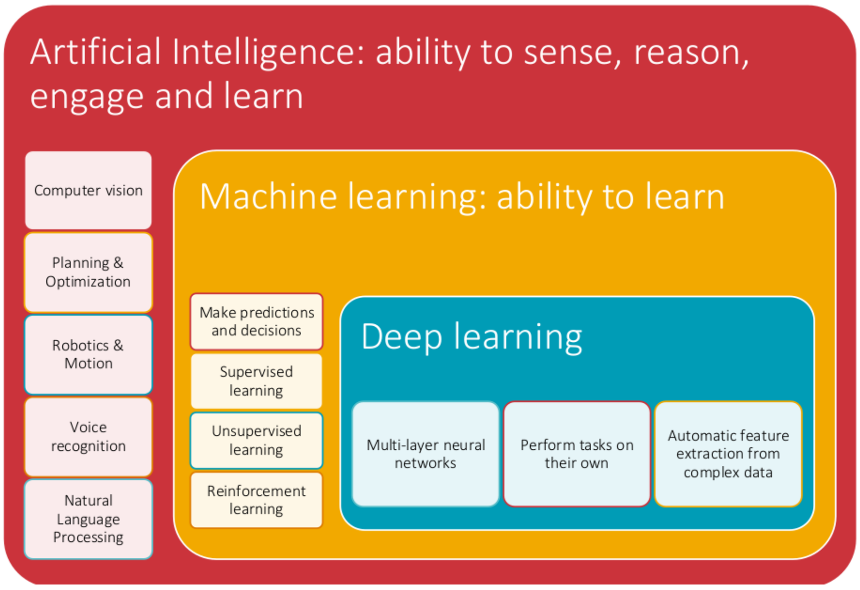
DTE-2502 Neural Networks: wk02L01

Introduction to AI



Artificial Intelligence: ability to sense, reason, engage and learn
Computer vision
Planning & Optimization
Robotics & Motion
Voice recognition
Natural Language Processing
Machine learning: ability to learn
Make predictions and decisions
Supervised learning
Unsupervised learning
Reinforcement learning
Deep learning
Multi-layer neural networks
Perform tasks on their own
Automatic feature extraction from complex data

Artificial Intelligence (AI) deals with developing *systems* that are capable for performing tasks that require human level of intelligence.

Machine learning (ML) deals with algorithms that are capable of learning from the data provided to recognize patterns make prediction from them.

Deep learning is a sub-class of ML algorithms that deal with developing model based on *neural networks*, which are programmable block that are designed to imitate biological neurons.

Broadly speaking Deep learning is a subset of ML which is subset of AI.

There are various fields of research within AI for example

<u>Computer vision:</u> Object recognition, Face recognition, AI generated art etc.

<u>Resource planning and optimization</u>: Movie/product recommendation, delivery routing etc

<u>Robotics and motion:</u> Autonomous driving, drones, robot janitor etc

<u>Voice recognition</u>: Siri, Alexa etc

<u>Natural language processing</u>: ChatGPT, Large language models etc

The are various problem that the researchers in field of ML would like to tackle:

- *Supervised learning:* In this learning paradigm is driven by learning by example. A *labeled dataset*, typically labeled by an external supervisor or an algorithm/program. The labeled data would have the input to the task and output data provided to learning algorithm called a *model*. Thus the model is guided from the input towards the output hence called supervised learning.

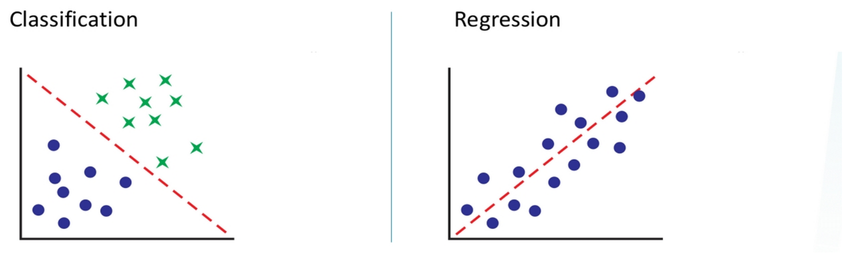
- *Unsupervised learning:* In this paradigm no explicit labels are provide to the model. The model tries to identify any patterns and relationships within the given raw dataset by grouping data into clusters or by association.

