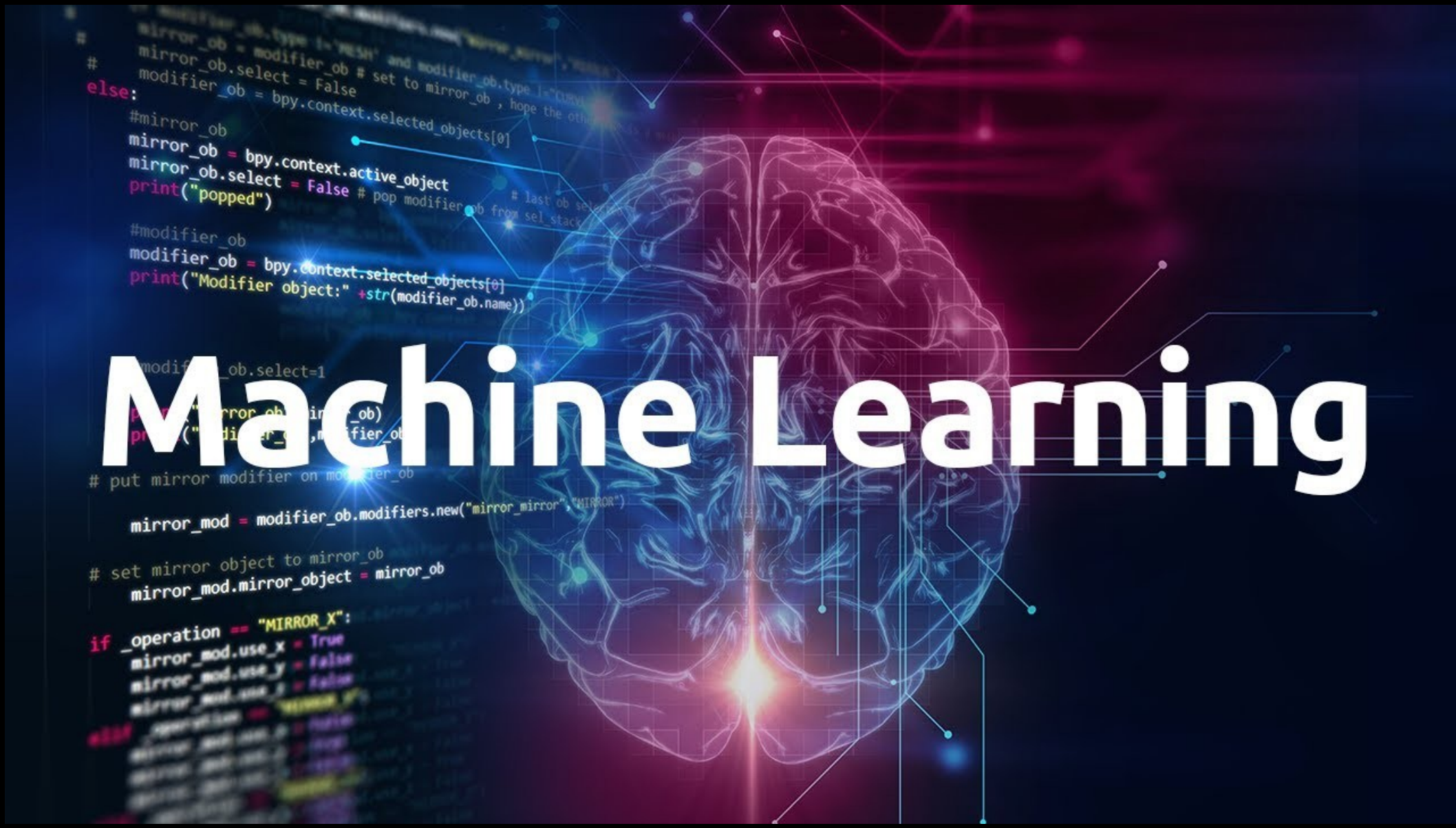
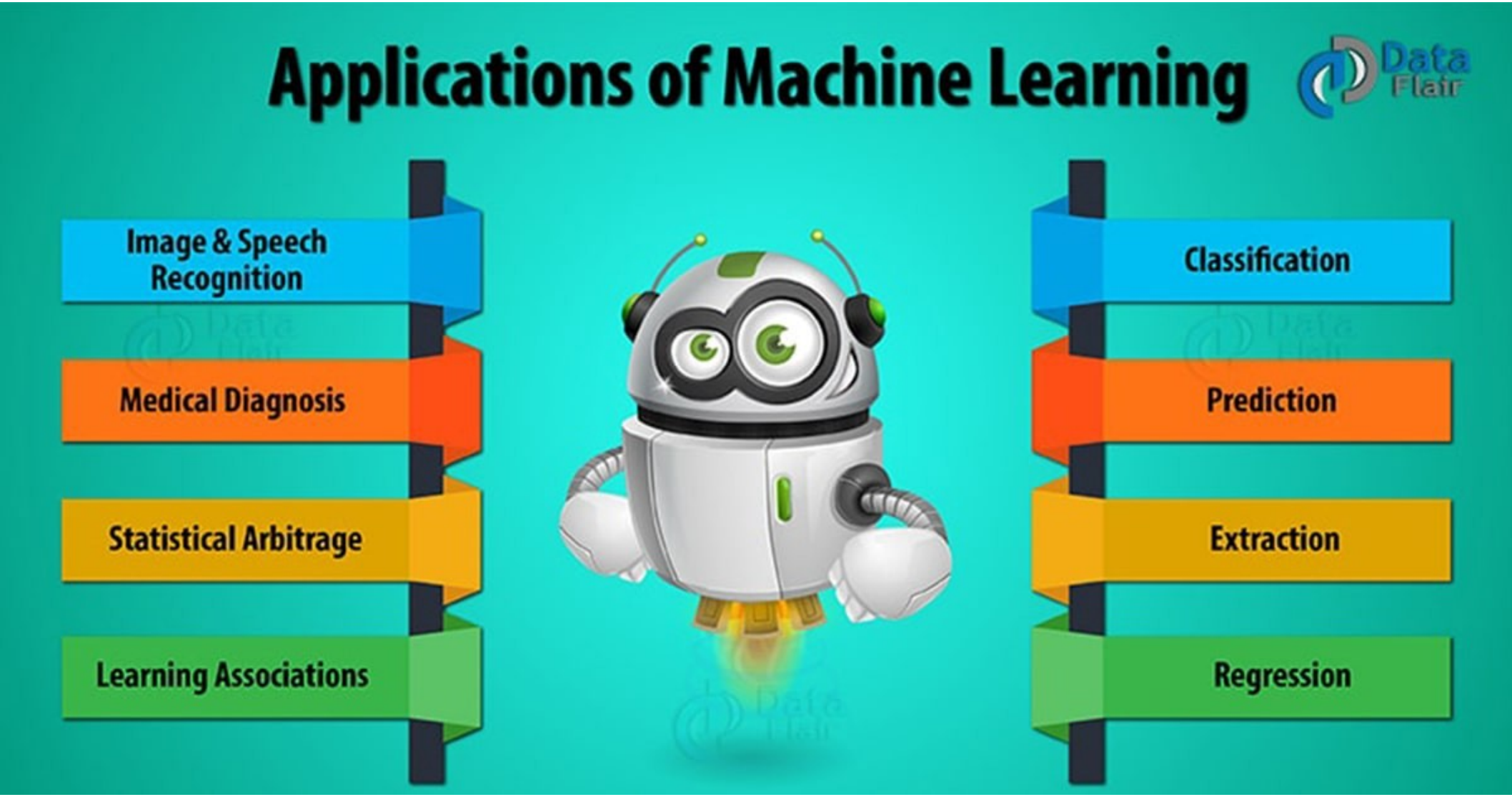
