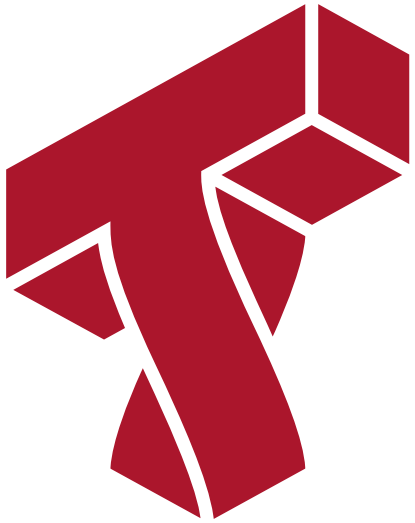


Lecture 1: Introduction to Machine Learning

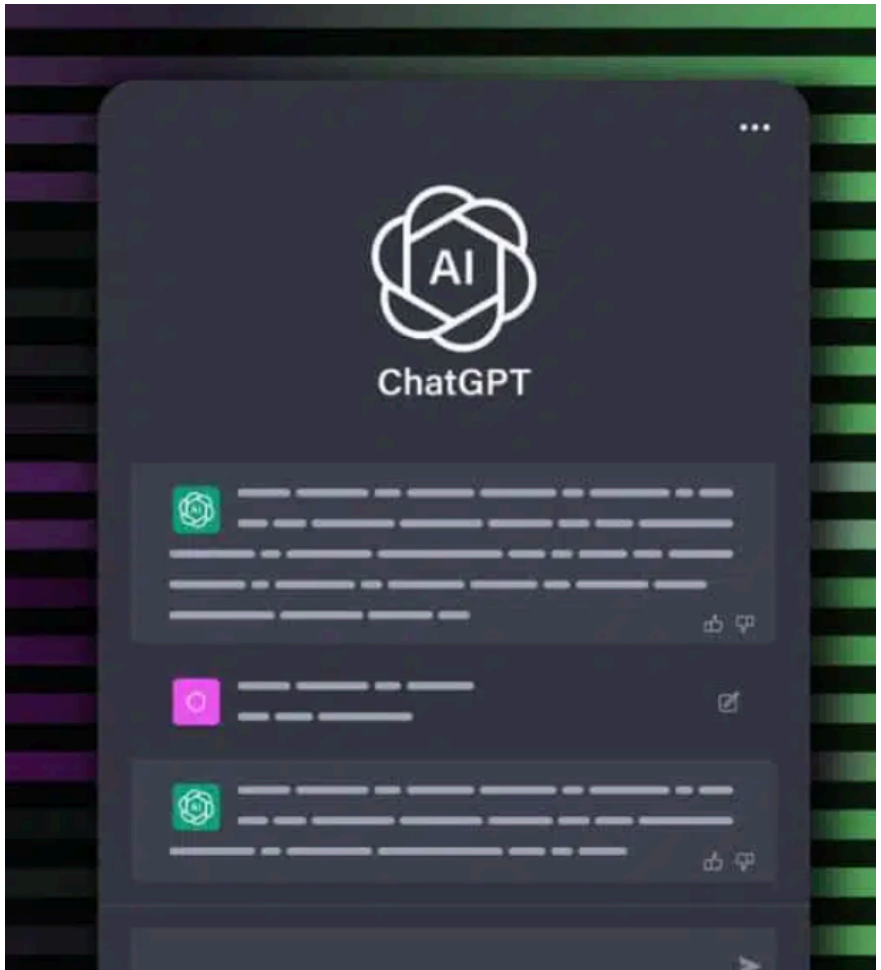
Welcome to Applied Machine Learning!

Machine learning is one of today's most exciting emerging technologies.

In this course, you will learn what machine learning is, what are some of the most important algorithms in machine learning, and how to apply them to solve problems in the real world.

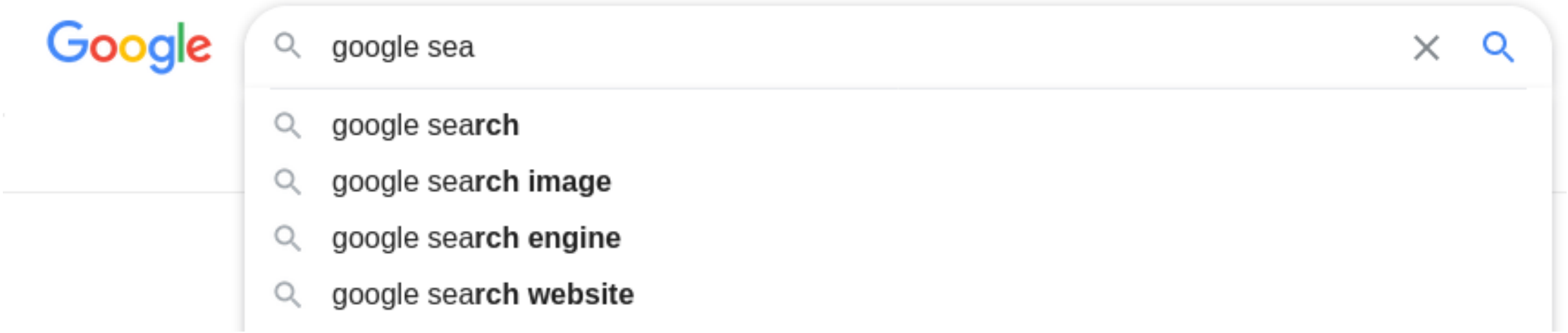
Part 1: What is Machine Learning?

The last few years have produced impressive advances in artificial intelligence.



