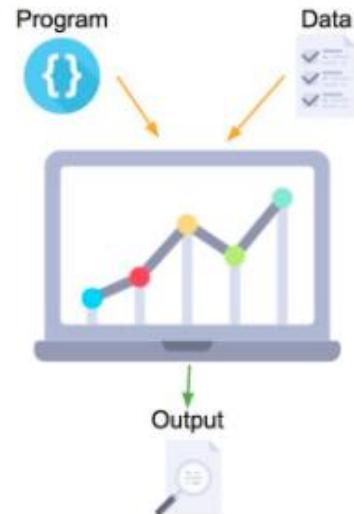
## Machine learning (ML)

- Algorithms find **patterns** in the data and **infer rules** by themselves
- Learn from data and **improve over time**
- These patterns can be used for automation or prediction
- ML is the most dominant approach of AI today

# Machine Learning

- The goal of machine learning is to allow computers to **learn on their own**
- ML is the field of study that gives "computers the ability to learn without being explicitly programmed"
  - Arthur Samuel (1959)
- Arthur Samuel, is credited for coining the term, "Machine Learning"

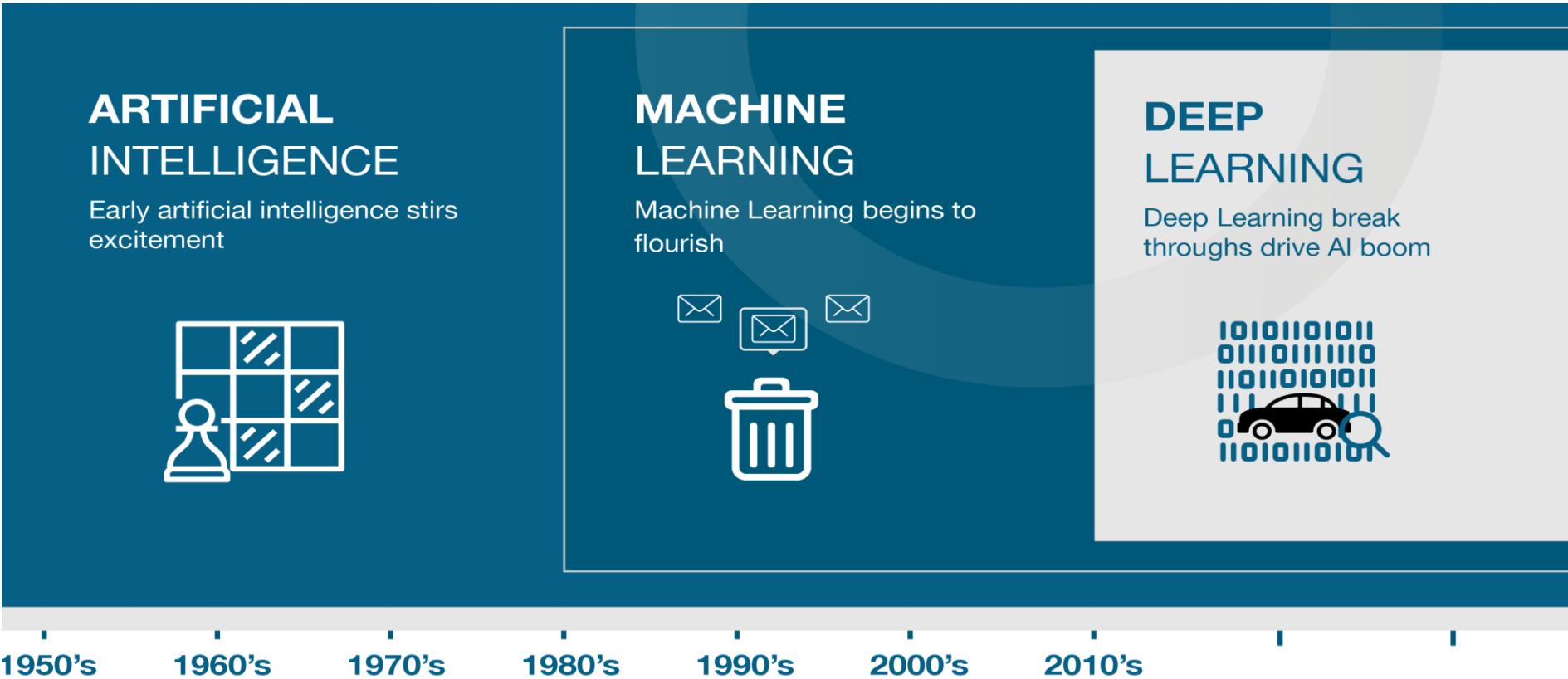
Traditional Programming



Machine Learning



# AI Timeline





The history of machine learning is a long and complex one, with many significant milestones along the way. Here are some of the most influential ones:

1. 1943 - McCulloch and Pitts introduce the first mathematical model of a neuron, which later forms the basis for artificial neural networks.
2. 1951 - Marvin Minsky and Dean Edmonds create the first neural network computer, called the SNARC.
3. 1957 - Frank Rosenblatt develops the Perceptron algorithm, which is the first successful algorithm for machine learning.
4. 1967 - The nearest neighbor algorithm is introduced, which is still widely used in machine learning today.
5. 1970s - Decision trees are developed, which are a popular technique for classification and prediction.
6. 1980s - Backpropagation is developed, which is a method for training artificial neural networks.
7. 1990s - Support vector machines are developed, which are a powerful tool for classification and prediction.
8. 1997 - IBM's Deep Blue defeats world chess champion Garry Kasparov, marking a significant milestone in the field of artificial intelligence.
9. 2006 - Geoffrey Hinton and his team develop deep learning algorithms that can learn features from large amounts of data, leading to breakthroughs in computer vision and speech recognition.
10. 2011 - The development of the first large-scale deep neural network for image recognition, known as AlexNet, which greatly improved the accuracy of image classification tasks.
11. 2012 - Google's self-driving car completes its first successful test drive, marking a major milestone in the development of autonomous vehicles.
12. 2015 - AlphaGo, a deep learning algorithm developed by Google, defeats world champion Lee Sedol at the ancient Chinese game of Go, demonstrating the power of deep learning in complex decision-making tasks.

These milestones and many others have contributed to the growth and development of machine learning, and continue to shape the field today.

# Machine Learning Stages

- Stage 1 (Learning)
  - A model needs to be learned by presenting data to it and training it
- Stage 2 (Prediction)
  - A "production" phase
  - New data is presented to the model
  - Produces responses
    - based on prior knowledge (stage 1)

Learning



Prediction

# Machine Learning Significance

Develop systems that can **adapt automatically** to users

- Personalized content recommendation

Discover **new knowledge** and insights from big data sources

- Market basket analysis

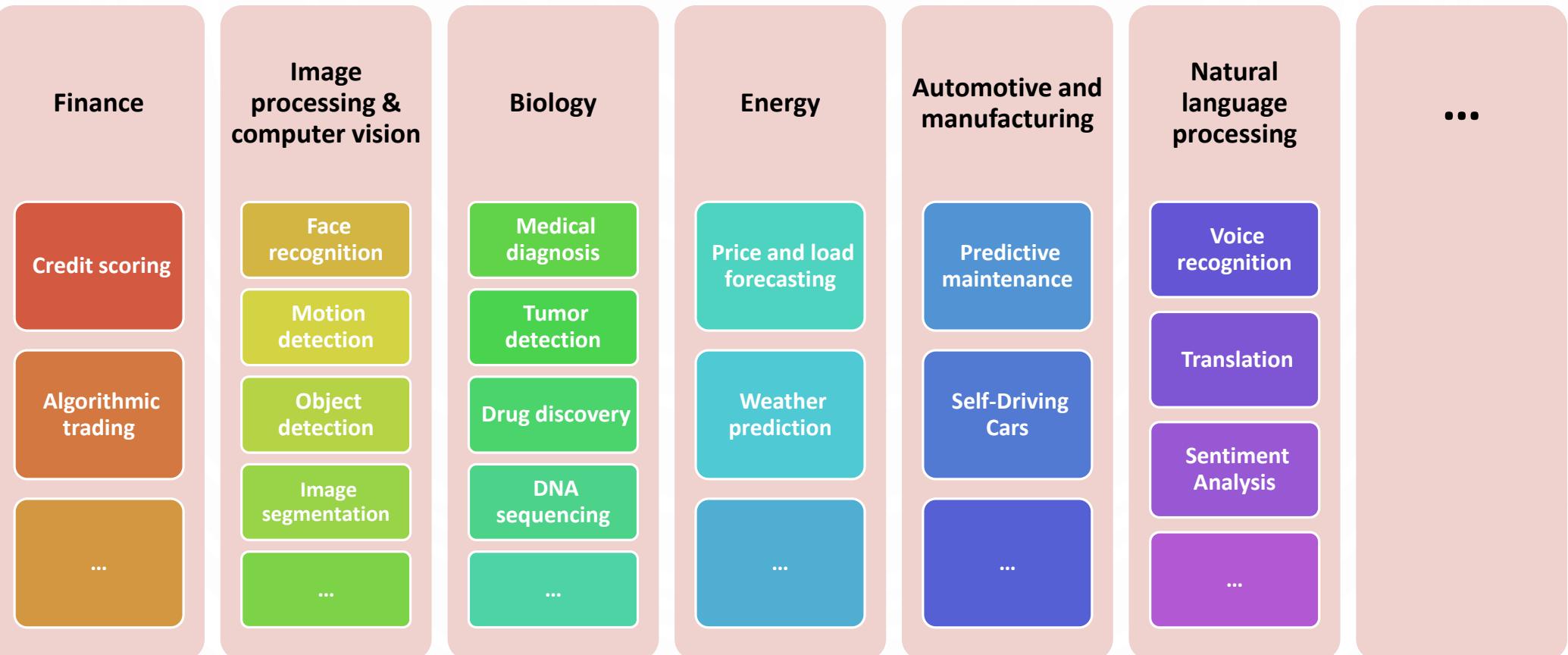
**Automate** tasks that require some intelligence

- Handwritten text recognition

Develop systems that are too difficult or expensive to build manually

- Self-driving car

# Machine Learning in Practice

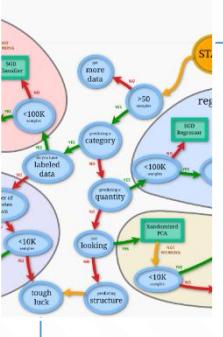


# Why Now?



## Big data availability

- Larger datasets
- Easier data collection and storage



## Increasing development in available models and algorithms

- New and improved models



## Increasing computing power

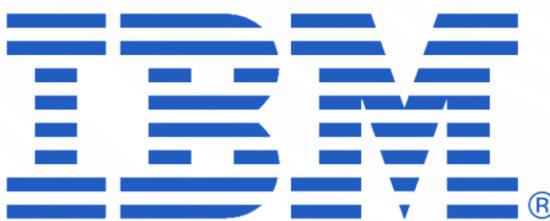
- Graphics Processing Units (GPUs) blowout
- Parallel computing availability