- *Reinforcement learning*: In this approach *an agent* (often a robot or device) interacts with the *environment* (scenario or region of operations). Through these interaction it learns to take actions in order to accomplish the end goal. Often the key learnings from interactions are are stored as (*state*, *action*) pair which are navigated towards the end goal by maximizing long-term *reward*. The reward can be seen as a mathematical function awarding points to successful state-action pairs while penalizing the unsuccessful ones. As the agent takes actions it needs to maintain a balance between *exploration* (taking more risky state-action pairs in the hope of better reward) and *exploitation* (taking a know state-action pair with know best reward) by performing a variety of actions using trial and error to favor the actions that yield the maximum reward in the future.

- *Semi-Supervised Learning*: This approach training model on a small number of labeled data and uses a large unlabeled dataset to capture the shape of the underlying data distribution and generalize better to new samples. This technique automates the data-labeling process with only a small labeled dataset and tries to combines the best of both supervised and unsupervised learning.

Supervised learning: Regression vs Classification

Two commonly attempted task in supervised learning as regression and classification



Regression

- Find the best fit curve, which can predict the output more accurately
- Output variable is continuous nature or real value.
- Eg: Weather Prediction, Stock price prediction etc
- Example labeled data for regression

Hour of the day	Temperature in ºC
01	26.704
02	27.434
03	28.101
04	26.140
05	25.427

The objective can be to predict temperature in 06th hour

Classification

- Find the best decision boundary, which can separate different classes in data.
- Output variable is discrete value.
- Eg: Spam emails, Face recognition etc
- Example labeled data for classification

Index	Sepal length	Sepal width	Petal length	Petal width	Flower name
1	5.1	3.5	1.4	0.2	*Setosa*
2	4.9	3.0	1.4	0.2	*Setosa*
3	4.7	3.2	1.3	0.2	*Setosa*
4	7.0	3.2	4.7	1.4	*Versicolor*
5	6.4	3.2	4.5	1.5	*Versicolor*
6	6.9	3.1	4.9	1.5	*Versicolor*
7	6.3	3.3	6.0	2.5	*Virginica*
8	6.3	3.3	6.0	2.5	*Virginica*
9	6.3	3.3	6.0	2.5	*Virginica*

(https://en.wikipedia.org/wiki/Iris_flower_data_set)

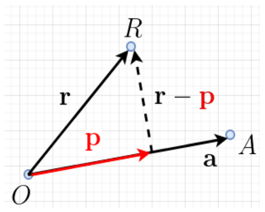
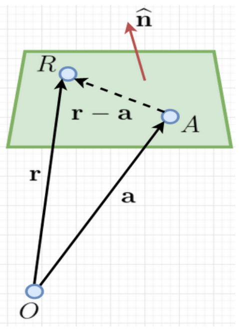
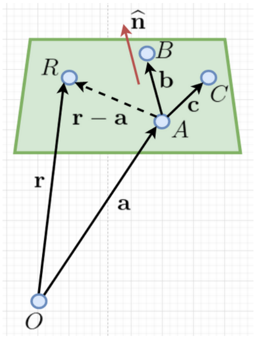
A more formal definitions are as follows:

Given a training set $X^l = (x_i, y_i)_{i=1}^l$, objects $x_i \in \mathbb{R}^n$, responses y_i

Regression	Classification
• $Y = \mathbb{R}, y_i \in Y$ • $a(x,w) = \sigma(\langle w, x_i \rangle)$ • $Q(w; X^l) = \sum_{i=1}^l (\sigma\langle w, x_i \rangle - y_i)^2 \rightarrow \min\limits_w$	• $Y = \{\pm 1\}, y_i \in Y$ • $a(x,w) = \text{sign}(\langle w, x_i \rangle)$ • $Q(w; X^l) = \sum_{i=1}^l [\langle w, x_i \rangle y_i < 0] \rightarrow \min\limits_w$

Where the data the activation function and the loss minimization are represented by *X, Y, a(.)* and *Q(.)* respectively.

