Machine learning:

It is a subset of AI that enables the machine to predict a solution without being explicitly programmed.

Why subset: In AI, we also need other engineering techniques like electrical & mechanical engineering.

Predict: it is based on probability and statistical models of maths.

Explicit Programmed: there is no direct formula to make prediction and it dynamically builds the logic based on given training datasets.

In normal programming:

We have a formula and input=➔output

In Machine learning:

We have input and optional output ➔build model ➔testing input➔predictive output

Applications of ML:

- ✓ Virtual Personal Assistants(Like Google assistant)
- ✓ Social Media Services
- ✓ Face book- 'People You May Know' and 'tag people'.
- ✓ Online Customer Support
- ✓ Chabot
- ✓ Product Recommendations
- ✓ Shopping platforms like Amazon and Jabong notice what products you look at and suggest similar products to you.
- ✓ Email Spam Filtering
- ✓ YouTube video recommendation
- ✓ Face detection
- ✓ Number plate recognition
- ✓ Stock price prediction

- ✓ News prediction
- ✓ Self driving car
- ✓ Etc.

Terms Used in machine Learning:

- ✓ Input/Features/Attributes/Dimensions refer to parameters (columns) of training dataset.
- ✓ Instance represents one sample of training dataset.
- ✓ Label/class/target refers output of training dataset.
- ✓ X is symbol for features and y is symbol for label

Example:

Age	**Salary**	**Purchased**
44	72000	No
27	48000	Yes
30	54000	No
38	61000	No
40	40000	Yes
35	58000	Yes
42	52000	No
48	79000	Yes
50	83000	No
37	67000	Yes

Here, Age and Salary are features(X) and Purchased is Label(y) and 44,72000 is one instance of features

Types of ML:

1. **Supervise learning:**
   Features and labels are given, further types are
   a. Classification
   b. Regression
2. **Unsupervised learning:**
   Only features are given, type is
   Clustering

3. **Reinforcement learning:**

No features and labels are given initially

Reward or feedback is given to an action, positive or negative moves become experience.

Move →becomes features

Reward/feedback->become label

Example: chess game

4. **Semi supervise learning:**

Combination of supervise and unsupervised learning, means in sample dataset some features are given with their labels and some features are given without labels.

Steps in Machine Learning:

1. Collect data
2. Prepare or clean data
3. Initialize a model/classifier/Algo
4. Train the model
5. Make prediction
6. Evaluation

Python Implementation of ML:

There are following two popular ML libraries in python:

1. Scikit-learn
2. Tensor flow (aka deep learning)

Scikit-learn:

It was developed by Google in 2007 and publically available from 2010 as open source library.

You can download Scikit learn by writing following command

cmd>*pip install scikit-learn*

or

If you are using anaconda IDE, it is by default bundled.

NOTE-1: *This library takes features in numeric form so we must convert all features into numbers before passing data to training.*

NOTE-2*: After converting features into numbers we must represents these features in 2d array or nested list. In the same way label must be represented as 1d array or list.*

Example-1: Consider following dataset of online purchased

Age	**Salary**	**Purchased**
44	72000	No
27	48000	Yes
30	54000	No
38	61000	No
40	40000	Yes
35	58000	Yes
42	52000	No
48	79000	Yes
50	83000	No
37	67000	Yes

By observing above dataset we may conclude that given example belongs to **classification problem of supervise learning.**

Features: Age, Salary

Label or Target: Purchased

Now,Represents these features as 2d array and label as 1d array

data=[[44,72000], [27,48000],[30,540000], [38,61000], [40,40000], [35,58000], [42,52000], [48,79000], [50,83000], [37,67000]]

label=[0,1,0,0,1,1,0,1,0,1]

Code:

```
from sklearn.neighbors import KNeighborsClassifier

knn=KNeighborsClassifier()

knn.fit(data,label)

knn.predict([[32,20000]])
```

NOTE: Most of the times we do not have cleaned or prepared dataset so we must prepare dataset before passing it to training.

Data Preprocessing:

- ✓ Identify missing values in features
- ✓ Fill these missing values by using mean or other approach
- ✓ You may also drop a row if most of the values are missing
- ✓ Sometimes not all features are required in training ,remove those features
- ✓ Converting features and label into numeric values so that dataset can be passed to training.
- ✓ Do feature Scaling if values of features are not in same range.