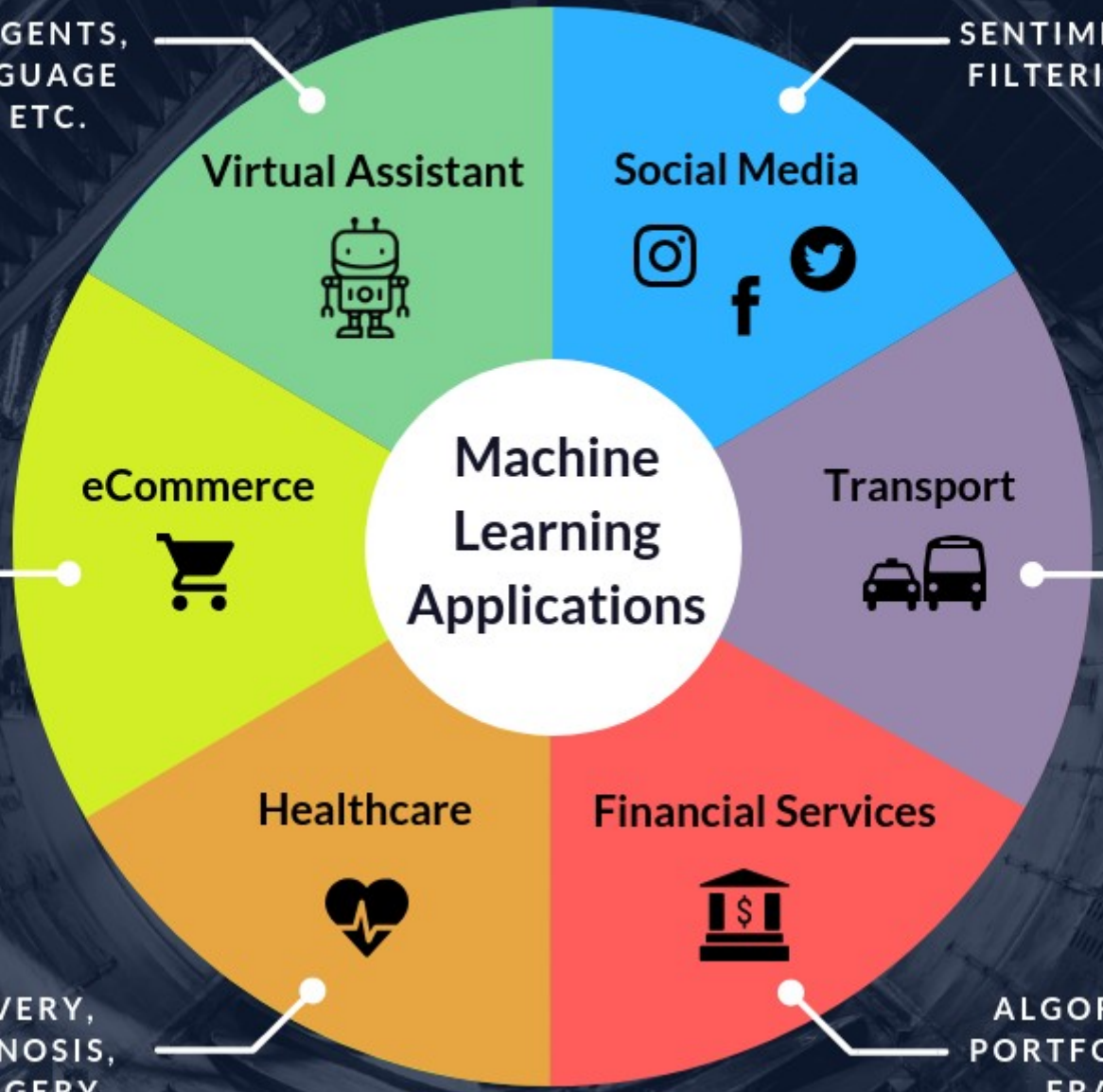