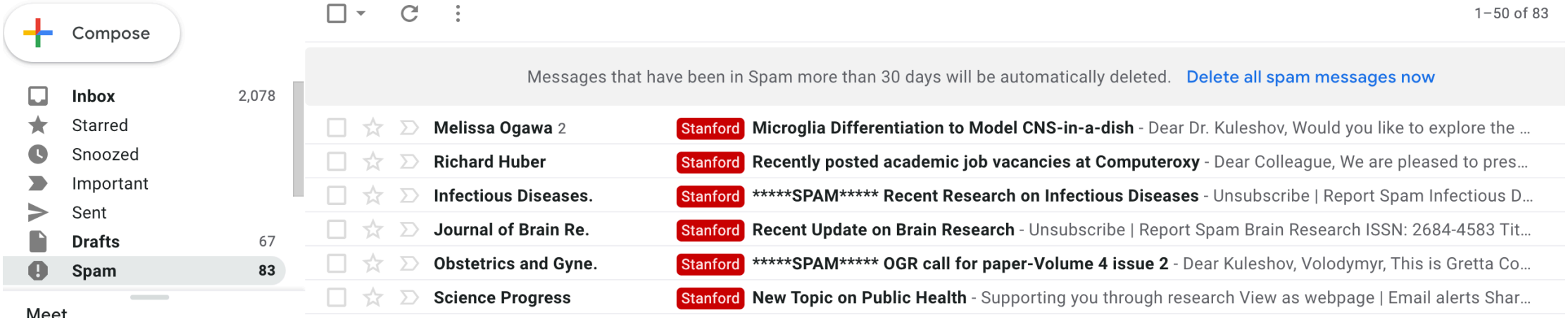
ML in Everyday Life: Search Engines

You use machine learning every day when use a search engine.



ML in Everyday Life: Spam/Fraud Detection

Machine learning is used in every spam filter, such as in Gmail.



ML systems are also used by credit card companies and banks to automatically detect fraudulent behavior.

ML in Everyday Life: Self-Driving Cars

One of the most exciting and cutting-edge uses of machine learning algorithms is in autonomous vehicles.

A Definition of Machine Learning

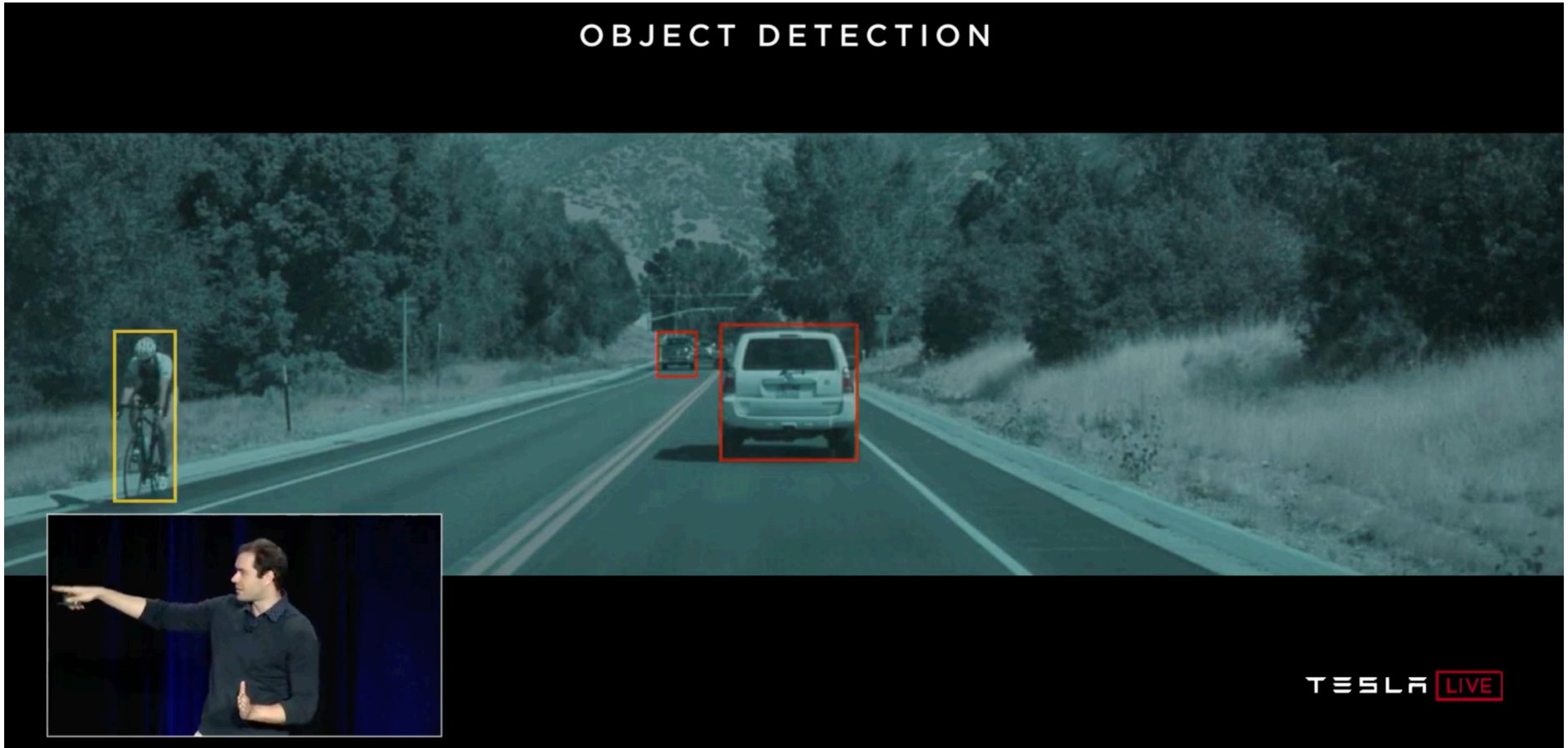
In 1959, Arthur Samuel defined machine learning as follows.

> *Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed.*

What does "learn" and "explicitly programmed" mean here? Let's look at an example.

An Example: Self Driving Cars

A self-driving car system uses dozens of components that include detection of cars, pedestrians, and other objects.