## Increasing support from industry

- Open-source frameworks

# Top AI Companies

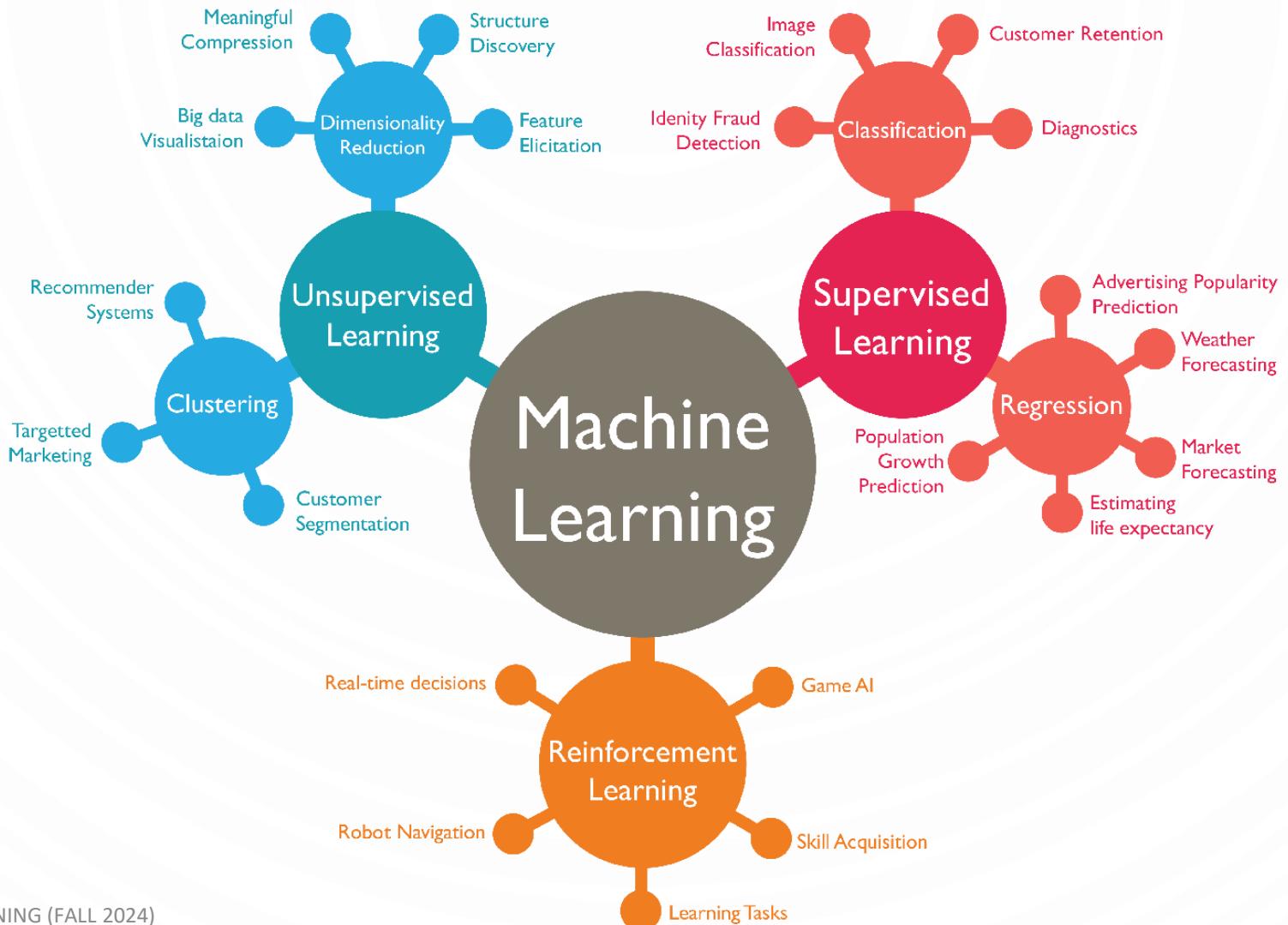


Google

Meta

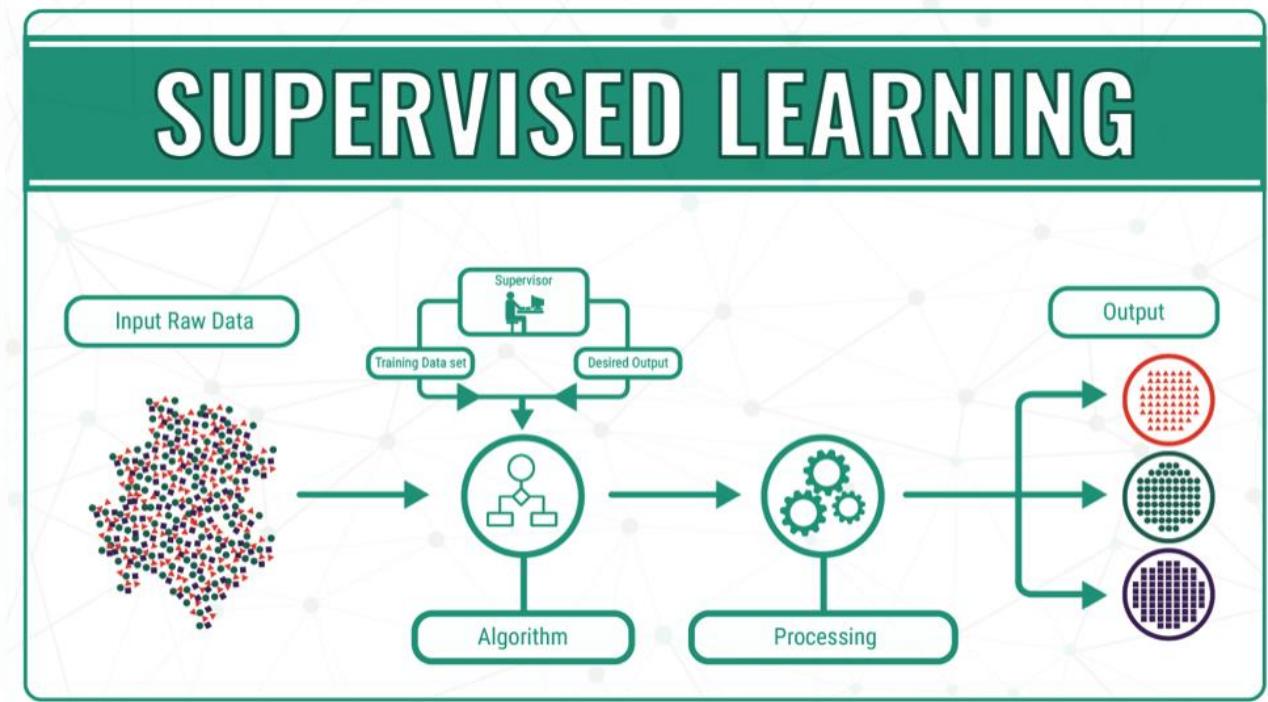
OpenAI

# Machine Learning Approaches



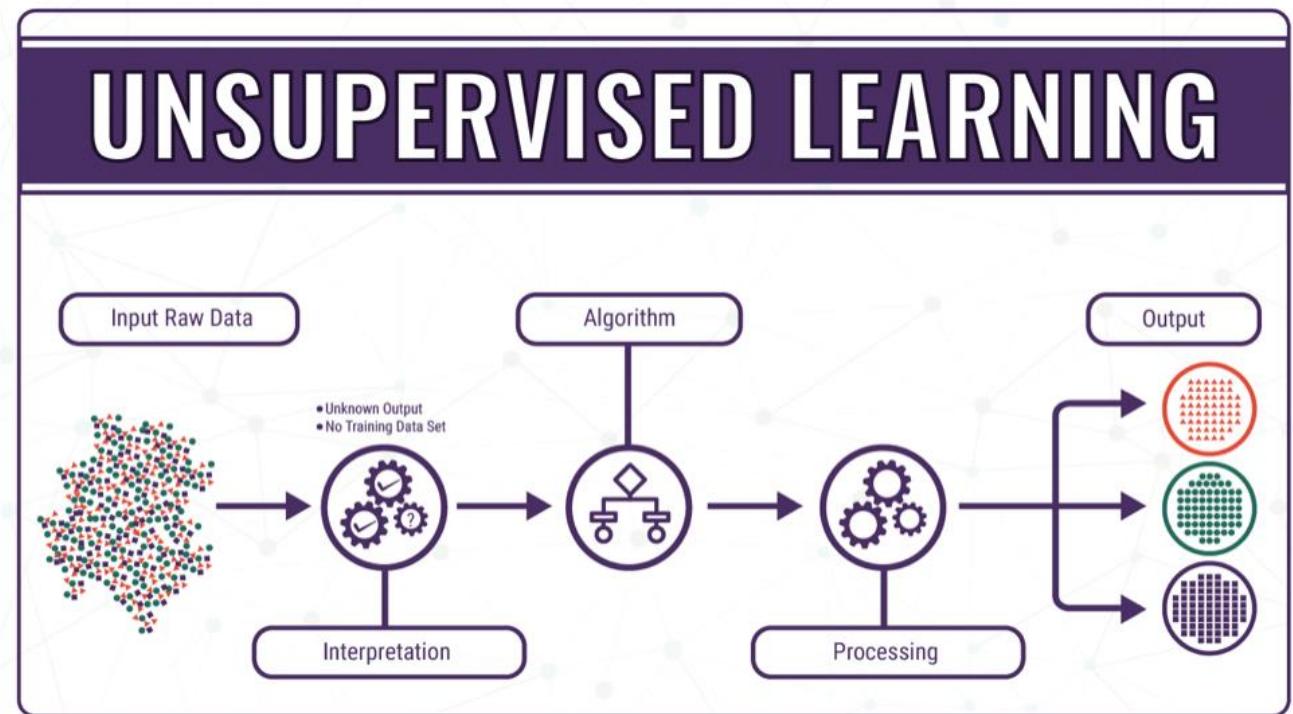
# Supervised Learning

- Supervised learning trains algorithms based on examples of **human-labeled** input and output data
- Common applications
  - Classification
  - Regression



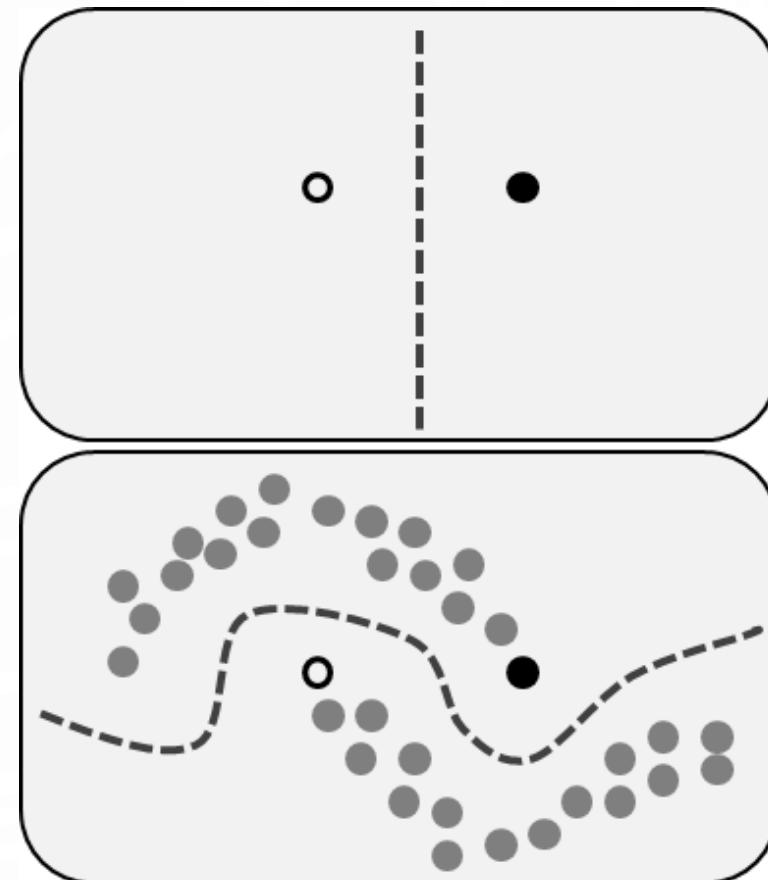
# Unsupervised Learning

- Unsupervised learning provides the algorithm with **unlabeled data** allowing it to find structures in it
- Common applications
  - Clustering
  - Dimensionality reduction