Linear algebra review

Projection operation	Vector equation of a plane
 $\mathbf{p} = (\mathbf{r} \cdot \mathbf{a})\mathbf{u}$ $\mathbf{u} = \dfrac{\mathbf{a}}{\lvert\mathbf{a}\rvert}$	  $(\mathbf{r} - \mathbf{a}) \cdot \hat{\mathbf{n}} = 0$ $\hat{\mathbf{n}} = \dfrac{\mathbf{b} \times \mathbf{c}}{\lvert\mathbf{b} \times \mathbf{c}\rvert}$

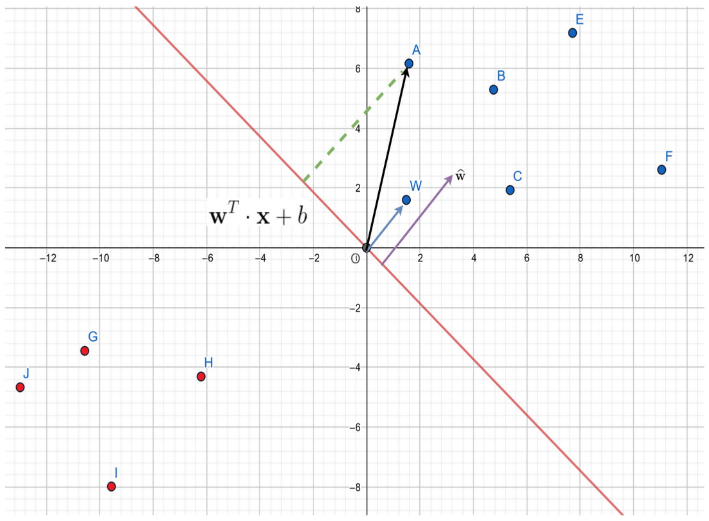
General equation of a hyperplane is $\mathbf{w}^T \cdot \mathbf{x} = 0$

Support vector machine as a classifier

- SVM is an algorithm that can learn a linear model from a set of labeled data available to train the model. Eg (classification, regression (Support Vector Regressor))
- We suppose that the data we want to classify can be separated into classes by a line
- We know that a line can be represented by the equation: $y = \mathbf{w}^T \cdot \mathbf{x} + b$
- We know that there is an infinity of possible lines obtained by changing the value of $\mathbf{w}$ and b
- We use an algorithm to determine which values of $\mathbf{w}$ and b give the "best" line separating the data.

Note: A linear system or operator *f(x)* is characterized by two properties:
- Additivity or superposition: $y_1 = f(x_1)$, $y_2 = f(x_2)$ then $f(x_1 + x_2) = y_1 + y_2$
- Scaling or homogeneity: $y = f(x)$, then $f(ax) = ay$ for any constant *a*. Verify that $y = \mathbf{w}^T \cdot \mathbf{x}$ is a linear operation

SVM algorithm

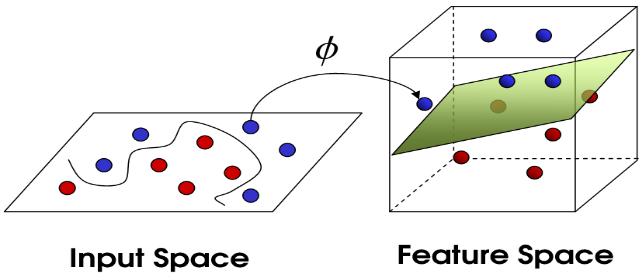


Consider a two class problem with red and blue dots as shown:

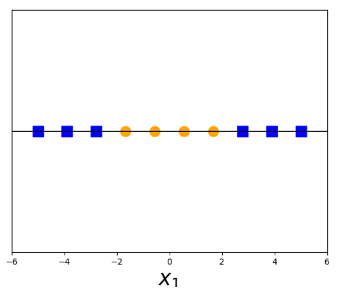
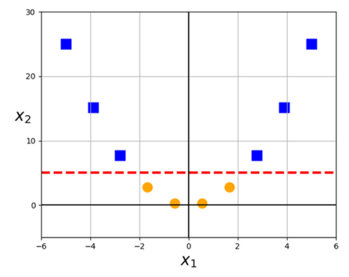
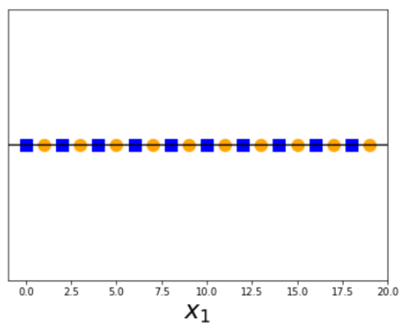
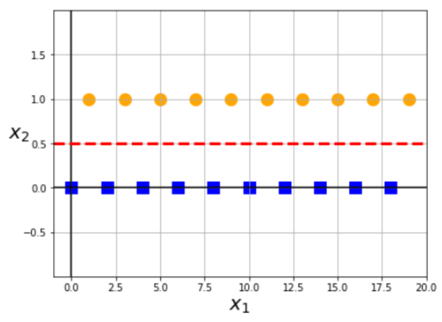
- The initial hyperplane is randomly initialized as the redline shown above.
- So we would like to find such a line that has maximum separation of the classes . The point closest to the line (boundary points) are the ones that are most likely to be misclassified. So the boundary point closest to the hyperplane should be maximally separated.
- As there a sense of distance involved in determining the boundary we can use the projection operation to define the distance to a point and hyperplane (for the point A in the figure) by:

$$|\mathbf{p}| = \widehat{\mathbf{w}} \cdot \mathbf{a}$$

Kernel trick for SVM



- If the data is not linearly separable in the original input, space then we apply transformations to the data, which map the data from the original space into a *higher dimensional feature space*.
- The goal is that after the transformation to the higher dimensional space, the classes are now linearly separable in this higher dimensional feature space.

Initial data	Kernel trick with higher features
	
	

Feature (x_1)	-5	-4	-3	-2	-1	1	2	3	4	5
Label	b	b	b	y	y	y	y	b	b	b

Feature (x_1)	-5	-4	-3	-2	-1	1	2	3	4	5
Feature (x_2=x_1^2)	25	16	9	4	1	1	4	9	16	25
Label	b	b	b	y	y	y	y	b	b	b

Feature (x_1)	0	1	2	3	4	5	6	7	8	….
Label	b	y	b	y	b	y	b	y	b	….

Feature (x_1)	0	1	2	3	4	5	6	7	8	….
Feature (x_2=$x_1$mod 2)	0	1	0	1	0	1	0	1	0	….
Label	b	y	b	y	b	y	b	y	b	….

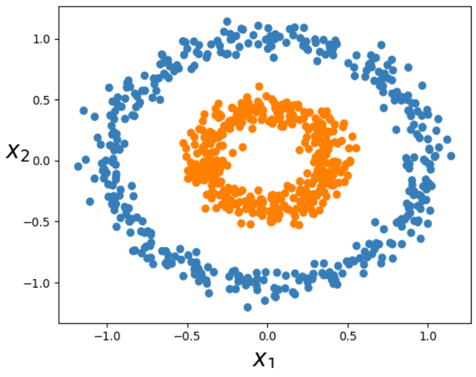
The above table shows some data points in 1D that are impossible to separate with linear hyperplane but by suitable projection into higher dimensions 2D they are separable by a linear hyperplane.

Kernel is defined as a function that acts on the input vectors in the original space and returns the dot product of the vectors in feature space.
Formally the kernel function *k(x,y)* = dot product of *ϕ(x), ϕ(y)* where *ϕ(.)* is the mapping function defined by the kernel *k*.

$$\mathbf{x},\mathbf{y} \in X \text{ and } \phi : X \to \mathbb{R}^n$$
$$\text{then } k(\mathbf{x},\mathbf{y}) = \langle \phi(\mathbf{x}), \phi(\mathbf{y}) \rangle$$

In the above examples the features are scalars hence the kernel mapping is also defined as a scalar function instead of a dot product.

Let us consider a more detailed example involving vectoral features in the following dataset



Here we have two features *(x_1, x_2)* defining the points marked as orange and blue. It is obvious to see that these points are impossible to separate using a linear hyperplane. But intuitively it is obvious that if we include the distance of the points from the origin as a third feature we can separate them out. Now let us define a formal mapping function for such a kernel as :

$$\phi(\mathbf{x}) = \phi\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} x_1^2 \\ \sqrt{2}x_1x_2 \\ x_2^2 \end{pmatrix}$$

Apply the kernel for any to vectors *k(***a**, **b***)* with result in:

$$\langle \phi(\mathbf{a}), \phi(\mathbf{b}) \rangle = \phi(\mathbf{a})^T \cdot \phi(\mathbf{b})$$

$$= \begin{pmatrix} a_1^2 & \sqrt{2}a_1a_2 & a_2^2 \end{pmatrix} \cdot \begin{pmatrix} b_1^2 \\ \sqrt{2}b_1b_2 \\ b_2^2 \end{pmatrix}$$

$$= (a_1^2b_1^2 + 2a_1b_1a_2b_2 + a_2^2b_2^2) = (a_1b_1 + a_2b_2)^2$$

$$= \left[\begin{pmatrix} a_1 & a_2 \end{pmatrix} \cdot \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right]^2 = \left[\mathbf{a}^T \cdot \mathbf{b} \right]^2$$

for **a=b** this is (**a**^{T}.**a**)^2 which is the distance of the vector **a** from the origin as intended.