Self Driving Cars: A Rule-Based Algorithm

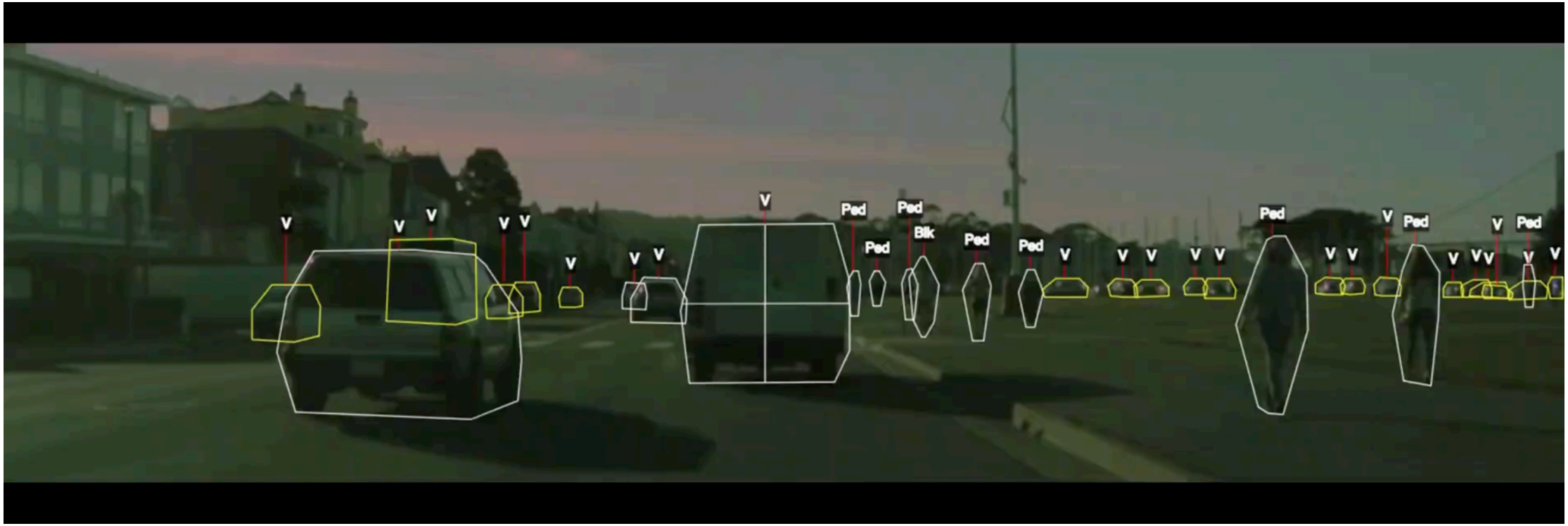
One way to build a detection system is to write down rules.

```
# pseudocode example for a rule-based classification system
object = camera.get_object()
if object.has_wheels(): # does the object have wheels?
    if len(object.wheels) == 4: return "Car" # four wheels => car
    elif len(object.wheels) == 2:,
        if object.seen_from_back():
            return "Car" # viewed from back, car has 2 wheels
        else:
            return "Bicycle" # normally, 2 wheels => bicycle
return "Unknown" # no wheels? we don't know what it is
```

In practice, it's almost impossible for a human to specify all the edge cases.

Self Driving Cars: An ML Approach

The machine learning approach is to teach a computer how to do detection by showing it many examples of different objects.



No manual programming is needed: the computer learns what defines a pedestrian

Revisiting Our Definition of ML

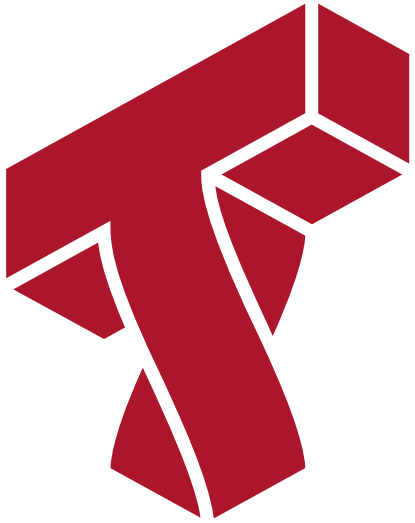
> *Machine learning is a field of study that gives computers the ability to learn without being explicitly programmed. (Arthur Samuel, 1959.)*

This principle can be applied to countless domains: medical diagnosis, factory automation, machine translation, and many more!

Why Machine Learning?

Why is this approach to building software interesting?

- It lets us build practical systems for real-world applications for which other engineering approaches don't work.
- Learning is widely regarded as a key approach towards building general-purpose artificial intelligence systems.
- The science and engineering of machine learning offers insights into human intelligence.



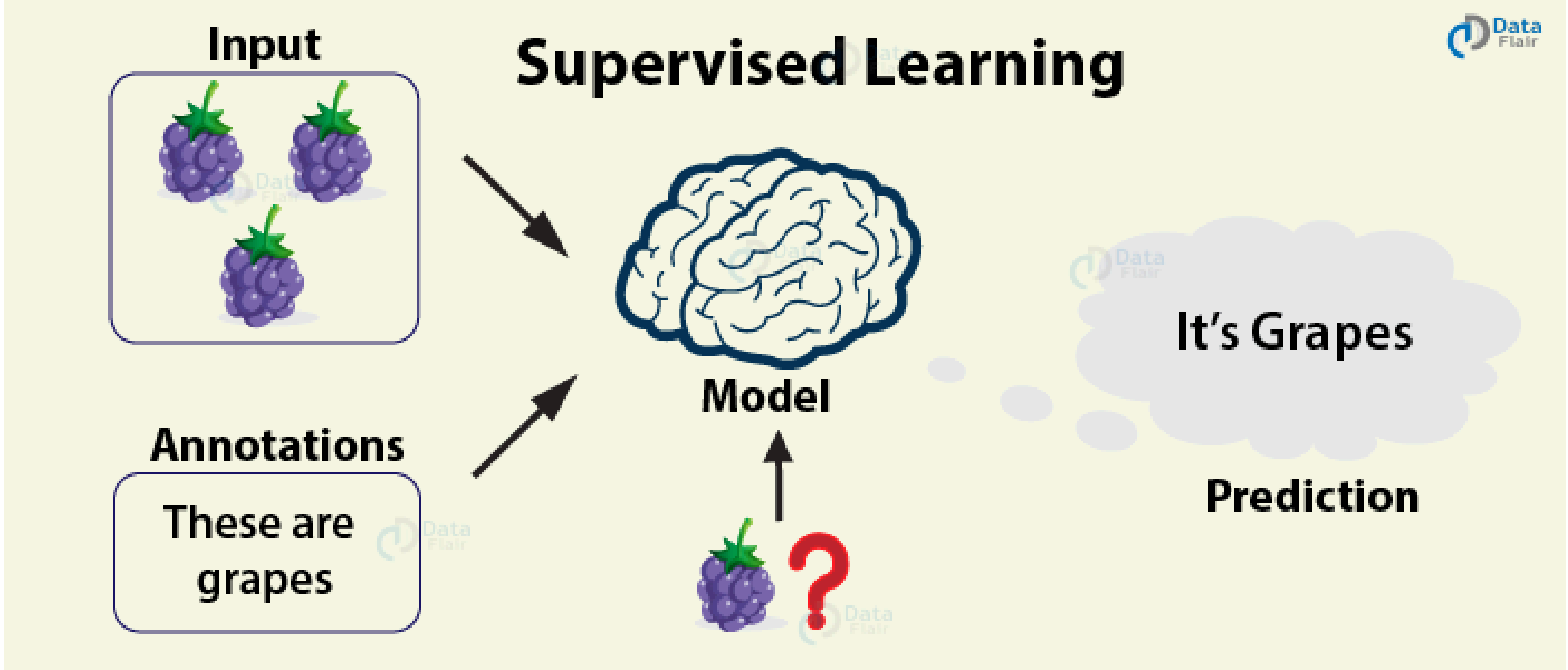
Part 2: Three Approaches to Machine Learning