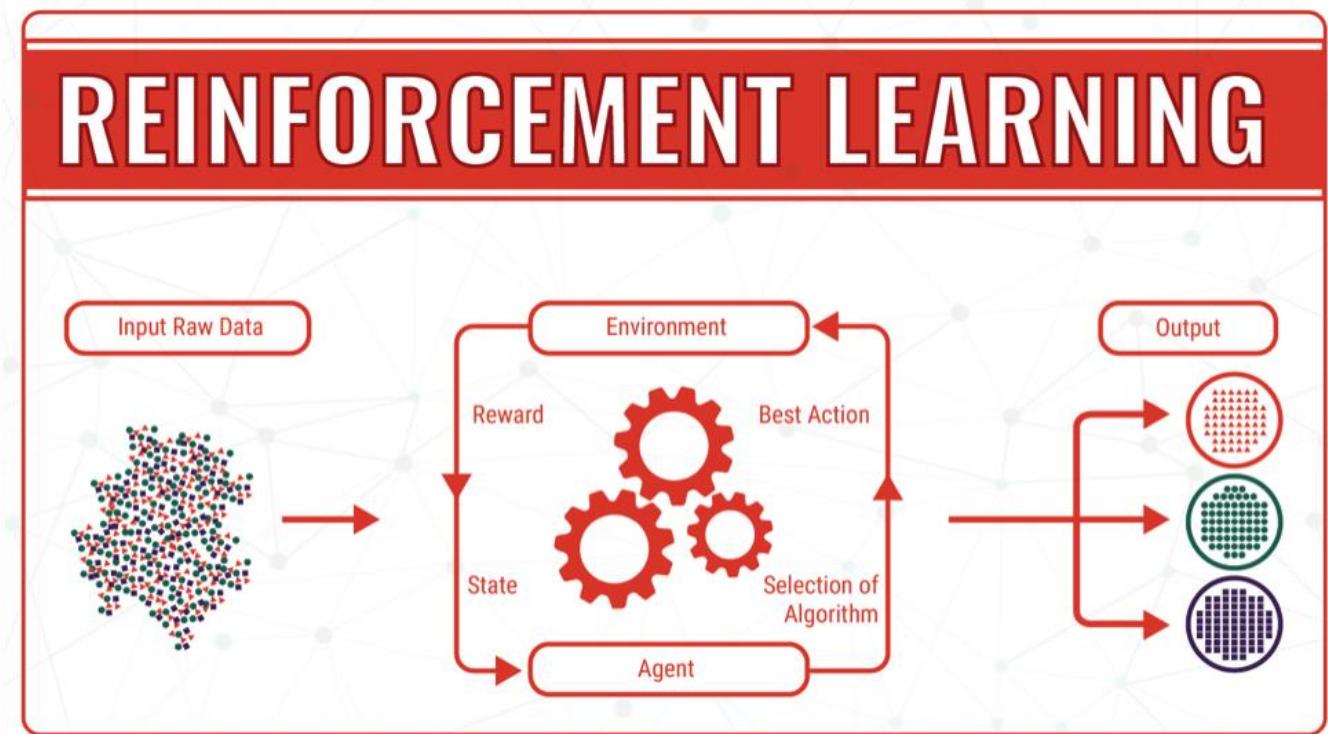
# Semi-Supervised Learning

- An approach that combines a small amount of labeled data with a large amount of unlabeled data during training
- Semi-supervised learning falls between unsupervised learning (with no labeled training data) and supervised learning (with only labeled training data)

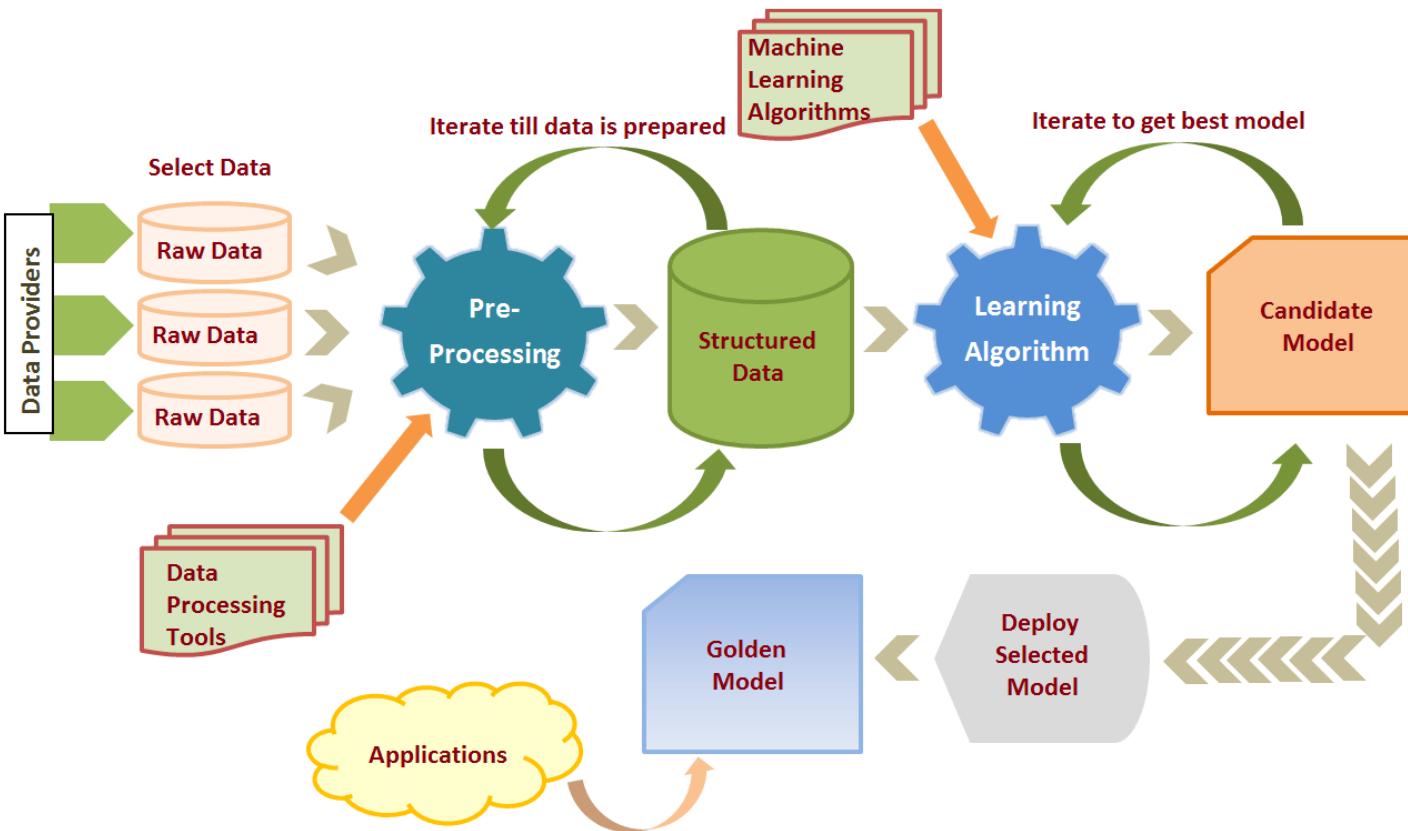


# Reinforcement Learning

- Enables an agent to learn in an interactive environment through **trial and error** using the feedback of its own actions and experiences
- Google's AlphaGo is a famous example of a reinforcement learning approach



# Machine Learning Process



# Iris Flower Classification Example

- Create a model that can classify the different species of the Iris flower
- Types of irises
  - Setosa
  - Versicolor
  - Virginica



# Iris Flower Dataset

- Small dataset from 1936
- This data set consists of 3 different types of irises
  - Setosa
  - Versicolor
  - Virginica
- Each row represents an iris flower
  - 150 rows (50 samples from each type)
- Each column contains
  - Sepal Length
  - Sepal Width
  - Petal Length
  - Petal Width



Iris Versicolor

Iris Setosa

Iris Virginica

Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
...	...	...	...	...

# Terminology

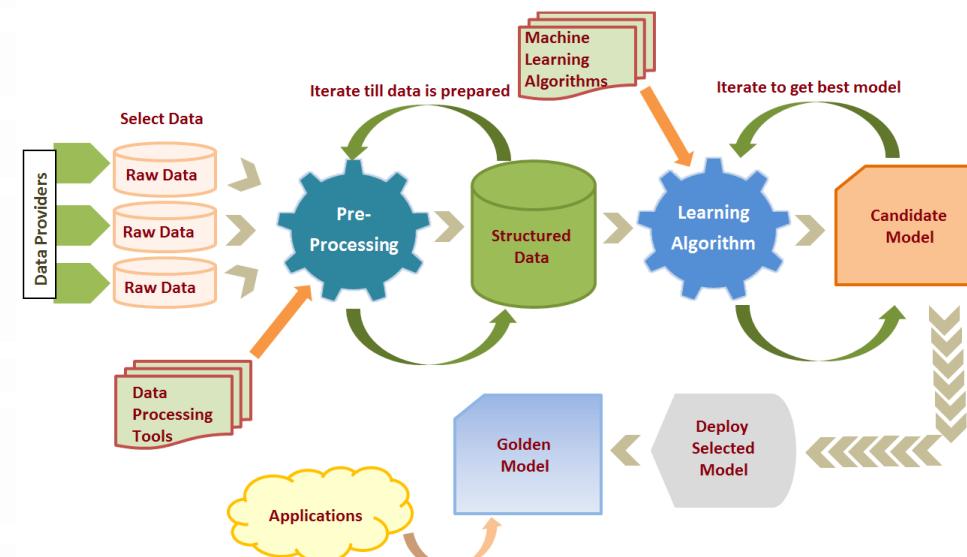


The diagram shows a table representing the Iris dataset. The columns are labeled 'Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width', and 'Species'. The rows contain data for four Iris-setosa samples, followed by an ellipsis. A bracket on the left side of the table groups the first four rows and is labeled 'Instance', 'Sample', 'Example', 'Observation', and 'Row' from top to bottom. Arrows point from each of these five labels to their corresponding row in the table.

Sepal Length	Sepal Width	Petal Length	Petal Width	Species
5.1	3.5	1.4	0.2	Iris-setosa
4.9	3.0	1.4	0.2	Iris-setosa
4.7	3.2	1.3	0.2	Iris-setosa
4.6	3.1	1.5	0.2	Iris-setosa
...	...	...	...	...

