Supervised,

Unsupervised,

Reinforcement learning

**Features** - Those columns which help you to predict the final result

**Label** - The column which contains the finally predicted data.

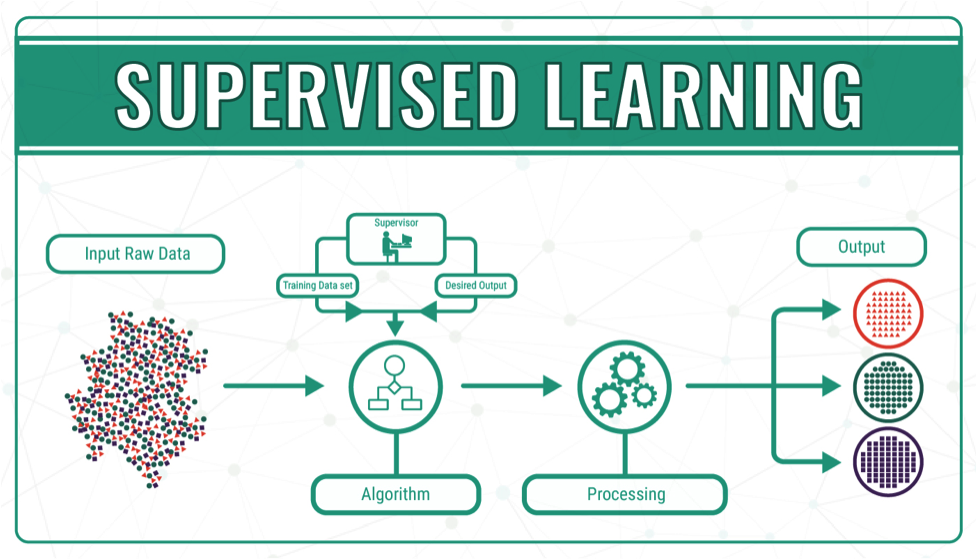
For example, here, we are trying to predict a flower name using its petal color, petal width, sepal width. Here, Petal width, Sepal width and petal color are **features** and Flower name is the **label**

| Petal width | Sepal width | Petal color | Flower name |
| --- | --- | --- | --- |
| 5 | 1.2 | red | Hibiscus |
| 8 | 1 | pink | Tulip |
| 12 | 2 | yellow | Sunflower |
| ... | ... | ... | ... |
| 14 | 3 | brown | ? |

**Supervised Machine Learning**

Most machine learning happens in a supervised way. It is the most popular type of machine learning. This type of machine learning is used in cases of data which has both features and labels. I.e, we know what we are going to predict. We already have the data of the label. For example, we have a table with columns midterm 1 test mark, midterm 2 test mark, final test mark, attendance, behaviour mark and grade, where grade is the label and rest are features. Here we feed on the features into the Machine learning model, it gives an output using an already existing function; Suppose the correct grade was 80 and it gave the output as 90. So, we correct the model and tell it t go down and suppose it goes to 75, and then to 82, then 79 and finally 80. Here **we supervised** the machine to correct each time and finally it found out the correct function which gives the best output. Now, it can be used for any data outside from our existing dataset and get a reliable output.

Note - In most cases, the machine has to correct itself thousands and thousands of times until it becomes so good that we can stop training.



**Unsupervised Machine Learning**

Here, You do not have a label, or to be exact, you do not know what you want to predict! Sounds strange right?

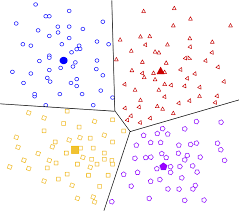
Think of this situation. You have certain tasks ahead of you and they have two features namely hardness and effectiveness

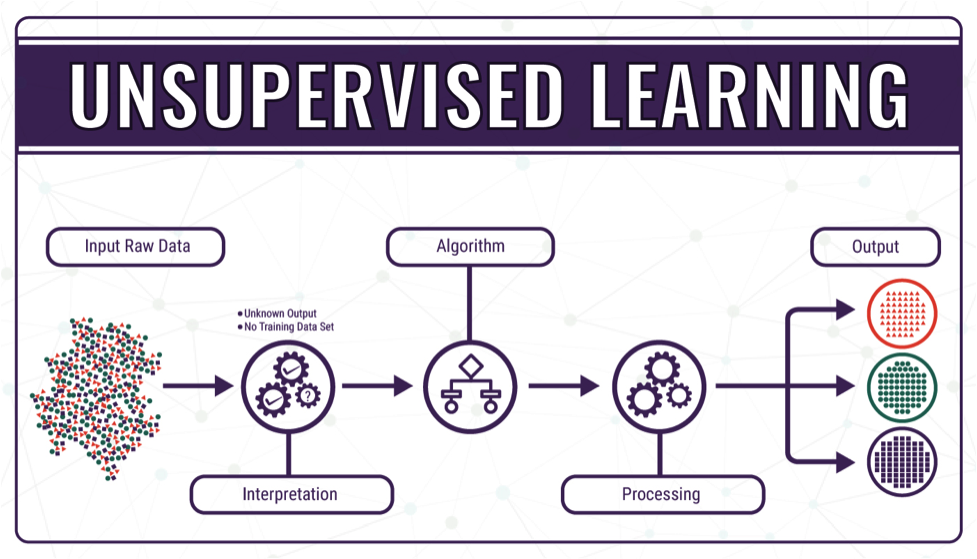
| Task name | Hardness (scale of 10) | Effectiveness( Scale of 10) |
| --- | --- | --- |
| Watch a cinema | 1 | 2 |
| Learn english | 5 | 6 |
| Say sorry to mom | 2 | 9 |
| Learn geography | 8 | 8 |
| ,,, | ,,, | ,,, |

Now, you want to ‘cluster’ these tasks into 4 groups -

1. Less hard, Less effective
2. Less hard, More effective
3. More hard, Less effective
4. More hard, More effective

So that you can do tasks in the order 2,4,1,3 .





**Reinforced Machine Learning**

Here, things are different. We are not talking about features and labels. Instead we are gonna learn some new terminologies. -

*Agent, Environment and Reward*

Lets understand these with a simple example. Lets say you are giving a machine a task to climb a hill. The machine steps in a random direction and for that, we give a *reward* (say points). Here, each time, the machine is not increasing the altitude, we give negative reward and the better it gets to the shortest route, the better its reward. Each time, the machine remembers the reward it got and tries to maximise the next reward. Here, the machine is the *agent* and The whole set up of the hill and surroundings where the agent plays is the *environment*. The crucial part is ‘how we can reward the agent’ in such a way that it reaches the goal in minimum time.

Fact: you can create a flappy bird model where you give a negative reward each time flappy hits the walls and by this way create an automated game where flappy wont hit any wall forever!