Machine Learning

Applications of Machine Learning



APPLICATIONS OF MACHINE LEARNING



INTELLIGENT AGENTS, NATURAL LANGUAGE PROCESSING ETC.

SENTIMENT ANALYSIS, FILTERING SPAM ETC.

CUSTOMER SUPPORT, PRODUCT RECOMMENDATION , ADVERTISING,

SAFETY MONITORING, AIR TRAFFIC CONTROL ETC.

DRUG DISCOVERY, DISEASE DIAGNOSIS, ROBOTIC SURGERY

ALGORITHMIC TRADING, PORTFOLIO MANAGEMENT, FRAUD DETECTION

Machine Learning in Healthcare

Diseases Identification & Diagnosis

Personalized Medicine/ Treatment

Drug Discovery & Manufacturing

Smart Health Records

Medical Imaging

Diseases Prediction

Data Science

- Need of entire analytics universe
- Branch that deals with data
- Different operations related to data i.e.
  - Data Gathering
  - Data Cleaning
  - Data Subsetting
  - Data Manipulation
  - Data Insights [Data Mining]

Machine Learning

- Combination of Machine and Data Science
- Machines utilize Data Science techniques to learn about the data hence called as Machine Learning
- Model Building, Model Evaluation and Validation
- 3 Types:
  - Unsupervised Learning
  - Reinforcement Learning
  - Supervised Learning
- Most popular tools are Python, R and SAS

Deep Learning

- Specific branch of Machine Learning that deals with different flavours of Neural Network
- Examples
  - Simple Neural Network
  - Convolutional Neural Network
  - Recurrent Neural Network
  - Long Short Term Memory
- Mainly utilized in..
  - Object detection in Image and Video
  - Speech Recognition
  - Natural Language Processing and Understandings