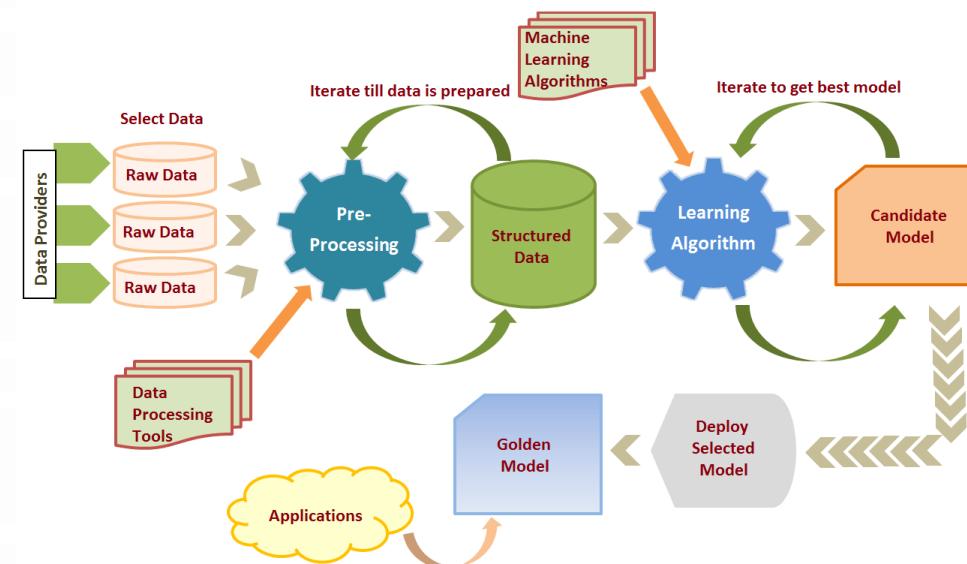
# Data Collection

- The first step in the machine learning process is to collect the data in an appropriate electronic format
- Data includes:
  - Numbers
  - Text
  - Images
  - Videos
  - Audio



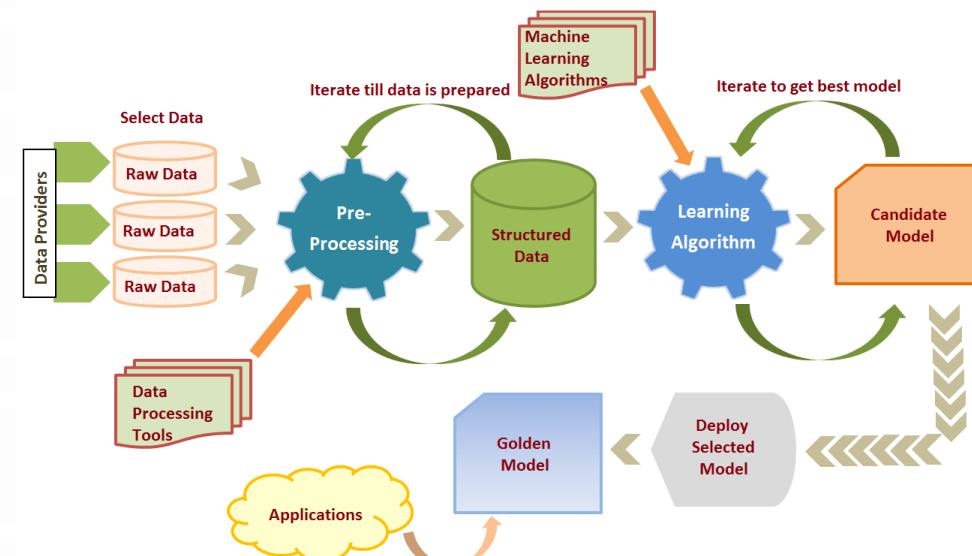
# Data Preparation

- The **quality** of data is one a critical factor in producing quality results
- In this step, the data must be preprocessed to produce **clean data**:
  - Removing **duplicates**
  - Handling **missing** values
  - Correcting **errors**
  - Normalization and **Scaling**
  - **Data type** conversions
  - **Randomizing** data (removes the effects of the particular order)
- This step also includes splitting the data into
  - Training sets
  - Evaluation sets



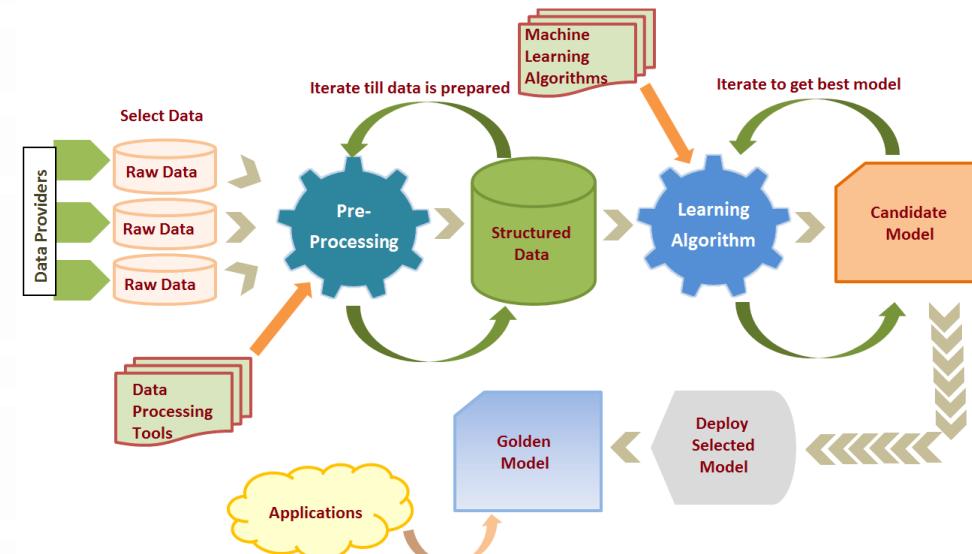
# Model Selection

- This step includes selecting the **appropriate** model (algorithm) for the task



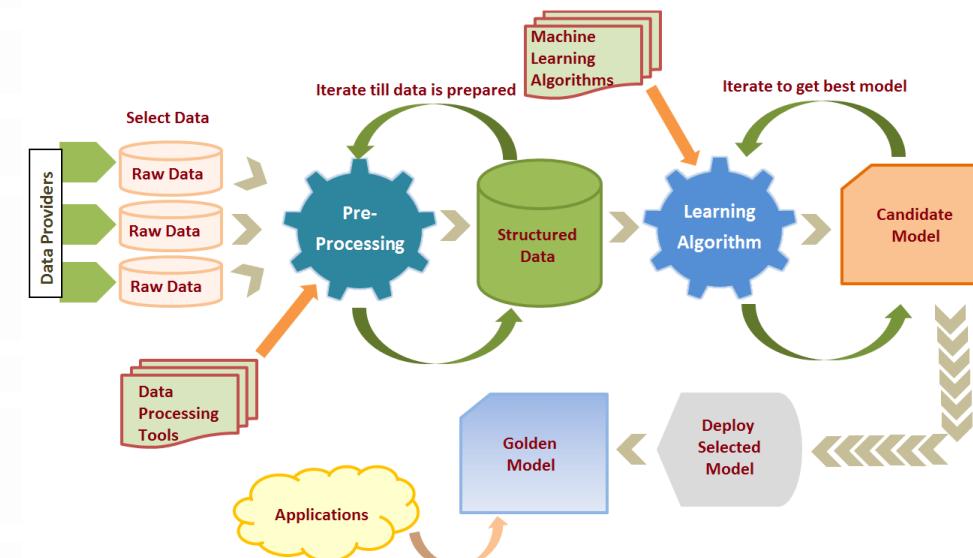
# Model Training

- The process of **building** the model is called training
- This step **fits** (trains) the model to the training data to produce the **candidate model**



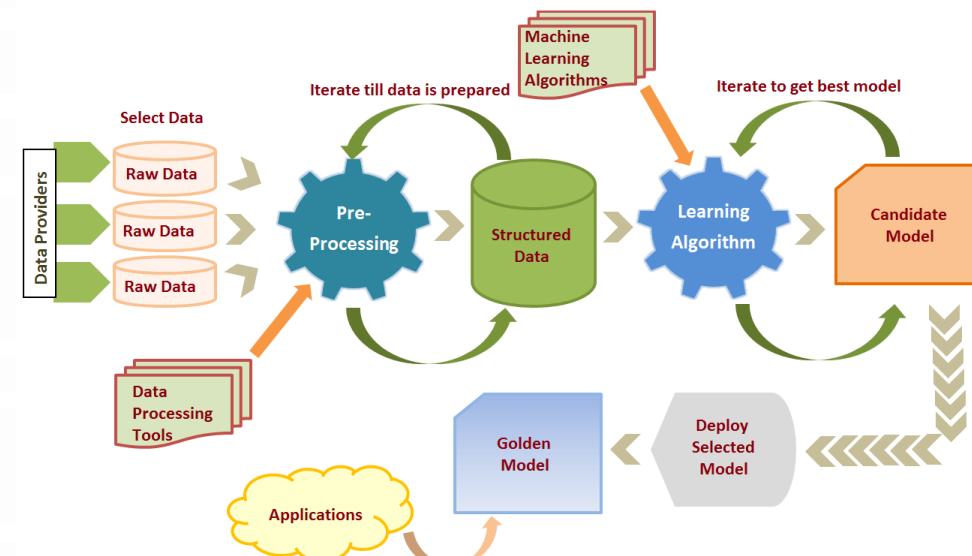
# Model Evaluation

- It is essential to **evaluate** how well the algorithm has learned from its experience
- How to differentiates one model from another?
  - **Accuracy** of the model
    - The ratio of examples whose labels are predicted correctly
  - **Speed** of model building
    - In many practical cases, you would prefer a learning algorithm that builds a less accurate model fast
  - **Prediction processing time**
    - Additionally, you might prefer a less accurate model that is much quicker at making predictions



# Model Evaluation

- Evaluation Techniques
  - Percentage Split
  - Cross-validation



# Evaluation Techniques

- Percentage Split
  - Randomly split the dataset into a training and a testing partitions
  - Give a quick estimate of performance
  - Preferable only **large datasets**
- Cross-validation
  - Split the dataset into **k-partitions** (folds)
  - Train a model on all of the partitions except one that is held out as the test set
  - Then, repeat this process creating k-different models and give each fold a chance of being held out as the test set
  - Then, calculate the average performance of all k models.

