LECTURE 1: INTRODUCTION

1. What is “nonlinear programming” (NLP)?
2. What are related problems?
3. How is NLP related to AI and machine learning?
4. What are essential to know about NLP?
5. What are the pre-requisites for this course?

What is nonlinear programming?

- Programming = ?
- Nonlinear = ?
- Nonlinear Programming (NLP) = ?

Unconstrained NLP problem

- Let $E^n = \{(x_1, x_2, \ldots, x_n) | \ x_i \in R\}$ be the n-dimensional Euclidian space (Real space)
- $f: E^n \rightarrow R$ be a real-valued continuous function, usually a twice continuously differentiable function ($f \in C^2$)
- $S \subset E^n$ is a "simple" set

- Unconstrained Problem in general form:
- $$Min \ f(x)$$
  $$s.t. \quad x \in S$$

Constrained NLP problem

- Let $E^n = \{(x_1, x_2, \dots, x_n) | \ x_i \in R\}$ be the n-dimensional Euclidian space (Real space)
- $f: E^n \rightarrow R$ be a real-valued continuous function, usually a twice continuously differentiable function ($f \in C^2$)
- $S \subset E^n$ is a "simple" set
- Constraint functions

$$h_i(\cdot), g_j(\cdot) \in C^2, i = 1,2, \dots, m, j = 1,2, \dots, p$$

- Constrained Problem in general form:
-

$$Min \ \ f(x)$$
$$s.t. \ \ h_i(x) = 0, \ \ i = 1,2, \dots, m$$
$$g_j(x) \leq 0, j = 1,2, \dots, p$$
$$x \in S$$

Constrained vs. unconstrained problems

- Unconstrained problems look easier than constrained problems.

- In a sense, a “constrained optimization" problem can be reformulated as an “unconstrained optimization" problem.

- Theory and algorithms of unconstrained NLP leads to that of constrained NLP.

Converting constrained NLP to unconstrained NLP?

Constrained Problem

$$\begin{aligned} \text{Min} \quad & f(x) \\ \text{s. t.} \quad & h_i(x) = 0, i = 1, 2, \cdots, m \\ & g_j(x) \leq 0, j = 1, 2, \cdots, p \\ & x \in S \subset E^n \end{aligned}$$

where $S \subset E^n$ is a "*simple set*"

Example

- Penalty method:

$$Min\ f(x) + \sum_{i=1}^{m} \lambda_i \|h(x)\|^2 + \sum_{j=1}^{p} \beta_j \times\ max\{g_j(x), 0\}$$

s.t. $x \in S$

for some big penalty parameters $\lambda_i > 0, \beta_j > 0$

Nonlinear vs. linear problems

- Linear problems look simpler than nonlinear problems.

- In a sense, “nonlinearity” is locally close to “linearity,” but it much better reflects reality.

- Theory and algorithms of linear programming may provide clues for nonlinear programming.

From nonlinear to linear ?

- Is this possible?

- How?

Example

- Taylor series expansion of a nonlinear function
  - for a given point $x^0 \in E^n$, the value of function $f \in C^\infty$ can be expressed and approximated as below

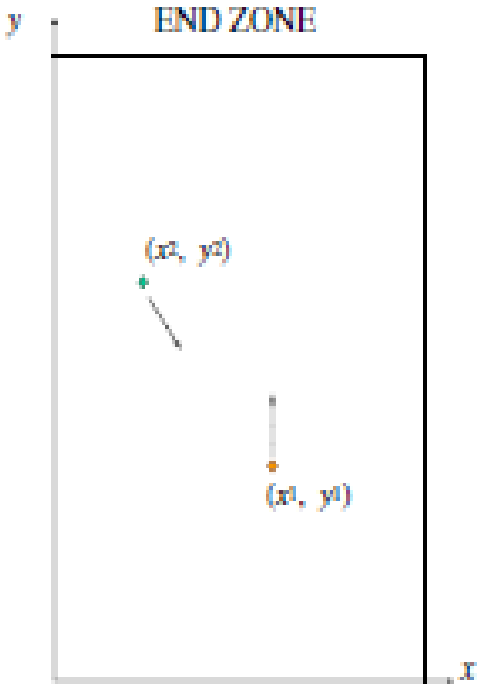
$$f(x) = f(x^0) + f'(x^0)(x - x^0) + \frac{1}{2} f''(x^0)(x - x^0)^2 + \dots$$
$$\approx f(x^0) + f'(x^0)(x - x^0)$$

as $x \approx x^0$.