Artificial Intelligence

- Big Umbrella
- Empowering machines to take decisions on their own
- As the name suggest imparting humans' natural intelligence in machines
- Thus machines have ability to understand and react according to the situation

1 Introduction

1.1 What is Machine Learning

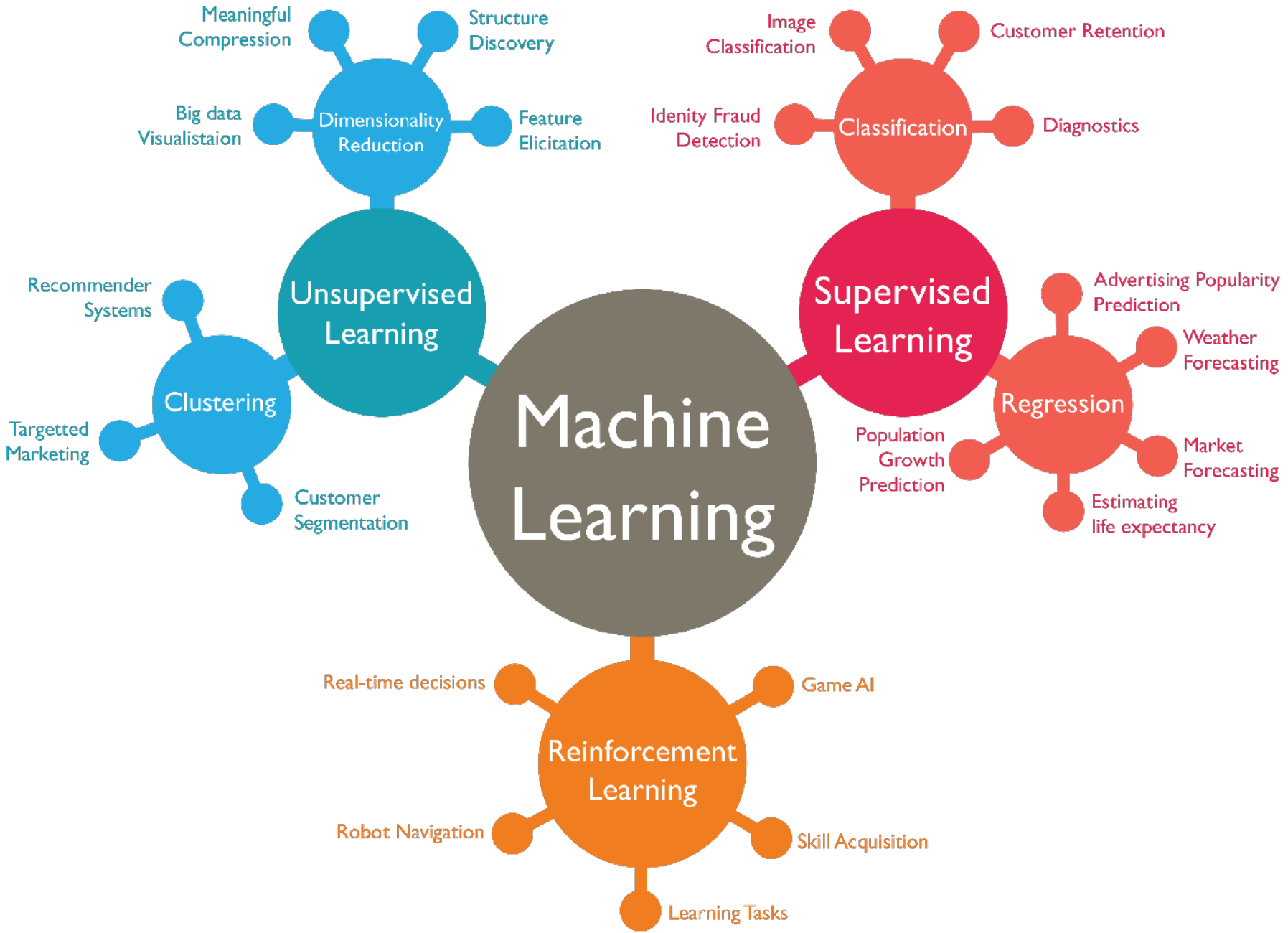
Machine learning is a subfield of computer science that is concerned with building algorithms which, to be useful, rely on a collection of examples of some phenomenon. These examples can come from nature, be handcrafted by humans or generated by another algorithm.

Machine learning can also be defined as the process of solving a practical problem by 1) gathering a dataset, and 2) algorithmically building a statistical model based on that dataset. That statistical model is assumed to be used somehow to solve the practical problem.