# Train, Validation and Test Sets

- **Training Dataset**

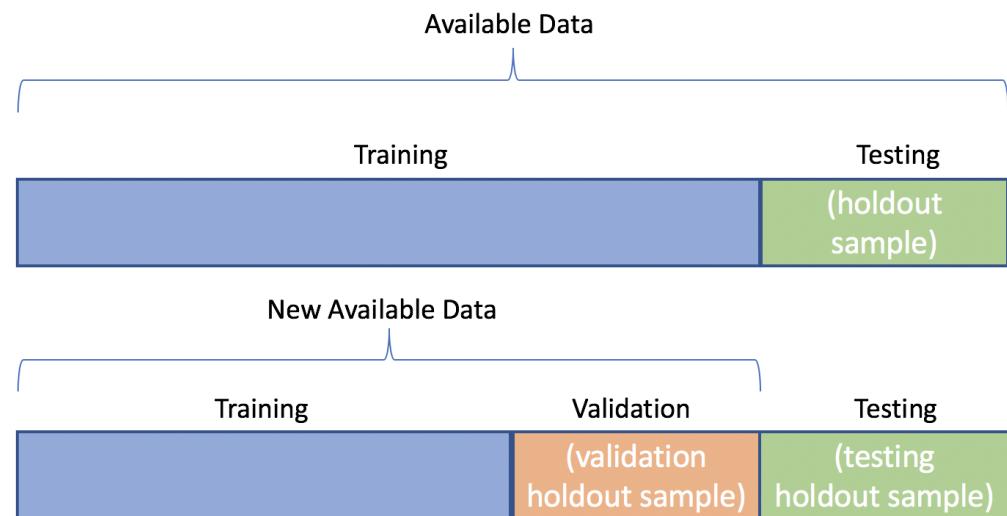
- The sample of data used to fit the model.

- **Validation Dataset**

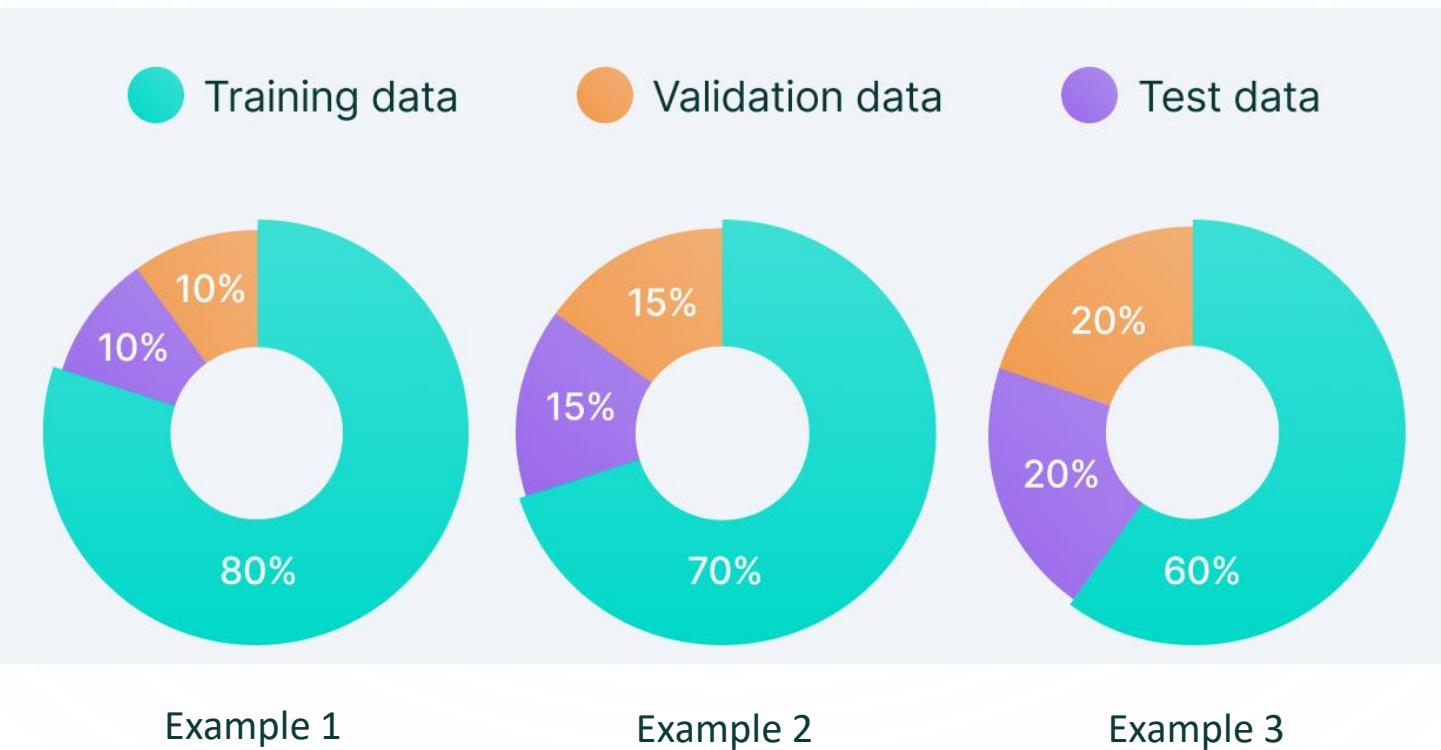
- The sample of data used to provide an unbiased evaluation of a model fit on the training dataset while tuning model hyperparameters.

- **Test Dataset**

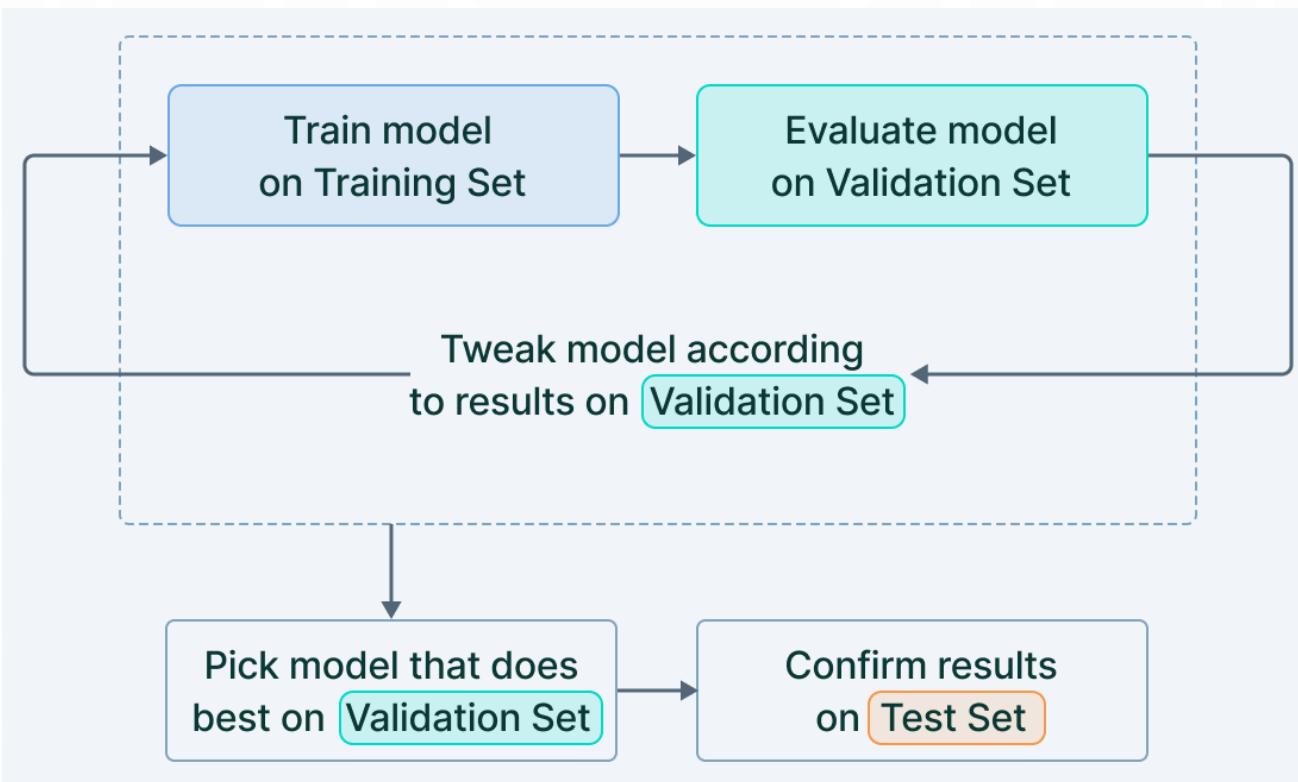
- The sample of data used to provide an unbiased evaluation of a final model fit on the training dataset.