Machine learning is broadly defined as the science of building software that has the ability to learn without being explicitly programmed.

How might we enable machines to learn? Let's look at a few examples.

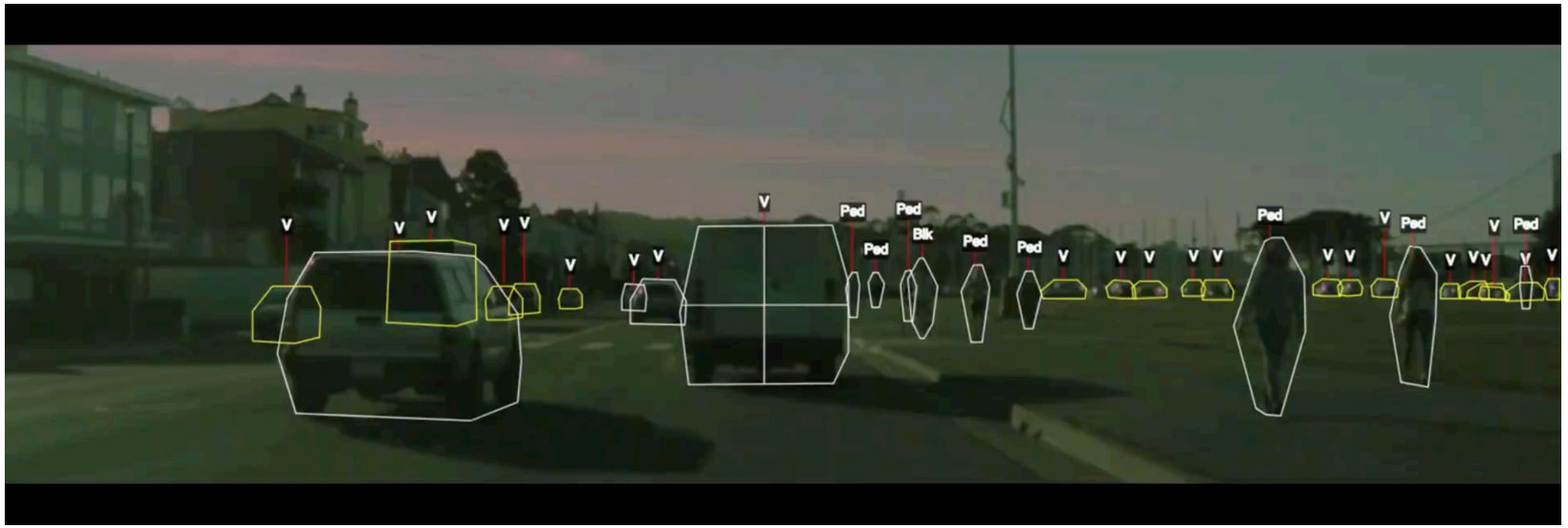
Supervised Learning

The most common approach to machine learning is supervised learning.



Supervised Learning: Object Detection

We previously saw an example of supervised learning: object detection.



1. We start by collecting a dataset of labeled objects.
2. We train a model to output accurate predictions on this dataset.
3. When the model sees new, similar data, it will also be accurate.