To save keystrokes, I use the terms "learning" and "machine learning" interchangeably.

1.2 Types of Learning

Learning can be supervised, semi-supervised, unsupervised and reinforcement.



1.2.1 Supervised Learning

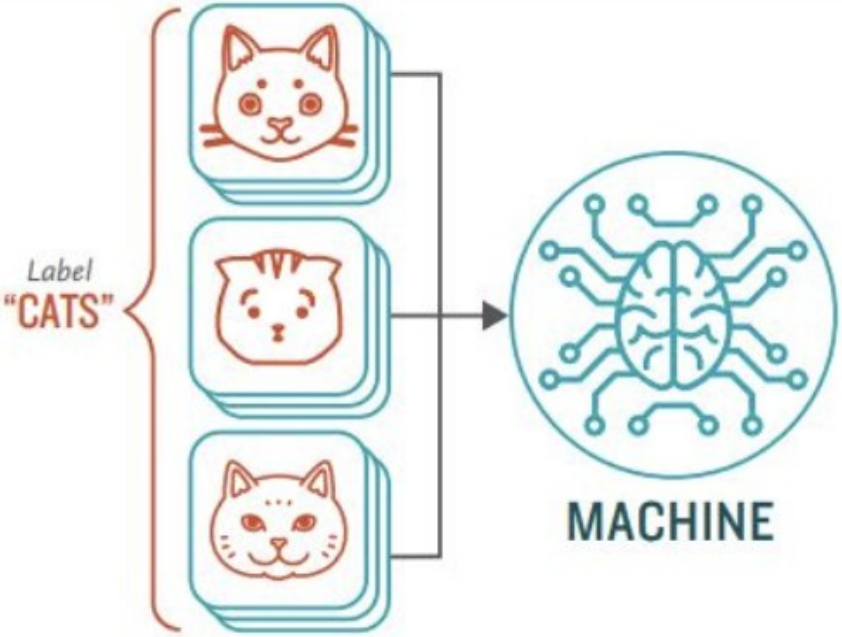
In **supervised learning**[1], the **dataset** is the collection of **labeled examples** $\{(\mathbf{x}_i, y_i)\}_{i=1}^{N}$. Each element $\mathbf{x}_i$ among N is called a **feature vector**. A feature vector is a vector in which each dimension $j = 1, \ldots, D$ contains a value that describes the example somehow. That value is called a **feature** and is denoted as $x^{(j)}$. For instance, if each example $\mathbf{x}$ in our collection represents a person, then the first feature, $x^{(1)}$, could contain height in cm, the second feature, $x^{(2)}$, could contain weight in kg, $x^{(3)}$ could contain gender, and so on. For all examples in the dataset, the feature at position j in the feature vector always contains the same kind of information. It means that if $x_i^{(2)}$ contains weight in kg in some example $\mathbf{x}_i$, then $x_k^{(2)}$ will also contain weight in kg in every example $\mathbf{x}_k$, $k = 1, \ldots, N$. The **label** y_i can be either an element belonging to a finite set of **classes** $\{1, 2, \ldots, C\}$, or a real number, or a more complex structure, like a vector, a matrix, a tree, or a graph. Unless otherwise stated, in this book y_i is either one of a finite set of classes or a real number. You can see a class as a category to which an example belongs. For instance, if your examples are email messages and your problem is spam detection, then you have two classes $\{spam, not\_spam\}$.

The goal of a **supervised learning algorithm** is to use the dataset to produce a **model** that takes a feature vector $\mathbf{x}$ as input and outputs information that allows deducing the label for this feature vector. For instance, the model created using the dataset of people could take as input a feature vector describing a person and output a probability that the person has cancer.