# Percentage Split Examples

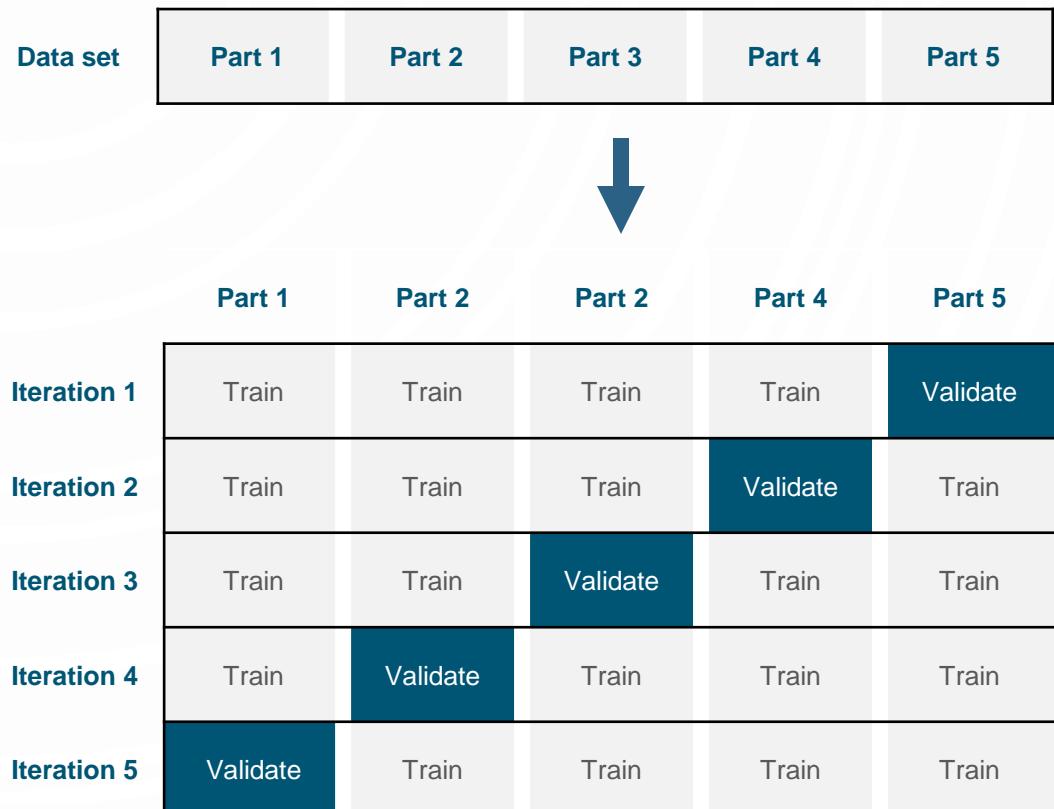


# Train, Validation and Test Sets

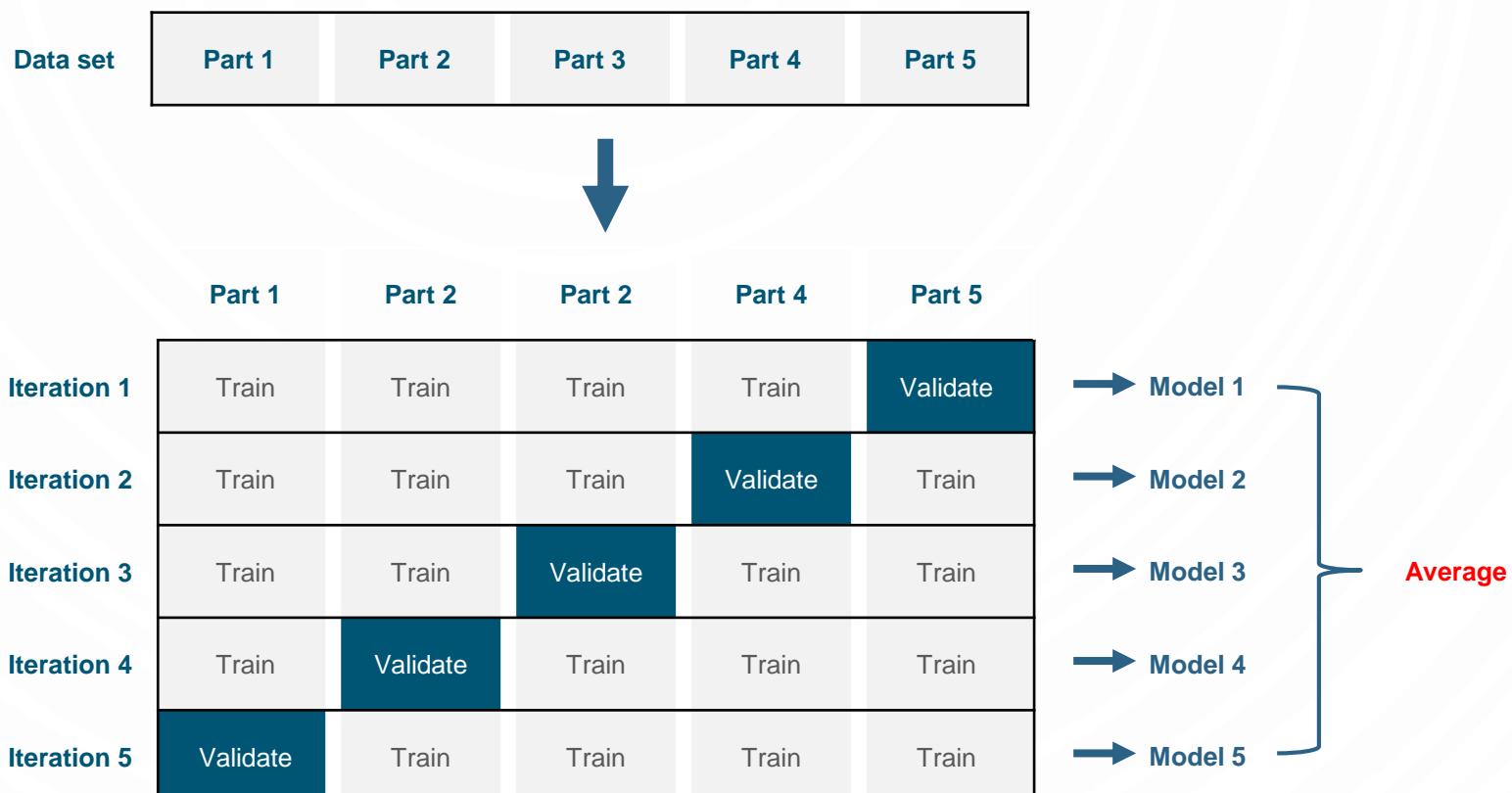


# Cross-validation

- The gold standard for evaluating model performance
- Reduce problems such as
  - Overfitting
  - Underfitting
- Gain insight into how the model is circulated to an independent data set
- Has the cost of creating many more models

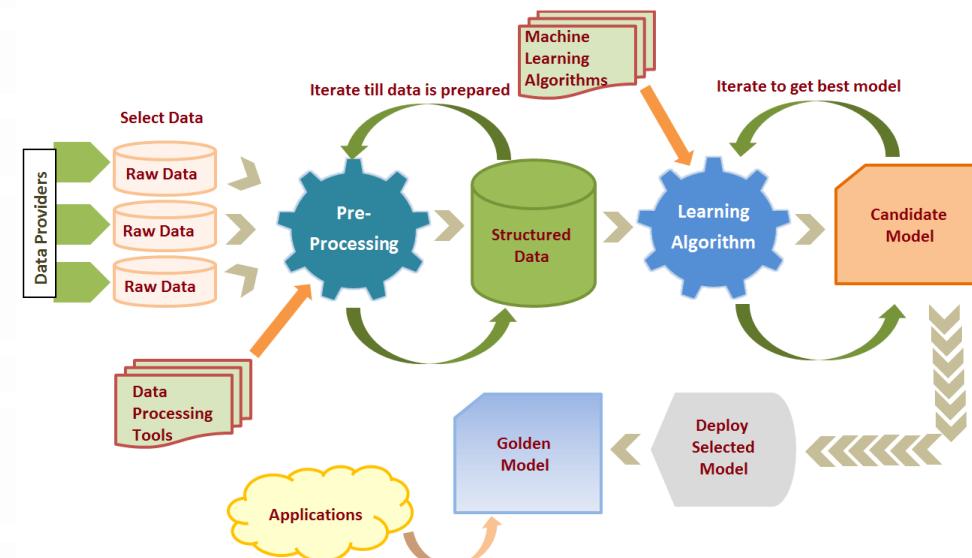


# Cross-validation



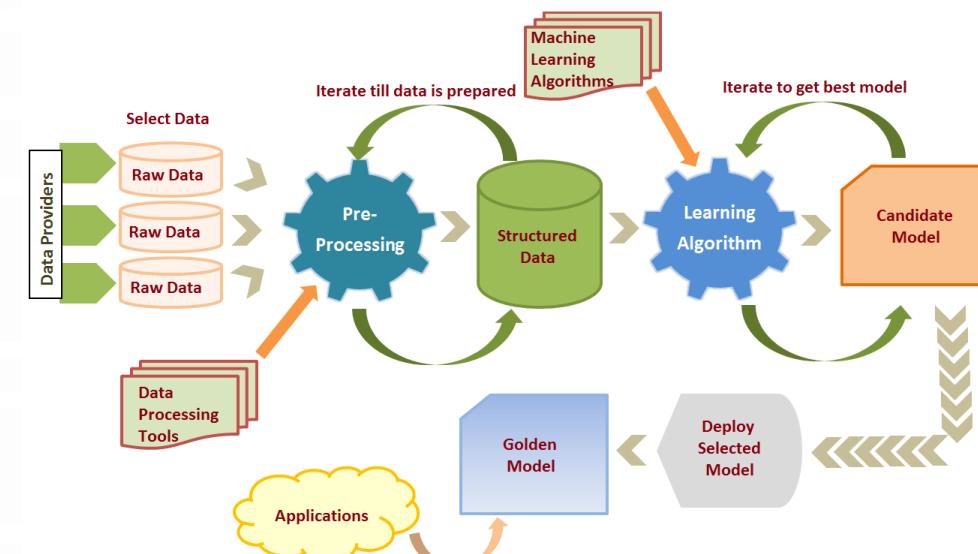
# Model Tuning

- If the performance of the candidate model is not satisfactory, it becomes necessary to
  - Use **advanced** strategies to improve it
  - Or **switch** to another model
  - Add **additional data**
  - Perform additional **data preparation** work



# Prediction

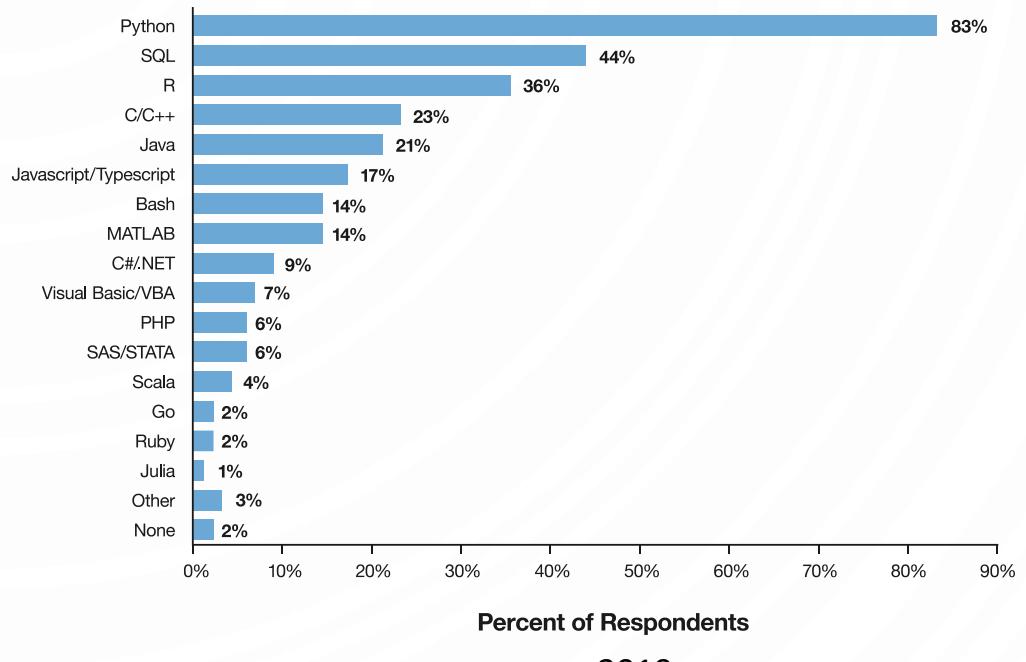
- Once previous steps are completed, it can be deployed for the task for which it is intended
  - If the model appears to be working satisfactorily



# How to start?

- Python is the most popular programming language to work in ML

What Programming Language Do You Use on a Regular Basis?