Feature Scaling:

It is technique to convert all features values within a range so that model may perform well.

- MinMaxScalar:
    - minrange+(maxrange-minrange)*(xi-xmin)/(xmax-xmin)
- Standard Scalar:
    - (xi-xmean)/std_of_feature
- Normalizer:
    - it works with row ,xi/np.sqrt(sum_of_square_each_element_in row)
- Binarizer:
    - all values above threshold will be 1 and less or same will be 0
- MaxAbsScaler:
    - xi/abs(xmax)

Performance Metrics(Parameters) for classification:

First obtain confusion matrix:

`sklearn.metrics.`**`confusion_matrix`**(*y_true*, *y_pred*)

	Predicted: 0	Predicted: 1
Actual: 0	TN	FP
Actual: 1	FN	TP

Terms associated with Confusion matrix:

1. **True Positives (TP):** True positives are the cases when the actual class of the data point was 1(True) and the predicted is also 1(True)

2. True Negatives (TN): True negatives are the cases when the actual class of the data point was 0(False) and the predicted is also 0(False

3. False Positives (FP): False positives are the cases when the actual class of the data point was 0(False) and the predicted is 1(True). False is because the model has predicted incorrectly and positive because the class predicted was a positive one. (1)

4. False Negatives (FN): False negatives are the cases when the actual class of the data point was 1(True) and the predicted is 0(False). False is because the model has predicted incorrectly and negative because the class predicted was a negative one. (0)

Accuracy *measures how well the test predicts both True and Negative classes.(Overall correctness of model)*

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN}$$

Sensitivity or Recall ***or True Positive Rate*** *measures the proportion of positives that are correctly identified as such (Accuracy of class 1)*

$$\text{Recall} = \frac{TP}{TP + FN}$$

Specificity *measures the proportion of negatives that are correctly identified as such. (Accuracy of class 0)*

$$\text{Specificity} = \frac{TN}{TN + FP}$$

Precision *is intuitively the ability of the classifier not to label as positive a sample that is negative.(How Many predicted 1 are actually 1)*

$$\text{Precision} = \frac{TP}{TP + FP}$$

FalsePositive Rate:

The **false positive rate** is the proportion of all negatives that still yield **positive** test outcomes.

False Positive Rate (FPR) = FP / (FP + TN)

F-1 Score:

If we have immbalanced data like in titanic we have majority of sample belonging to 0 class.

Or suppose consider:

100 samples(instances)-→class 0
20 samples(instances)-→class 1

The above data is immbalanced.

So ,with immbalanced data we should test model performance by using F-1 score.

```
F1 = 2 * (precision * recall) / (precision + recall)
```

Underfitting:

We train the model with given traing data set and model is not performning well if we test this model with same dataset.

Ex: A player does not perform well on the same playground where he was trained.

Overfitting:

We train the model with given traing data set and model is performning well if we test this model with same dataset but not performing well with different(unseen) dataset.

Ex:

A player performs well on the same playground where he was trained but does not perform well on other playground.

Bias:

high bias--->Underfitting
unbias------>Overfitting

Regularization:

>it solves overfitting problem by adding some bias(penalty) in model and also reduces coef.

>Ridge & Lasso are two techniques for regulerization
Ridge(a.k.a. L2)
Lasso(a.k.a L1)

Variance:

variability(spread) of prediction of testing sample over different subset of a dataset.

we should try to keep low variance

Ideal Statement for a Model:

low bias and low variance

Logistic Regression:

>it is a model to solve classification problem
>generally we use this model in binary classification
>it internally uses linear regression and a probability function to predict a class.

Cross Validation:

In train_test_split,testing data is never used in training,and this testing data might effect performance of model so cross validation uses multiple folds to check model performance (each fold has different testing data set)and returns performance of each fold.Bydefault it uses 3 folds, but generally we should 10 folds or based on datasize,hence K-fold.

```
From sklearn.model_selection import cross_val_score
1darray=cross_val_score(model,X,y,cv=10)
```

We may use cross validation in:

- parameter tuning
- fetaures selection
- model selection