How **Supervised** Machine Learning Works

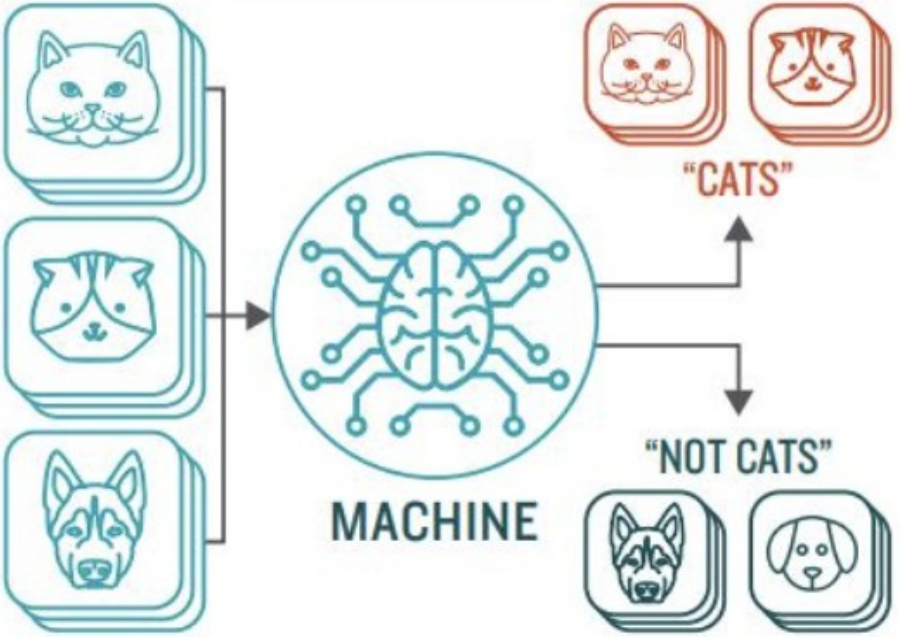
STEP 1

Provide the machine learning algorithm categorized or "labeled" input and output data from to learn

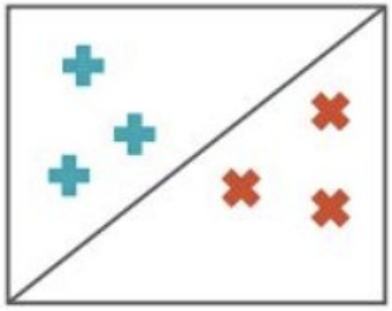


STEP 2

Feed the machine new, unlabeled information to see if it tags new data appropriately. If not, continue refining the algorithm