# Popular Machine Learning Frameworks

## Scikit-Learn

- A Python module for machine learning
- Started in 2007 as a Google Summer of Code project



## TensorFlow

- An open-source software library for machine learning applications
- Used at Google



TensorFlow

## PyTorch

- Has grown rapidly in popularity
- Supported by Facebook



# Other Machine Learning Frameworks

## Machine Learning Library (MLlib) in Apache Spark

- Scalable machine learning library by Apache
- Spark ML (spark.ml) is a new package introduced in Spark 1.2

## Caffe

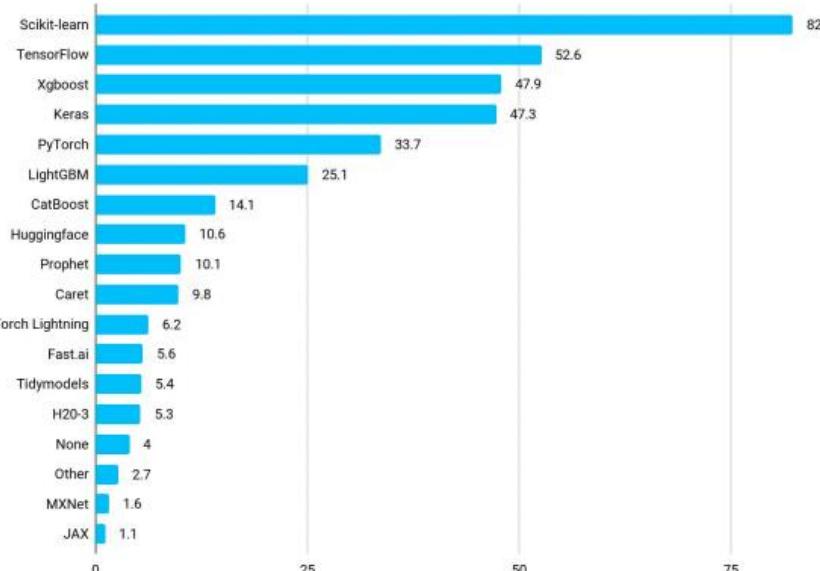
- A deep learning framework
- Developed by Berkeley AI Research (BAIR)

## Weka

- A collection of machine learning algorithms for data mining tasks
- Provide a GUI for intuitive experimentation

## Orange

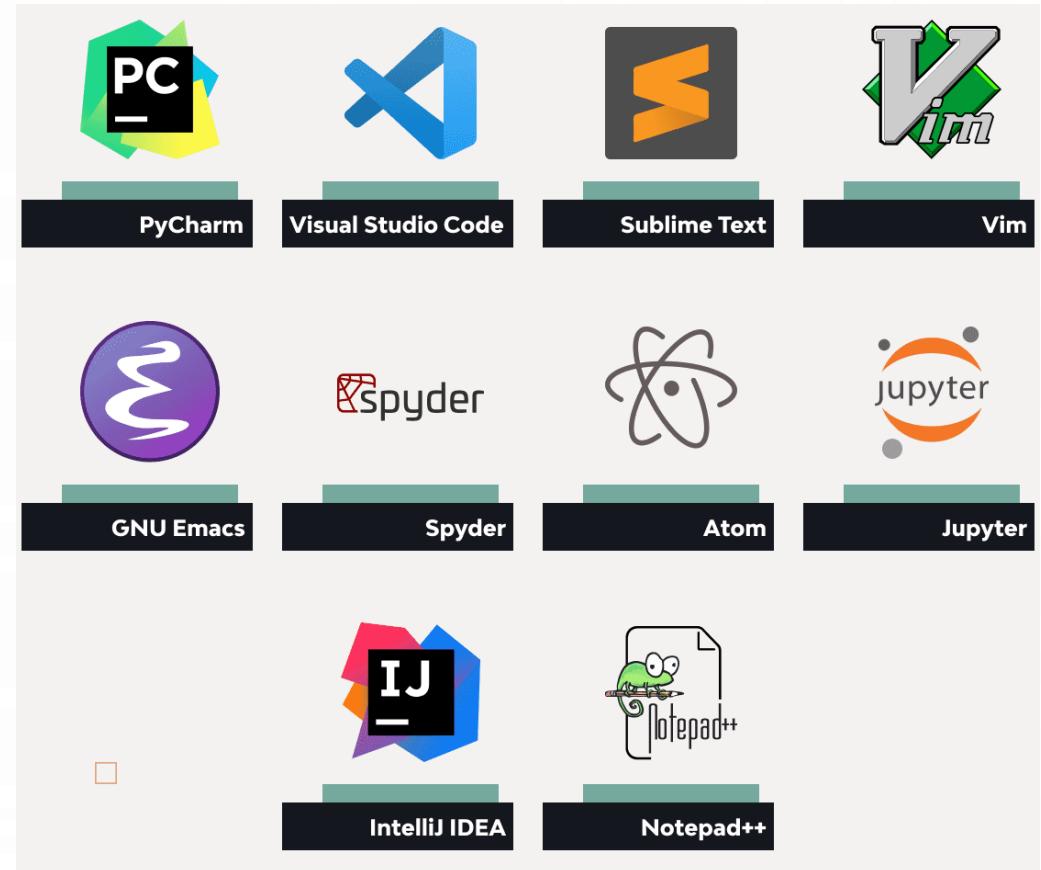
- Open-source machine learning and data visualization.



Top machine learning frameworks  
used by working data scientists 2021

# Python IDEs

- Jupyter Notebook
  - Google Colab
  - Kaggle Notebooks
  - Amazon SageMaker Studio Lab
  - Paperspace Gradient
- PyCharm
- Spyder
- Visual Studio Code



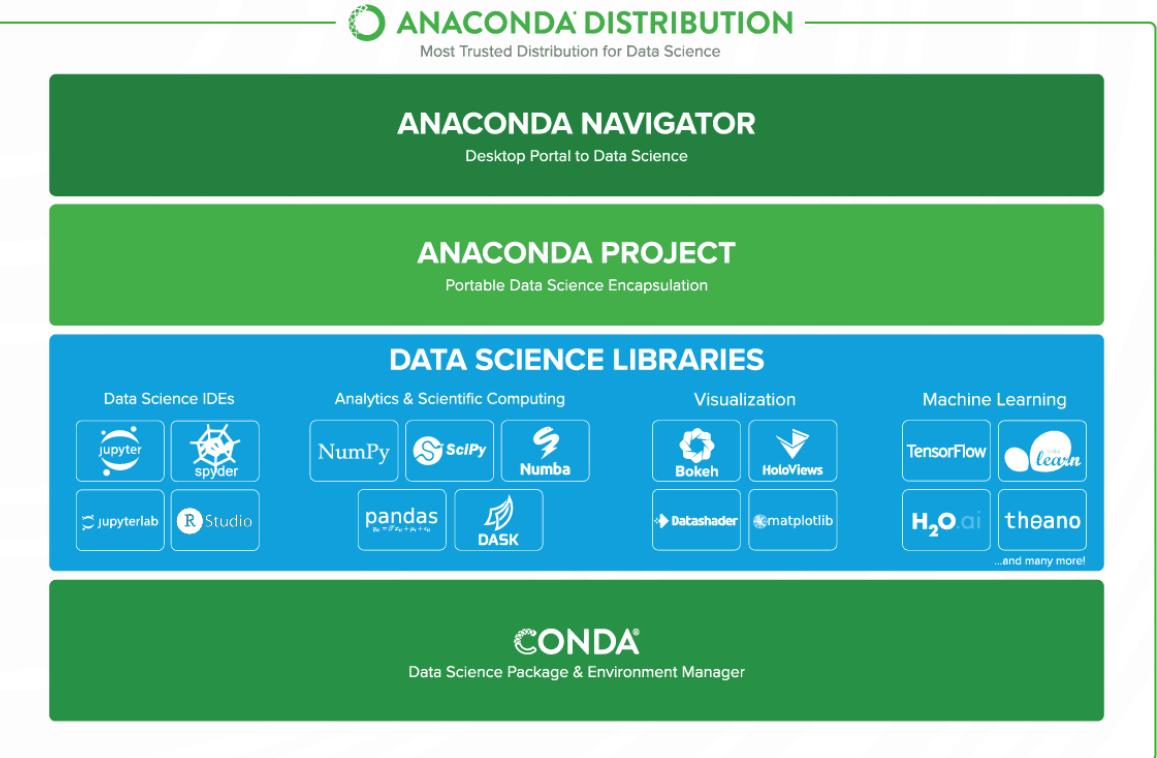
# Jupyter



- **Jupyter** is an interactive computational environment, in which you can combine code execution, rich text, mathematics, plots and rich media.
- It was spun off from **IPython** in 2014.
- The easiest way for a beginner to get started with Jupyter Notebooks is by installing **Anaconda**.

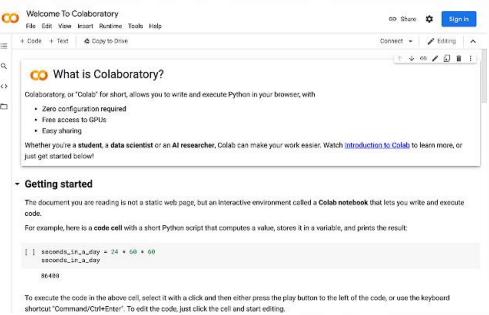
# Anaconda

- **Anaconda** is the most widely used Python distribution for data science and comes pre-loaded with all the most popular libraries and tools.
- To get Anaconda, simply:
  1. Download the latest version of Anaconda:
    - <https://www.anaconda.com>
  2. Install Anaconda by following the instructions.
  3. Start “Jupyter Notebook”



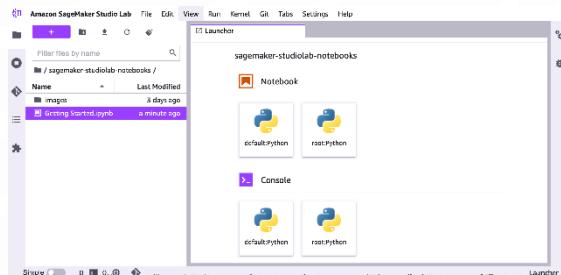
# Cloud Notebooks

## Google Colab



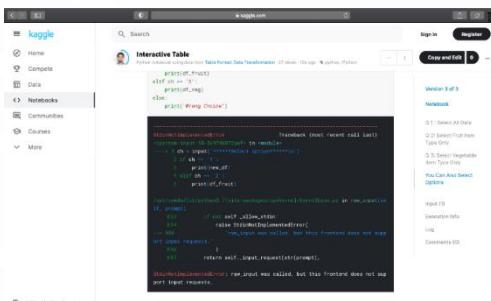
[colab.research.google.com](https://colab.research.google.com)

## Amazon SageMaker Studio Lab



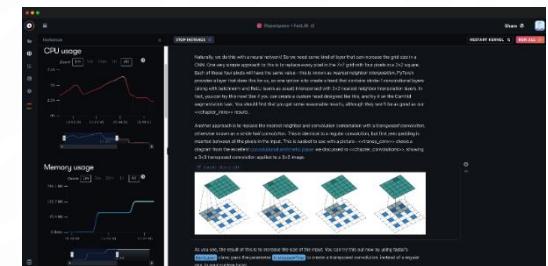
[studiolab.sagemaker.aws](https://studiolab.sagemaker.aws)

## Kaggle Notebooks



[www.kaggle.com/code](https://www.kaggle.com/code)

## Paperspace Gradient



[www.paperspace.com/gradient/notebooks](https://www.paperspace.com/gradient/notebooks)

# Cloud Notebooks Comparison\*

	Platform		
	Gradient	Colab	Kaggle
<b>Max Execution Time Per Session</b>	6 hours	12 hours	9 hours
<b>Idle Time</b>	1-6hours	30 Minutes	20 Minutes
<b>Usage Limit</b>	Subjected to availability	No limit	30 hours per week
<b>GPU Model</b>	Quadro M4000	K80	P100
<b>GPU Memory</b>	8GB	12GB	16GB
<b>CPU Model</b>	Intel(R) Xeon(R)	Intel(R) Xeon(R)	Intel(R) Xeon(R)
<b>CPU Clock Speed</b>	2.6GHz	2.3GHz	2.0GHz
<b>CPU RAM</b>	30GB	13GB	13GB
<b>Persistent Storage</b>	5GB	15GB*	20GB
<b>Private Notebooks</b>	No	Yes	Yes
<b>Background Execution</b>	No	No	Yes