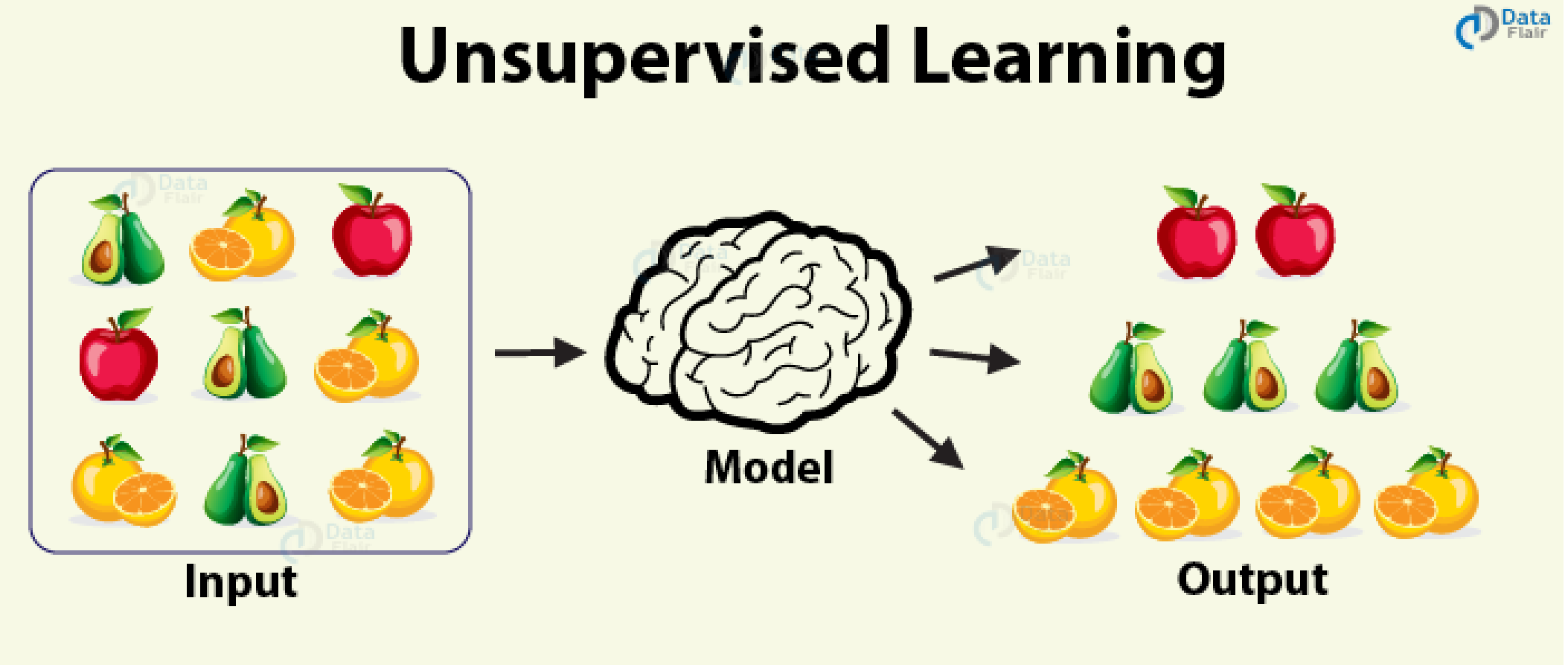
Applications of Supervised Learning

Many important applications of machine learning are supervised:

- Classifying medical images.
- Translating between pairs of languages.
- Detecting objects in autonomous driving.

Unsupervised Learning

Here, we have a dataset *without* labels. Our goal is to learn something interesting about the structure of the data.



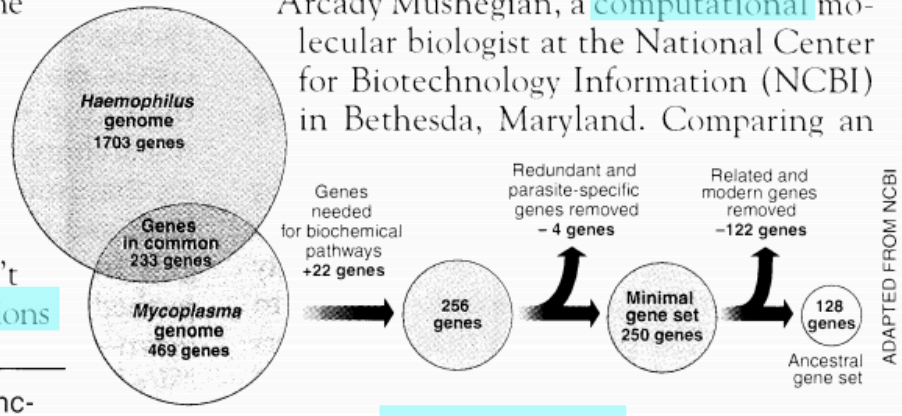
Unsupervised Learning : Text Analysis

In this next example, we have a text containing at least four distinct topics.

Seeking Life's Bare (Genetic) Necessities

COLD SPRING HARBOR, NEW YORK—How many genes does an organism need to survive? Last week at the genome meeting here,* two genome researchers with radically different approaches presented complementary views of the basic genes needed for life. One research team, using computer analyses to compare known genomes, concluded that today's organisms can be sustained with just 250 genes, and that the earliest life forms required a mere 128 genes. The other researcher mapped genes in a simple parasite and estimated that for this organism, 800 genes are plenty to do the job—but that anything short of 100 wouldn't be enough.

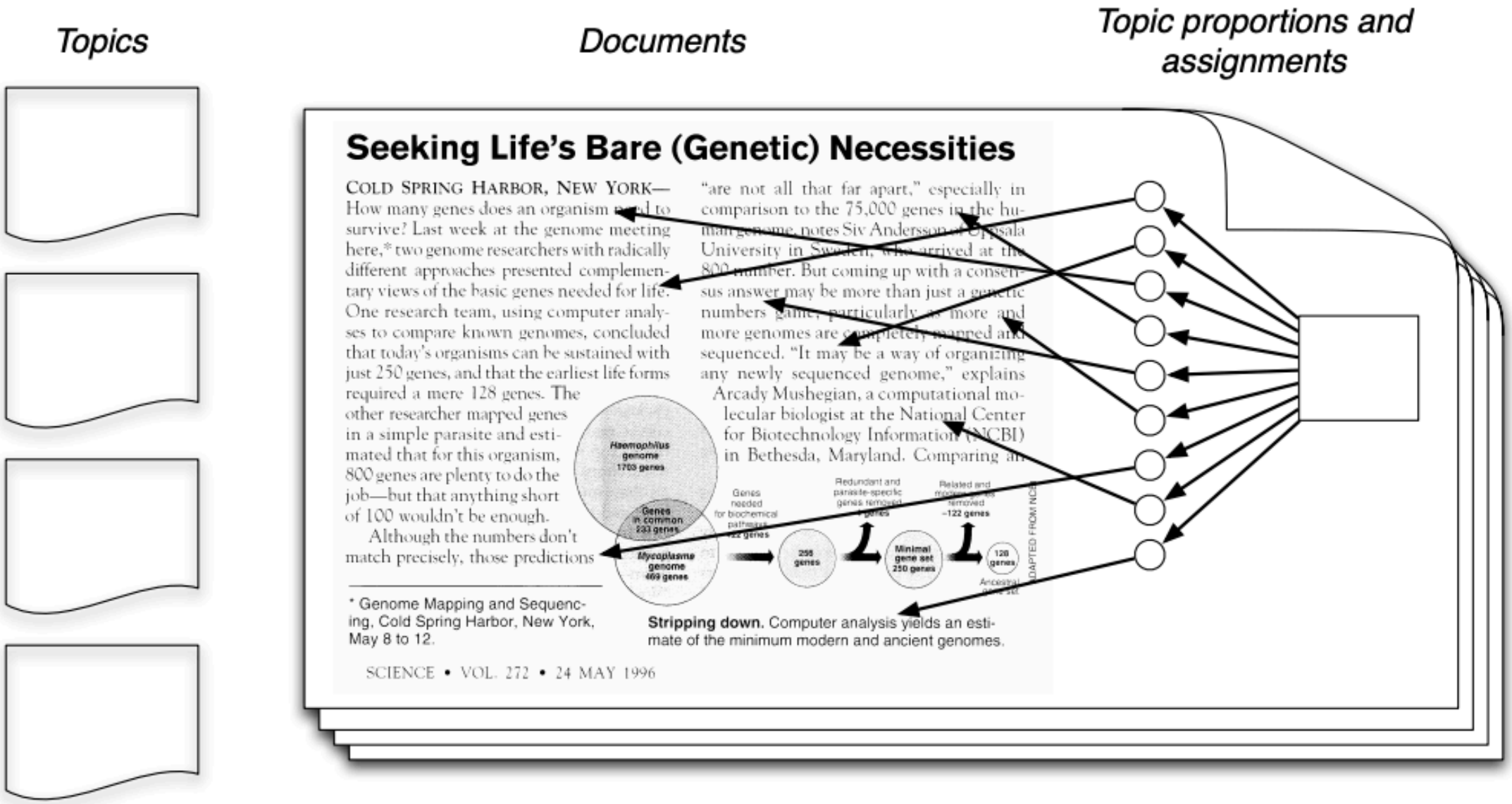
Although the numbers don't match precisely, those predictions "are not all that far apart," especially in comparison to the 75,000 genes in the human genome, notes Siv Andersson of Uppsala University in Sweden, who arrived at the 800 number. But coming up with a consensus answer may be more than just a genetic numbers game, particularly as more and more genomes are completely mapped and sequenced. "It may be a way of organizing any newly sequenced genome," explains Arcady Mushegian, a computational molecular biologist at the National Center for Biotechnology Information (NCBI) in Bethesda, Maryland. Comparing an