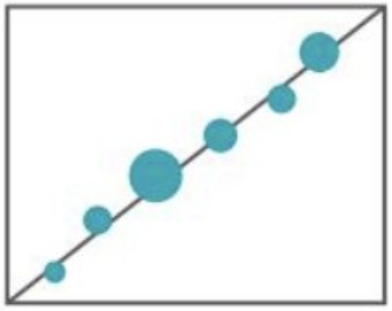


TYPES OF PROBLEMS TO WHICH IT'S SUITED



CLASSIFICATION

Sorting items into categories

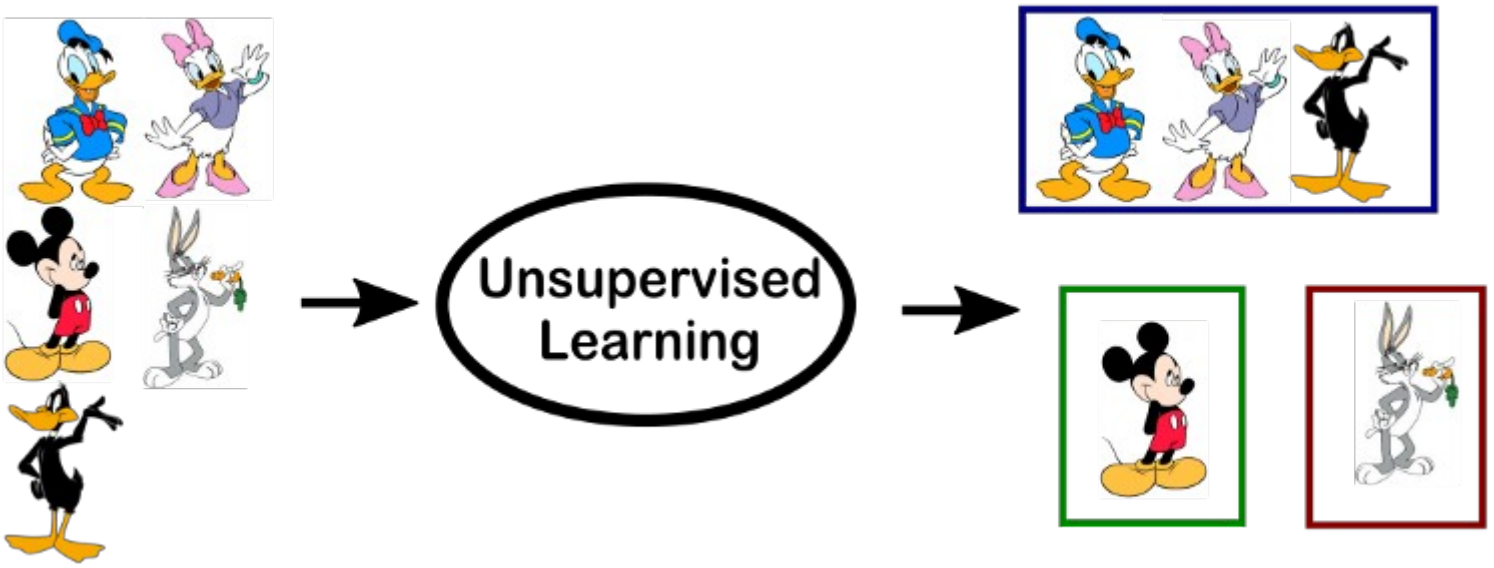


REGRESSION

Identifying real values (dollars, weight, etc.)

1.2.2 Unsupervised Learning

In **unsupervised learning**, the dataset is a collection of **unlabeled examples** $\{\mathbf{x}_i\}_{i=1}^N$. Again, $\mathbf{x}$ is a feature vector, and the goal of an **unsupervised learning algorithm** is to create a **model** that takes a feature vector $\mathbf{x}$ as input and either transforms it into another vector or into a value that can be used to solve a practical problem. For example, in **clustering**, the model returns the id of the cluster for each feature vector in the dataset. In **dimensionality reduction**, the output of the model is a feature vector that has fewer features than the input $\mathbf{x}$; in **outlier detection**, the output is a real number that indicates how $\mathbf{x}$ is different from a "typical" example in the dataset.



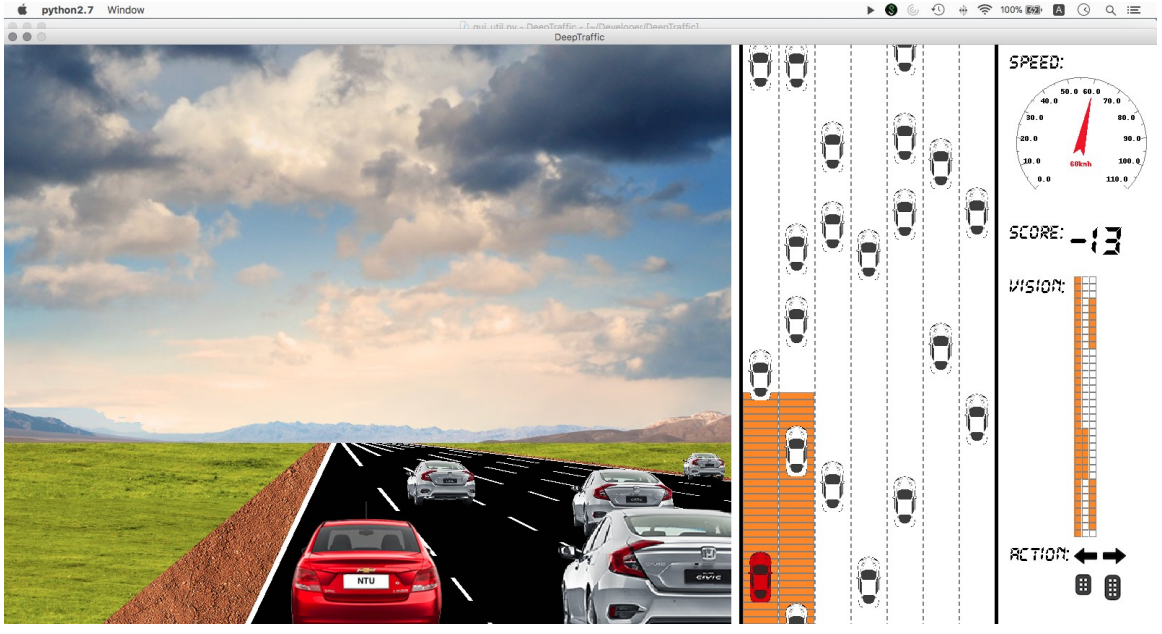
1.2.3 Semi-Supervised Learning

In **semi-supervised learning**, the dataset contains both labeled and unlabeled examples. Usually, the quantity of unlabeled examples is much higher than the number of labeled examples. The goal of a **semi-supervised learning algorithm** is the same as the goal of the supervised learning algorithm. The hope here is that using many unlabeled examples can help the learning algorithm to find (we might say "produce" or "compute") a better model .