\* Subject to change

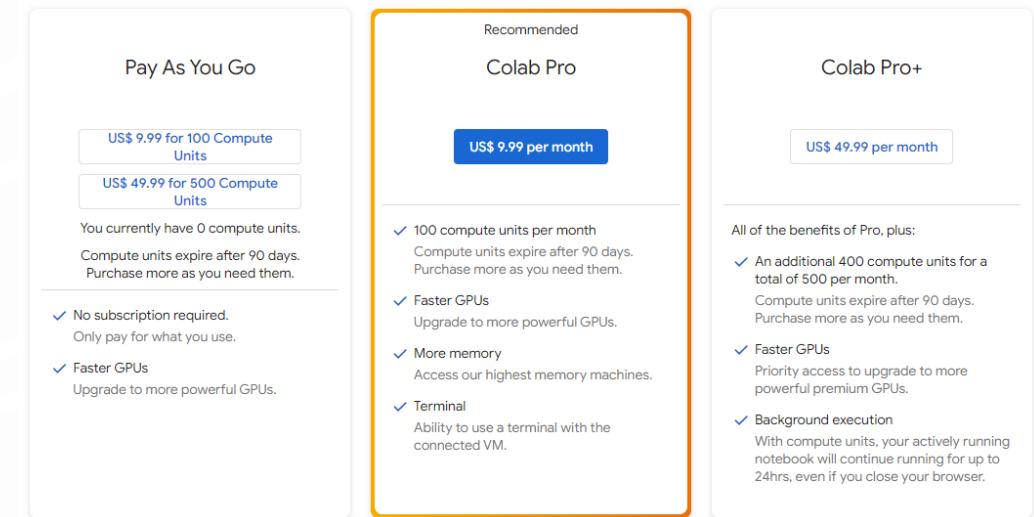
# Colab

colab

- Colaboratory, or “Colab” for short, is a product from Google Research.
  - Colab allows anybody to write and execute arbitrary python code through the browser, and is especially well suited to machine learning, data analysis and education.
- **Colab** is a hosted **Jupyter** notebook service that requires no setup to use, while providing access free of charge to computing resources including GPUs.
- Colab resources are **not guaranteed** and not unlimited, and the usage limits sometimes fluctuate.

# How long can notebooks run in Colab?

- Colab prioritizes interactive compute. Runtimes will time out if you are idle.
- In the version of Colab that is free of charge notebooks can run for at most 12 hours, depending on availability and your usage patterns.
  - Colab Pro, Pro+, and Pay As You Go offer you increased compute availability based on your compute unit balance.
- Colab Pro+ supports continuous code execution for up to 24 hours if you have sufficient compute units.
  - Idle timeouts only apply if code execution terminates.



# Supporting libraries used for machine learning



# Introduction to NumPy



- Numpy is an open-source library for working efficiently with **arrays**.
- Developed in 2005 by Travis Oliphant, the name stands for **Numerical Python**.

- How to create a NumPy array?

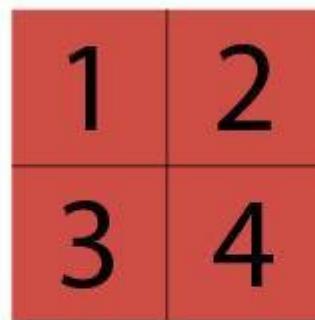
```
import numpy as np  
list1 = [0,1,2,3,4]  
arr1d = np.array(list1)  
print(arr1d) >>> array([0, 1, 2, 3, 4])
```

# Dimensions



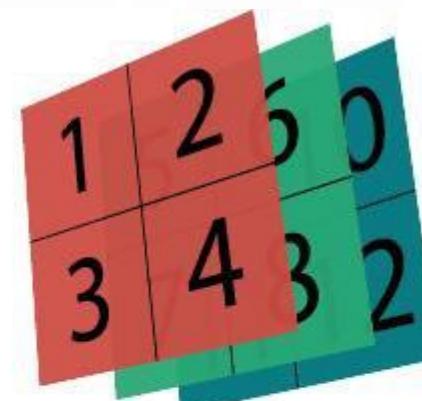
Vector

```
np.array([1, 2])
```



Matrix

```
np.array([[1, 2], [3, 4]])
```



3D Matrix

```
np.array([[[1, 2], [3, 4]],  
         [[5, 6], [7, 8]],  
         [[9, 10], [11, 12]]])
```

# Introduction to Pandas



- Pandas is an open-source library that is made mainly for working with relational or labeled data both easily and intuitively.
- It provides various data structures and operations for manipulating numerical data and time series.
- This library is built **on top of the NumPy library**.
- Pandas is fast and it has high performance & productivity for users.
- Pandas generally provide two data structures for manipulating data:
  - Series
  - DataFrame

# Series VS Dataframe



**Series**

	apples
0	3
1	2
2	0
3	1

**Series**

	oranges
0	0
1	3
2	7
3	2

+

**DataFrame**

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

=

# Introduction to Matplotlib

matplotlib

- There are thousands of libraries in Python, and Matplotlib is one of the most powerful tools for data visualization in Python.
- Matplotlib tries to make easy things easy and hard things possible.
- You can generate plots, histograms, power spectra, bar charts, errorcharts, scatterplots, etc., with just a few lines of code.

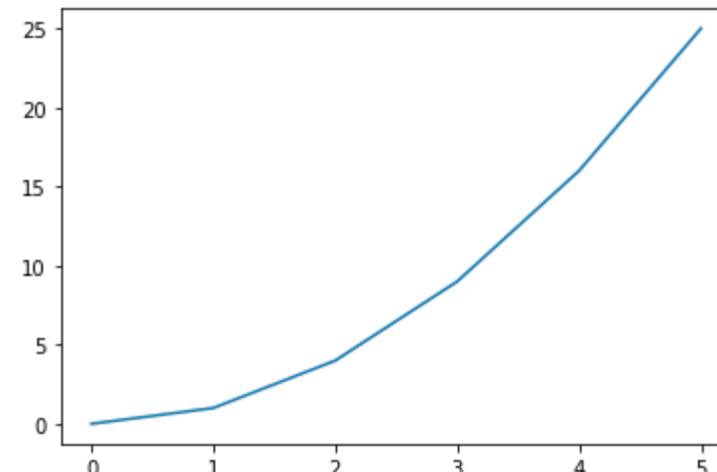


# Type of graphs

matplotlib

## Line Graphs

```
#create data for plotting  
x_values = [0,1,2,3,4,5]  
squares = [0,1,4,9,16,25]  
  
#the default graph style for plot is a line  
plt.plot(x_values, squares)  
  
#display the graph  
plt.show
```



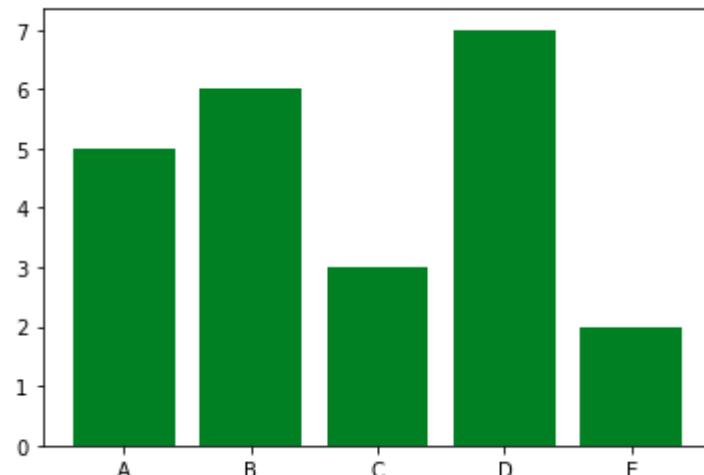
# Type of graphs

matplotlib

## Bar graphs

```
#create data for plotting
x_values = [5,6,3,7,2]
y_values = ["A", "B", "C", "D", "E"]

plt.bar(y_val,x_values, color = "green")
plt.show()
```



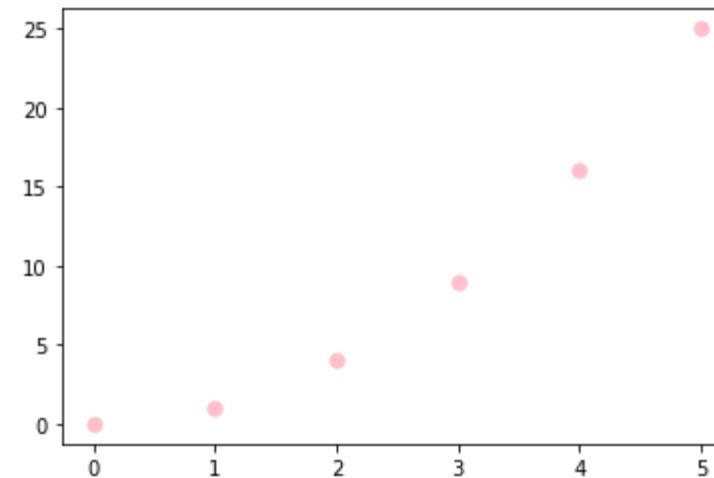
# Type of graphs

matplotlib

## Scatter Plots

```
#create data for plotting  
x_values = [0,1,2,3,4,5]  
squares = [0,1,4,9,16,25]  
  
plt.scatter(x_values,squares, s=10, color = "pink")  
plt.show()
```

marker size



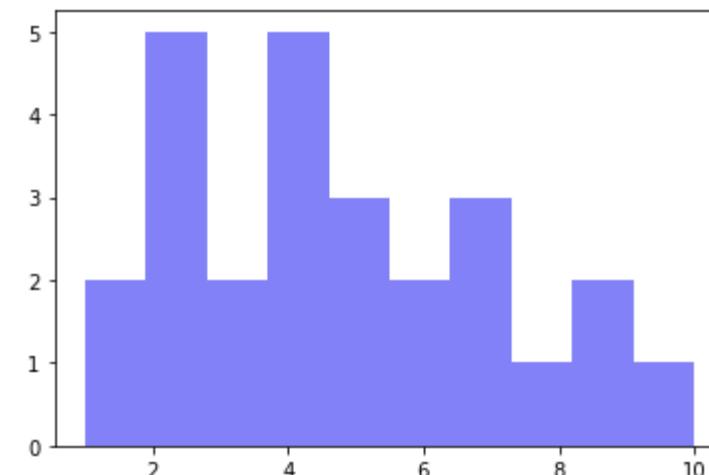
# Type of graphs

matplotlib

## Histograms

```
#generate fake data  
x = [2,1,6,4,2,4,8,9,4,2,4,10,6,4,5,7,7,3,2,7,5,3,5,9,2,1]  
  
#plot for a histogram  
plt.hist(x, bins = 10, color='blue', alpha=0.5)  
plt.show()
```

↑  
transparency value



# Introduction to Scikit-Learn



- Scikit-learn (or sklearn for short) is a free open-source machine learning library for Python.
- It is designed to cooperate with NumPy library and simplifies data science techniques in Python with built-in support for popular classification, regression, and clustering machine learning algorithms.
- Scikit-learn provides tools for:
  - Regression
  - Classification
  - Model selection
  - Clustering
  - Preprocessing

# Simple sklearn Example

```
from sklearn import svm
from sklearn import datasets

# Load dataset
iris = datasets.load_iris()

# Define
clf = svm.LinearSVC()

# learn from the data
clf.fit(iris.data, iris.target)

# predict for unseen data
clf.predict([[ 5.0,  3.6,  1.3,  0.25]])
```