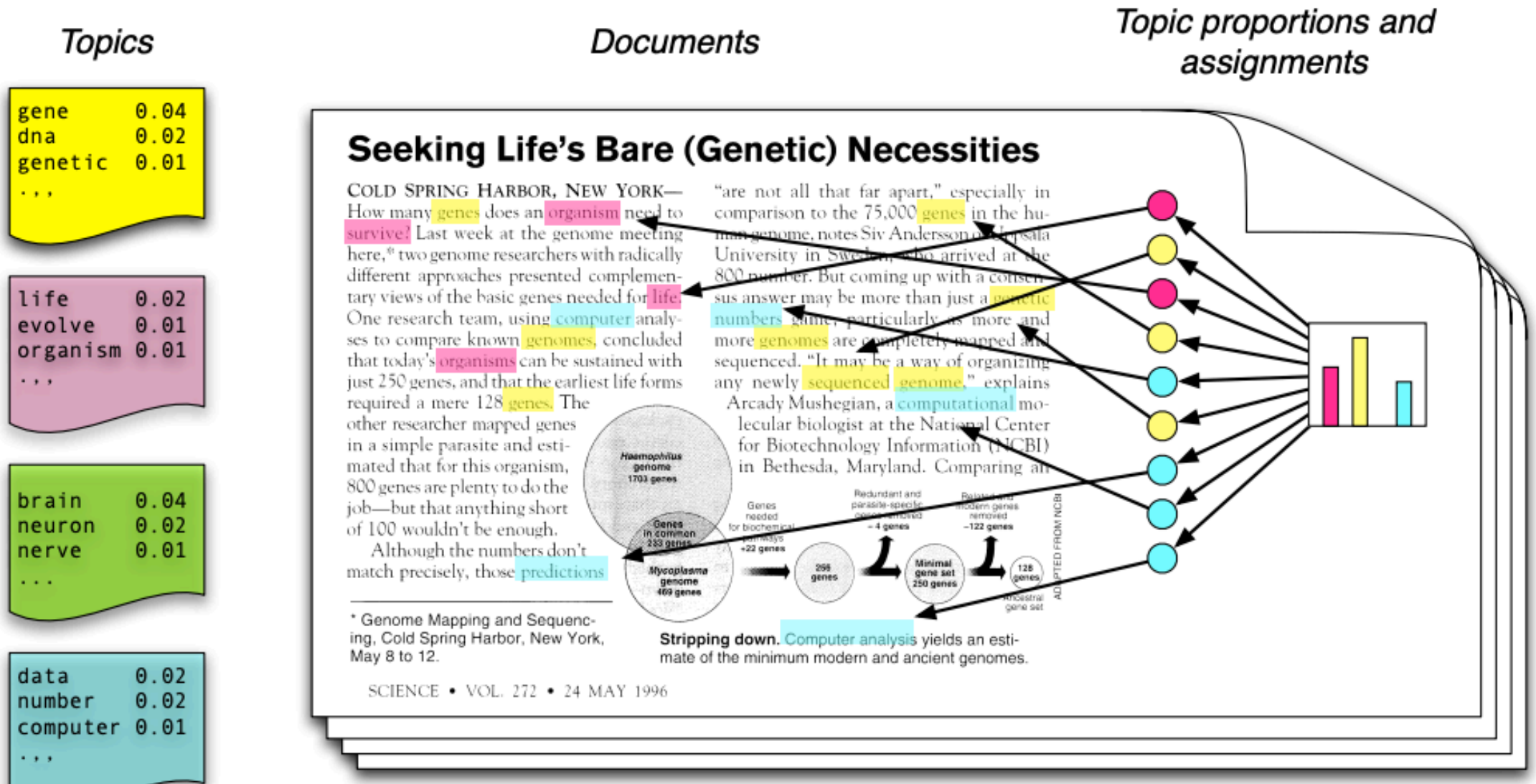
Stripping down. Computer analysis yields an esti-

* Genome Mapping and Sequencing, Cold Spring Harbor, New York,

However, we initially do not know what the topics are.



Unsupervised *topic modeling* algorithms assign each word in a document to a topic and compute topic proportions for each document.