1.2.4 Reinforcement Learning

Reinforcement learning is a subfield of machine learning where the machine "lives" in an environment and is capable of perceiving the **state** of that environment as a vector of features. The machine can execute **actions** in every state. Different actions bring different **rewards** and could also move the machine to another state of the environment. The goal of a reinforcement learning algorithm is to learn a **policy**. A policy is a function f (similar to the model in supervised learning) that takes the feature vector of a state as input and outputs an optimal action to execute in that state. The action is optimal if it maximizes the **expected average reward**.

Reinforcement learning solves a particular kind of problems where decision making is sequential, and the goal is long-term, such as game playing, robotics, resource management, or logistics.



1.3 How Supervised Learning Works

In this section, I briefly explain how supervised learning works so that you have the picture of the whole process before we go into detail. I decided to use supervised learning as an example because it's the type of machine learning most frequently used in practice.

The supervised learning process starts with gathering the data. The data for supervised learning is a collection of pairs (input, output). Input could be anything, for example, email messages, pictures, or sensor measurements. Outputs are usually real numbers, or labels (e.g. "spam", "not_spam", "cat", "dog", "mouse", etc). In some cases, outputs are vectors (e.g., four coordinates of the rectangle around a person on the picture), sequences (e.g. ["adjective", "adjective", "noun"] for the input "big beautiful car"), or have some other structure.

Let's say the problem that you want to solve using supervised learning is spam detection. You gather the data, for example, 10,000 email messages, each with a label either "spam" or "not_spam" (you could add those labels manually or pay someone to do that for us). Now, you have to convert each email message into a feature vector.

The data analyst decides, based on their experience, how to convert a real-world entity, such as an email message, into a feature vector. One common way to convert a text into a feature vector, called **bag of words**, is to take a dictionary of English words (let's say it contains 20,000 alphabetically sorted words) and stipulate that in our feature vector:

- the first feature is equal to 1 if the email message contains the word "a"; otherwise, this feature is 0;
- the second feature is equal to 1 if the email message contains the word "aaron"; otherwise, this feature equals 0;
- ...
- the feature at position 20,000 is equal to 1 if the email message contains the word "zulu"; otherwise, this feature is equal to 0.

You repeat the above procedure for every email message in our collection, which gives us 10,000 feature vectors (each vector having the dimensionality of 20,000) and a label ("spam"/"not_spam").

Now you have a machine-readable input data, but the output labels are still in the form of human-readable text. Some learning algorithms require transforming labels into numbers. For example, some algorithms require numbers like 0 (to represent the label "not_spam") and 1 (to represent the label "spam"). The algorithm I use to illustrate supervised learning is called **Support Vector Machine** (SVM). This algorithm requires that the positive label (in our case it's "spam") has the numeric value of +1 (one), and the negative label ("not_spam") has the value of −1 (minus one).

At this point, you have a **dataset** and a **learning algorithm**, so you are ready to apply the learning algorithm to the dataset to get the **model**.

SVM sees every feature vector as a point in a high-dimensional space (in our case, space is 20,000-dimensional). The algorithm puts all feature vectors on an imaginary 20,000-dimensional plot and draws an imaginary 20,000-dimensional line (a *hyperplane*) that separates examples with positive labels from examples with negative labels. In machine learning, the boundary separating the examples of different classes is called the **decision boundary**.

The equation of the hyperplane is given by two **parameters**, a real-valued vector $\mathbf{w}$ of the same dimensionality as our input feature vector $\mathbf{x}$, and a real number b like this:

$$\mathbf{wx} - b = 0,$$

where the expression $\mathbf{wx}$ means $w^{(1)}x^{(1)} + w^{(2)}x^{(2)} + \ldots + w^{(D)}x^{(D)}$, and D is the number of dimensions of the feature vector $\mathbf{x}$.

Now, the predicted label for some input feature vector $\mathbf{x}$ is given like this:

$$y = \text{sign}(\mathbf{w}\mathbf{x} - b),$$

where sign is a mathematical operator that takes any value as input and returns $+1$ if the input is a positive number or -1 if the input is a negative number.

The goal of the learning algorithm — SVM in this case — is to leverage the dataset and find the optimal values $\mathbf{w}^*$ and b^* for parameters $\mathbf{w}$ and b. Once the learning algorithm identifies these optimal values, the **model** $f(\mathbf{x})$ is then defined as:

$$f(\mathbf{x}) = \text{sign}(\mathbf{w}^*\mathbf{x} - b^*)$$

Therefore, to predict whether an email message is spam or not spam using an SVM model, you have to take a text of the message, convert it into a feature vector, then multiply this vector by $\mathbf{w}^*$, subtract b^* and take the sign of the result. This will give us the prediction ($+1$ means "spam", -1 means "not_spam").