MACHINE LEARNING

Machine learning is a branch of artificial intelligence (AI) and computer science which focuses on the use of data and algorithms to imitate the way that humans learn, gradually improving its accuracy.

TYPES OF MACHINE LEARNING

Supervised Learning: In supervised learning, the algorithm is trained on a labeled dataset, where the input data is paired with the corresponding correct output. The goal is for the algorithm to learn a mapping between inputs and outputs, so it can make accurate predictions or classifications on new, unseen data.

- **Classification**: The algorithm learns to assign inputs to a predefined set of classes or categories. Example applications include email spam detection and image classification.
- **Regression**: The algorithm learns to predict a continuous output value based

on input features. This is often used for tasks like predicting house prices or stock prices.

Unsupervised Learning: Unsupervised learning involves training on an unlabeled dataset, where the algorithm seeks to discover patterns, relationships, or structures within the data without any predefined outputs.

- Clustering: Algorithms group similar data points together into clusters. Common applications include customer segmentation and image segmentation.

Reinforcement Learning: Reinforcement learning involves training agents to take actions in an environment to maximize a cumulative reward. The agent learns through trial and error, receiving feedback in the form of rewards or penalties for its actions. This is

often used in game playing, robotics, and autonomous systems.

REGRESSION

Regression models (both linear and non-linear) are used for predicting a real value, like salary for example. If your independent variable is time, then you are forecasting future values, otherwise your model is predicting present but unknown values.

Types of regression:

1. Simple Linear Regression
2. Multiple Linear Regression
3. Polynomial Regression
4. Support Vector for Regression (SVR)
5. Decision Tree Regression
6. Random Forest Regression

LINEAR REGRESSION

Linear regression analysis is used to predict the value of a variable based on the value of another variable. The variable you want to predict is called the dependent variable. The variable you are using to predict the other variable's value is called the independent variable.

VARIANCE

In terms of linear regression, variance is a measure of how far observed values differ from the average of predicted values, i.e., their difference from the predicted value mean.

REGRESSION COEFFICIENT

The regression coefficients are a statically measure which is used to measure the average functional relationship between variables. In regression analysis, one variable is dependent and other is independent. Also, it measures the degree of dependence of one variable on the other(s).

MULTIPLE REGRESSION

Multiple regression is a statistical technique that can be used to analyze the relationship between a single dependent variable and several independent variables. The objective of multiple regression analysis is to use the independent variables whose values are known to predict the value of the single dependent value.

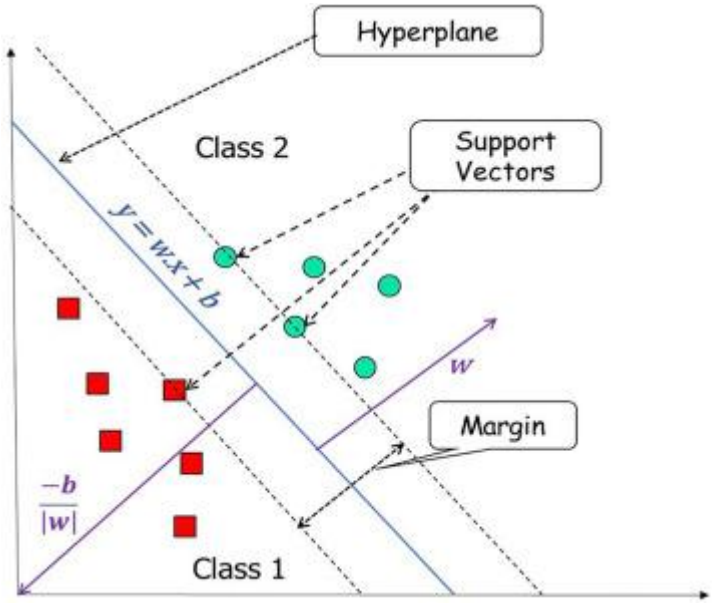
CONFUSION MATRIX

A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm.

- *True Positive* = Actually true & predicted true
- *True Negative* = Actually false & predicted false
- *False Positive* = Actually true but predicted false
- *False Negative* = Actually false but predicted true
- *Accuracy* = [(TP + TN) / (All)] , where all is => TP + TN + FP + FN
- *Error Rate* = 1 - *Accuracy*
- *Precision* = how often is the model right when it predicted +ve (predicted values) = [TP / (TP + FP)]

- *Recall or senstivity* = how often does the model predicted yes when it was actually yes (actual values) = [TP / (TP + FN)]
- *Fallout %age or False +ve rate* = actual total no values and usme se predicted false = [FP / (FP + TN)]
- *True -ve rate or specificity* = actual values no and usme se model predicted no = [TN / (FP + TN)]
- *F-1 score* = [(2 * Precision * Recall) / (Precision + Recall)]

SVM



Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane.