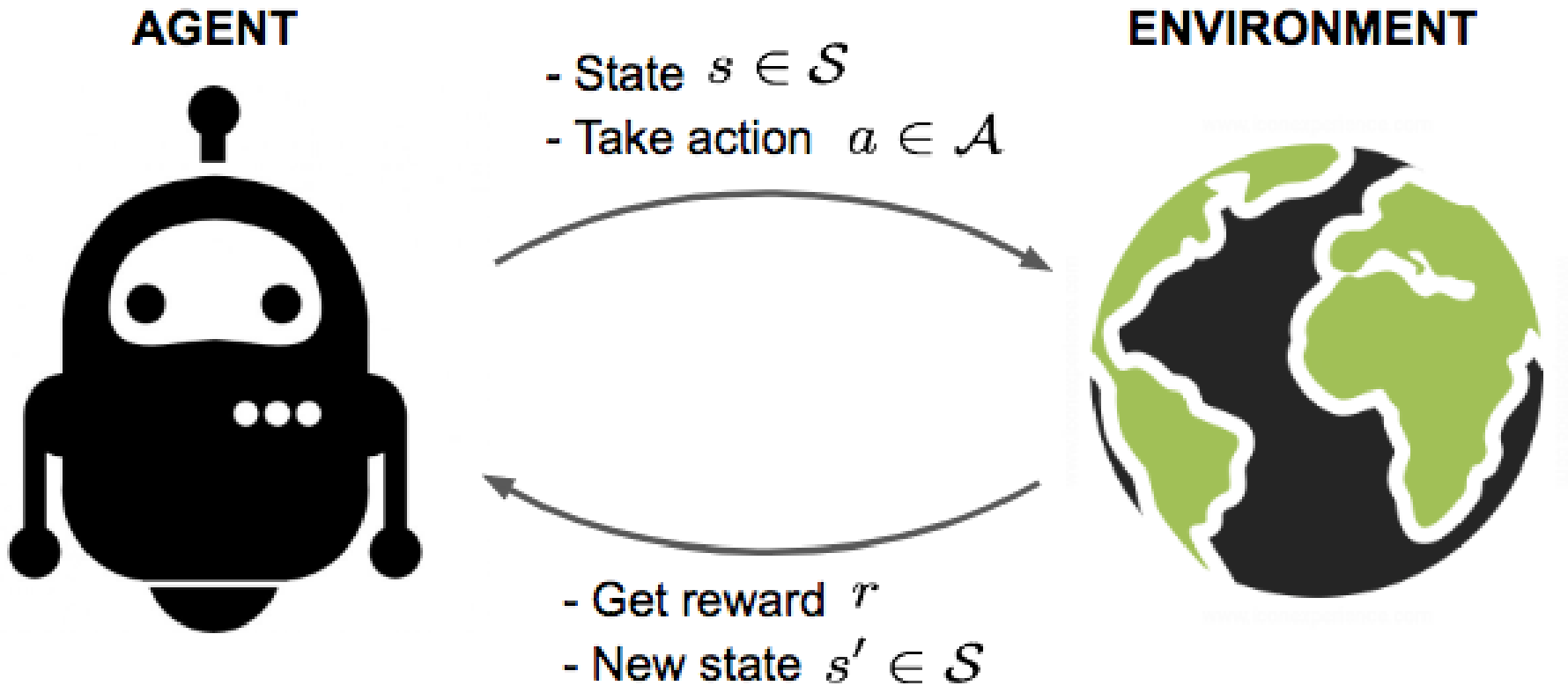
Applications of Unsupervised Learning

Unsupervised learning methods have many other applications:

- **Recommendation systems**: suggesting movies on Netflix.
- **Anomaly and outlier detection**: identifying factory components that are likely to fail soon.
- **Signal processing**: extracting clean human speech from a noisy audio recording.

Reinforcement Learning

In reinforcement learning, an agent is interacting with the world over time. We teach it good behavior by providing it with rewards.



Applications of Reinforcement Learning

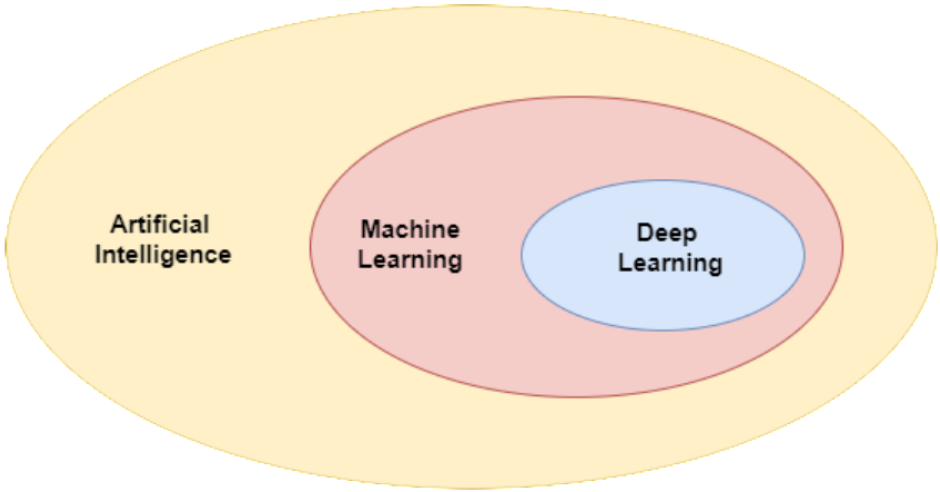
Applications of reinforcement learning include:

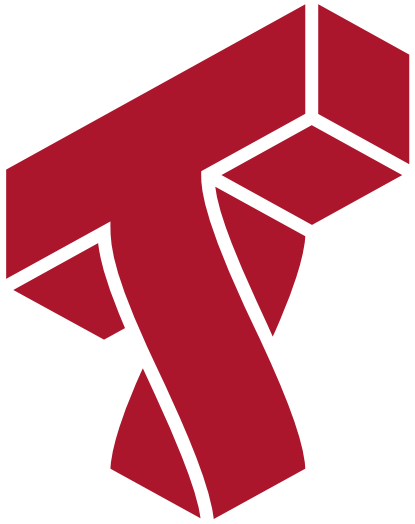
- Creating **agents** that play games such as Chess or Go.
- **Industrial control**: automatically operating cooling systems in datacenters to use energy more efficiently.
- **Generative design** of new drug compounds.

Artificial Intelligence and Deep Learning

Machine learning is closely related to these two fields.

- AI is about building machines that exhibit intelligence.
- ML enables machines to learn from experience, a useful tool for AI.
- Deep learning focuses on a family of learning algorithms loosely inspired by the brain.





Part 3: Logistics

We conclude the lecture with the logistical aspects of the course.

What Is the Course About?