Example

Example: Football Running Game (Touching Game)

- Assumption: Rivals are of "equal speed", "equal strength" and "equal intelligence".
- Goal: Touch the goal line.
- Constraint: Stay in the field.



Discrete vs. continuous problems

- Continuous problems look more familiar than discrete problems.

- In a sense, “discreteness” can be represented by “continuity,” but it indicates much stricter local information.

From discrete to continuous?

- Can integer-valued discrete variables be expressed as continuous variables and/or constraints?

Example

Example: 0-1 Integer Programming

$$x_i \in \{0, 1\} \quad \Leftrightarrow \quad x_i(x_i - 1) = 0$$

where $x_i \in R$

- How about $x_i \in \{2.5, 3.65, 100, \ldots\}$

Smooth vs. non-smooth problems

- Smooth (differentiable) problems look easier than non-smooth (non-differentiable) problems.

- “Smoothness” carries more local information and “non-smooth” problems may have a smooth approximation.

From non-smooth to smooth?

- Can the missing local information in a non-smooth function be recovered in a smooth function?

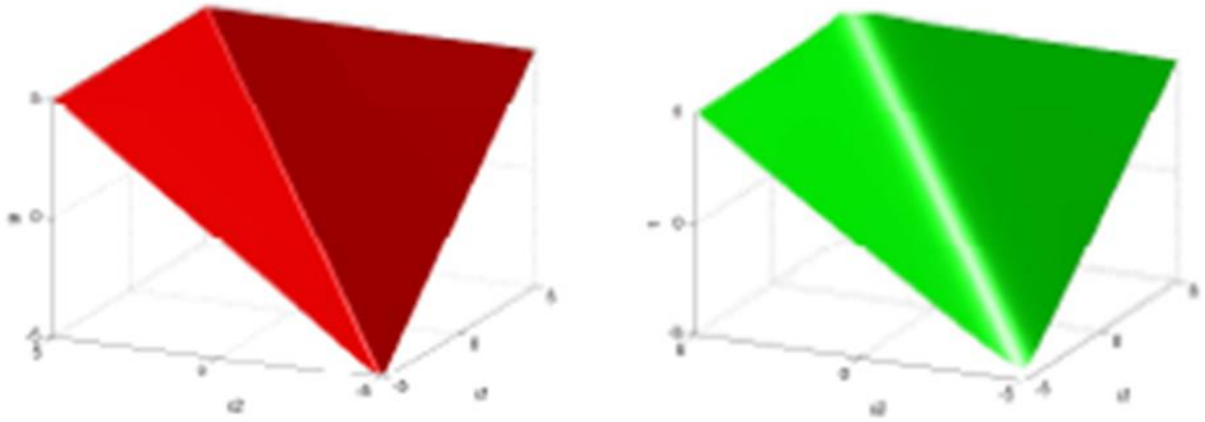
Example

Example: Nonsmoothing Functions

$$g(x) = \max\{x_1, x_2\}, \quad \forall x \in E^2$$

can be approximated by

$$f(x) = \frac{\sqrt{(x_1 - x_2)^2 + \varepsilon} + x_1 + x_2}{2}, \quad \varepsilon \searrow 0.$$



Related problems : Variational Inequalities (VI)

Variational Inequality Problem(VI)

Given $F : E^n \to E^n, X \subset E^n$

find $x \in X$ such that

$$\langle x' - x, F(x) \rangle \geq 0 \quad \forall x' \in X$$

Example – NLP as VI

Example: Optimization Problem

$$\begin{array}{ll} \text{Min} & f(x) \\ \text{s.t.} & x \in X \subset E^n \\ \text{Where} & X \text{ is a } \textit{convex set} \end{array}$$

We can solve the following (VI) for local minimums: Find $x \in X$ s.t.