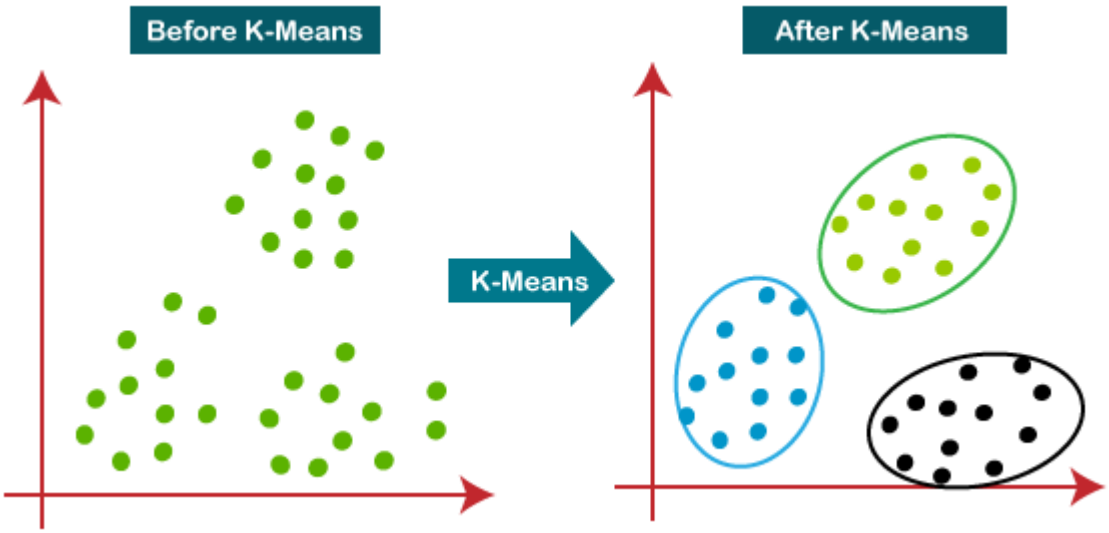
DECISION TREE

A decision tree is a decision support hierarchical model that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility.

- Supervised
- used for both classification (preferred) and Regression
- tree-structured classifier
- internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome
- DT has 2 nodes, Decision Node and Leaf Node
- DN => used to make any decision and have multiple branches
- LN => are the output of those decisions and do not contain any further branches

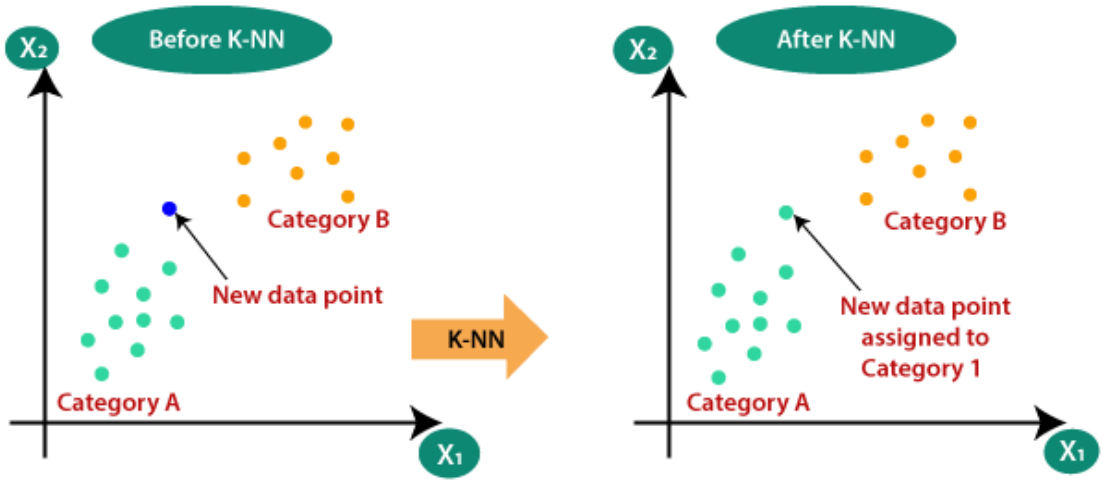
K MEANS CLUSTERING

K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process, as if K=2, there will be two clusters, and for K=3, there will be three clusters, and so on.