This course studies the foundations and applications of machine learning.

- **Algorithms**: We cover a *broad* set of ML algorithms: linear models, boosted decision trees, neural networks, SVMs, etc.
- **Foundations**: We explain why they work using *math*. We cover maximum likelihood, generalization, regularization, etc.
- **Implementation**: We teach how to *implement* algorithms from scratch using `numpy` or `sklearn`

We also cover many practical aspects of applying machine learning.

Course Contents

Some of the most important sets of topics we will cover include:

- **Basics of Supervised Learning**: Regression, classification, overfitting, regularization, generative vs. discriminative models
- **Unsupervised Learning**: Clustering, dimensionality reduction, etc.
- **Advanced Supervised Learning**: Support vector machines, kernel methods, decision trees, boosting, deep learning.
- **Applying ML**: Overfitting, error analysis, learning curves, etc.

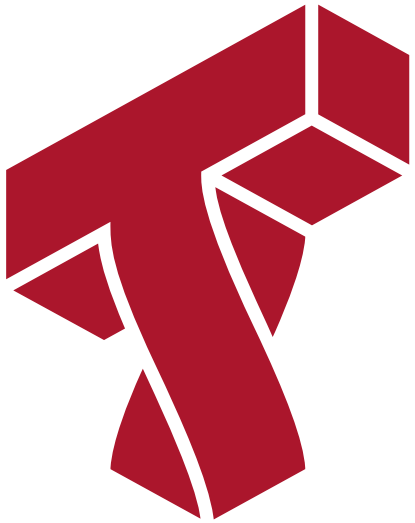
Prerequisites: Is This Course For You?

The main requirements for this course are:

- **Programming**: At least one year of experience, preferably in Python.
- **Linear Algebra**: College-level familiarity with matrix operations, eigenvectors, the SVD, vector and matrix norms, etc.
- **Probability**. College-level understanding of probability distributions, random variables, Bayes' rule, etc.

This course does not assume any prior ML experience.

Again, Welcome to Applied Machine Learning!



Software You Will Use

You will use Python and popular machine learning libraries such as:

- `scikit-learn`. It implements most classical machine learning algorithms.
- `tensorflow`, `keras`, `pytorch`. Standard libraries for modern deep learning.
- `numpy`, `pandas`. Linear algebra and data processing libraries used to implement algorithms from scratch.

Executable Course Materials

The core materials for this course (including the slides!) are created using Jupyter notebooks.

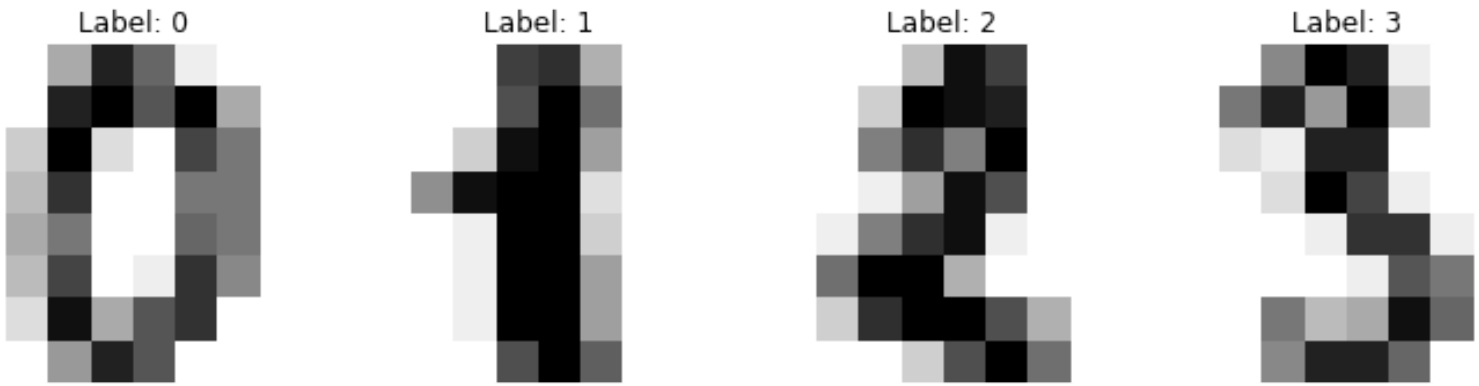
- We are going to embed an execute code directly in the slides and use that to demonstrate algorithms.
- These slides can be downloaded locally and all the code can be reproduced.

```
import numpy as np
import matplotlib.pyplot as plt
from sklearn import datasets, neural_network
plt.rcParams['figure.figsize'] = [12, 4]
```

We can use these libraries to load a simple datasets of handwritten digits.

```
# https://scikit-learn.org/stable/auto_examples/classification/plot_digits_classification.html
# load the digits dataset
digits = datasets.load_digits()

# The data that we are interested in is made of 8x8 images of digits, let's
# have a look at the first 4 images.
_, axes = plt.subplots(1, 4)
images_and_labels = list(zip(digits.images, digits.target))
for ax, (image, label) in zip(axes, images_and_labels[:4]):
    ax.set_axis_off()
    ax.imshow(image, cmap=plt.cm.gray_r, interpolation='nearest')
    ax.set_title('Label: %i' % label)
```



We can now load and train this algorithm inside the slides.

```
np.random.seed(0)
# To apply a classifier on this data, we need to flatten the image, to
# turn the data in a (samples, feature) matrix:
data = digits.images.reshape((len(digits.images), -1))

# create a small neural network classifier
from sklearn.neural_network import MLPClassifier
classifier = MLPClassifier(alpha=1e-3)

# Split data into train and test subsets
X_train, X_test, y_train, y_test = sk.model_selection.train_test_split(data, digits.target, test_size=0.5, shuffle=False)

# We learn the digits on the first half of the digits
classifier.fit(X_train, y_train)

# Now predict the value of the digit on the second half:
predicted = classifier.predict(X_test)
```

We can now visualize the results.

```
_, axes = plt.subplots(1, 4)
images_and_predictions = list(zip(digits.images[n_samples // 2:], predicted))
for ax, (image, prediction) in zip(axes, images_and_predictions[:4]):
    ax.imshow(image, cmap=plt.cm.gray_r, interpolation='nearest')
    ax.set_title('Prediction: %i' % prediction)
```