$$\langle x' - x, \nabla f(x) \rangle \geq 0, \quad \forall x' \in X$$

Example – Nash equilibrium as VI

Example: Nash Equilibrium

For an N-person non-cooperative game with strategy space $X_j \subset E^{n_j}$ and smooth payoff function $f_j : E^{n_1} \times \cdots \times E^{n_N} \rightarrow R$ for each player j, the necessary condition for $(\bar{x}^1, \cdots, \bar{x}^N)$ being a Nash Equilibrium solution, i.e.,

$$\bar{x}^j \in \arg\min_{x^j \in X_j} f_j(\bar{x}^1, \cdots, \bar{x}^{j-1}, x_j, \bar{x}^{j+1}, \cdots, \bar{x}^N)$$

is that $(\bar{x}^1, \cdots, \bar{x}^N)$ solves (VI) with

$$X = X_1 \times \cdots \times X_N$$

and

$$F(x^1, \cdots, x^N) = (\bigtriangledown_{x^1} f_1, \cdots, \bigtriangledown_{x^N} f_N)$$

Related problems : LCoP

- Linear Conic Programming (LCoP)

  A general conic optimization problem is as follows:

$$
\begin{aligned}
(P) \quad & \text{minimize} && c \circ x \\
& \text{subject to} && A \circ x = b \\
& && x \in \mathrm{K}
\end{aligned}
$$

where K is a closed and convex cone and

"$\circ$" is a linear operator like "inner product."

Related problems : SDP, SOCP, CoP

- Linear Conic Programming (LCoP)
  - When *K* is a cone of symmetric positive semi-definite matrices, LCoP becomes a Semi-Definite Programming (SDP) problem.

  - When *K* is a second order cone, LCoP becomes a Second Order Cone Programming (SCOP) problem.

  - When *K* is a cone of symmetric co-positive matrices, LCoP becomes a Copositive Programming (CoP) problem.

Example

- Any convex optimization problem can be formulated in the conic form.

  -- consider a general convex program with functional constraints:

$$
\begin{aligned}
(CP) \quad & \text{minimize} \quad && c^T x \\
& \text{subject to} \quad && Ax = b \\
& && f_i(x) \le 0,\ i = 1, \ldots, t,
\end{aligned}
$$

where $f_i(x)$'s are convex, $i = 1, \ldots, t$.

Example

For simplicity, consider $t = 1$, A is $m \times n$.

Let

$$\bar{x} := \begin{bmatrix} p \\ q \\ x \end{bmatrix} \in \Re^1 \times \Re^1 \times \Re^n.$$

Define the problem data as

$$\bar{c} := \begin{bmatrix} 0 \\ 0 \\ c \end{bmatrix} \in \Re^1 \times \Re^1 \times \Re^n, \; \bar{b} := \begin{bmatrix} 1 \\ 0 \\ b \end{bmatrix} \in \Re^1 \times \Re^1 \times \Re^m,$$

$$\bar{A} := \begin{bmatrix} 1 & 0 & \mathbf{0}^T \\ 0 & 1 & \mathbf{0}^T \\ \mathbf{0} & \mathbf{0} & A \end{bmatrix} \in \Re^{(m+2)\times(n+2)}.$$

Example

Let

$$\mathcal{K} = \text{cl } \{\bar{x} \mid p > 0,\, q - pf(x/p) \geq 0\} \subseteq \Re^{n+2}.$$

It is a closed convex cone.

Problem (CP) can be written as

$$\begin{array}{lll} (CP) & \text{minimize} & \bar{c}^T \bar{x} \\ & \text{subject to} & \bar{A}\bar{x} = \bar{b} \\ & & \bar{x} \in \mathcal{K}. \end{array}$$

Related problems : NLCP

Nonlinear Complementarity Problem(NLCP)

Given $F : E^n \to E^n$,

find $x \in E^n_+$ (or a *cone* $X \subset E^n$)

s. t. $F(x) \in E^n_+$(*dual cone* of X)

and $\langle x, F(x) \rangle = 0$

Example

Example: For a cone $X \subset E^n$ and

$$dual(X) = \{y \in E^n \mid \langle x, y \rangle \geq 0, \forall x \in X\}$$

In this case, (VI) reduces to (NLCP):

$$x \in X, \quad F(x) \in dual(X)$$
$$\langle x, F(x) \rangle = 0$$

Related problems: MPEC

- Mathematical Programming with Equilibrium Constraints (MPEC)

$$\begin{aligned} \text{Min} \quad & f(x) \\ \text{s. t.} \quad & x \in X \subset E^n \\ & \langle x' - x, F(x) \rangle \geq 0, \ \forall x' \in X \end{aligned}$$

or

$$\begin{aligned} \text{Min} \quad & f(x) \\ \text{s. t.} \quad & x \in X, \ F(x) \in dual(X) \\ & \langle x, F(x) \rangle = 0 \end{aligned}$$

How is NLP related to AI and Machine Learning?