KNN

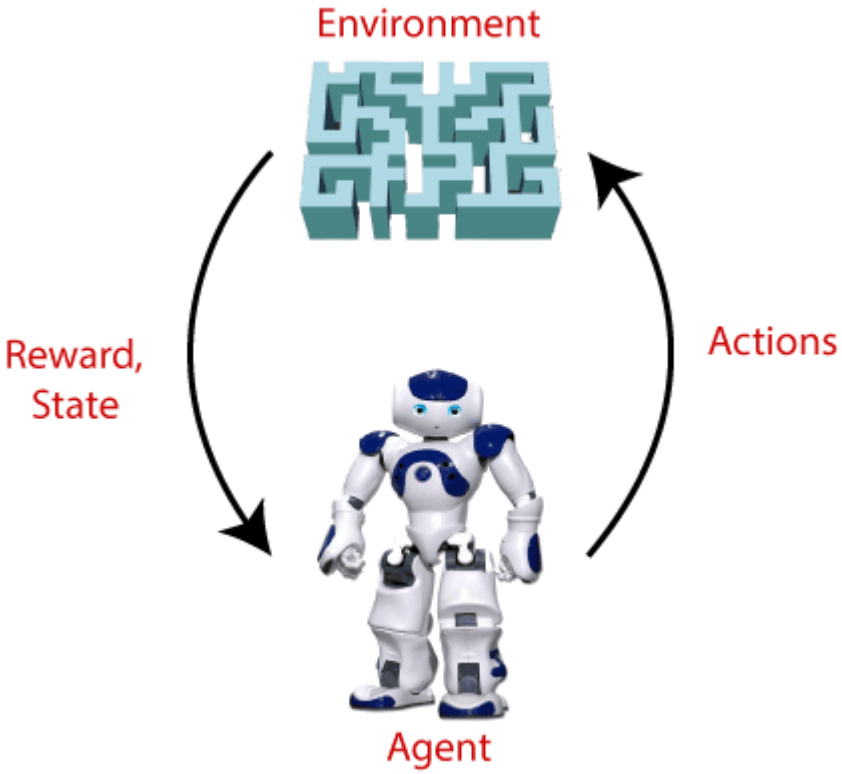
K-Nearest Neighbors (KNN) is a simple and widely used algorithm in machine learning, primarily for classification and regression tasks. It is a type of instance-based learning or lazy learning algorithm, meaning it doesn't explicitly learn a model from the training data. Instead, it makes predictions based on the proximity of data points in the feature space.



REINFORCEMENT LEARNING

Reinforcement Learning is a feedback-based Machine learning technique in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent

gets positive feedback, and for each bad action, the agent gets negative feedback or penalty.



CODE KMEANS in IRIS DATASET from SKLEARN

```
[2] from sklearn.datasets import load_iris
    import pandas as pd
    import numpy as np
    from sklearn.cluster import KMeans
    import matplotlib.pyplot as plt
    import matplotlib.patches as mpatches
    import sklearn.metrics as sm

    iris = load_iris()
    print(iris.data)
```

```
[16] iris.target

array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
       1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2,
       2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2])
```

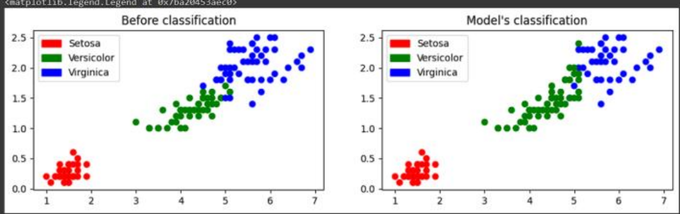
```
[7] x = pd.DataFrame(iris.data, columns=['Sepal Length', 'Sepal Width', 'Petal Length', 'Petal Width'])
    y = pd.DataFrame(iris.target, columns=['Target'])
```

```
[8] x
```

	Sepal Length	Sepal Width	Petal Length	Petal Width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2
...	...	...	...	...
145	6.7	3.0	5.2	2.3
146	6.3	2.5	5.0	1.9
147	6.5	3.0	5.2	2.0
148	6.2	3.4	5.4	2.3

```
[11] kmeans = KMeans(n_clusters = 3)

[13] KMmodel = kmeans.fit(iris.data)

/usr/local/lib/python3.10/dist-packages/sklearn/cluster/_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` ex
  warnings.warn(

[15] print(kmeans.labels_)

[1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
 0 0 0 2 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 2 0 2 2 2 2 0 2 2 2 2
 2 2 0 0 2 2 2 2 0 2 0 2 0 2 2 0 0 2 2 2 2 2 0 2 2 2 2 0 2 2 2 0 2 2 2 0 2
 2 0]

[17] print (kmeans.cluster_centers_)

[[5.9016129  2.7483871  4.39354839 1.43387097]
 [5.006      3.428      1.462      0.246     ]
 [6.85       3.07368421 5.74210526 2.07105263]]
```

```
plt.figure(figsize=(12,3))

colors = np.array(['red', 'green', 'blue'])

predictedY = np.choose(kmeans.labels_, [1, 0, 2]).astype(np.int64)

plt.subplot(1, 2, 1)
plt.scatter(x['Petal Length'], x['Petal Width'], c=colors[y['Target']])
plt.title('Before classification')
plt.legend(handles=[red_patch, green_patch, blue_patch])

plt.subplot(1, 2, 2)
plt.scatter(x['Petal Length'], x['Petal Width'], c=colors[predictedY])
plt.title("Model's classification")
plt.legend(handles=[red_patch, green_patch, blue_patch])

<matplotlib.legend.Legend at 0x7ba20453aec0>
```




```
[23] pd.crosstab(iris.target , KMmodel.labels_)
```

```
sm.confusion_matrix(predictedY, y['Target'])

array([[50,  0,  0],
       [ 0, 48, 14],
       [ 0,  2, 36]])
```