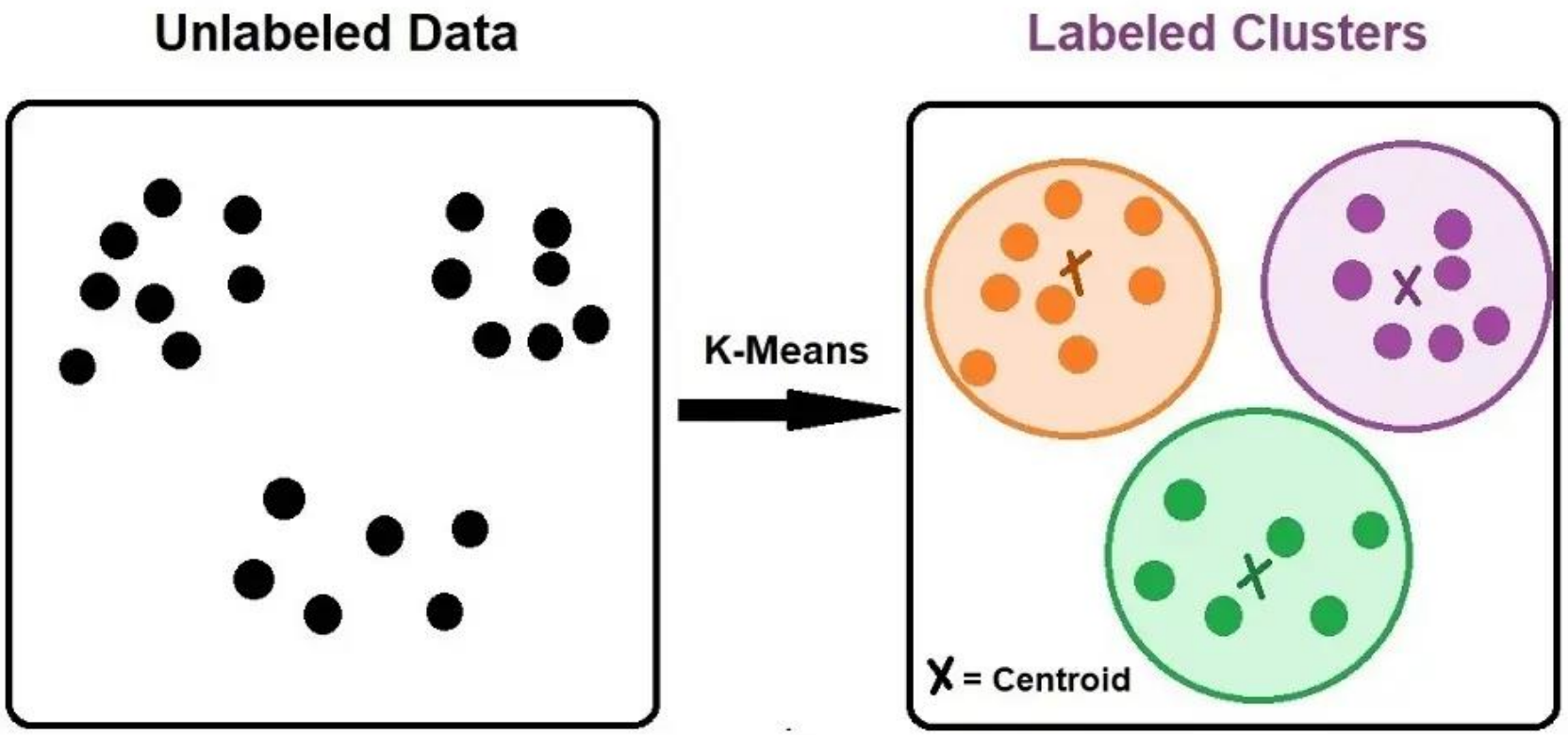
- NLP is the basis of machine learning, especially supervised learning and unsupervised learning,

  such as

  - Data Clustering (C-mean methods)
  - Classification (support vector machines)
  - Prediction (support vector regression)
  - Neural Networks (multi-layer deep leaning)

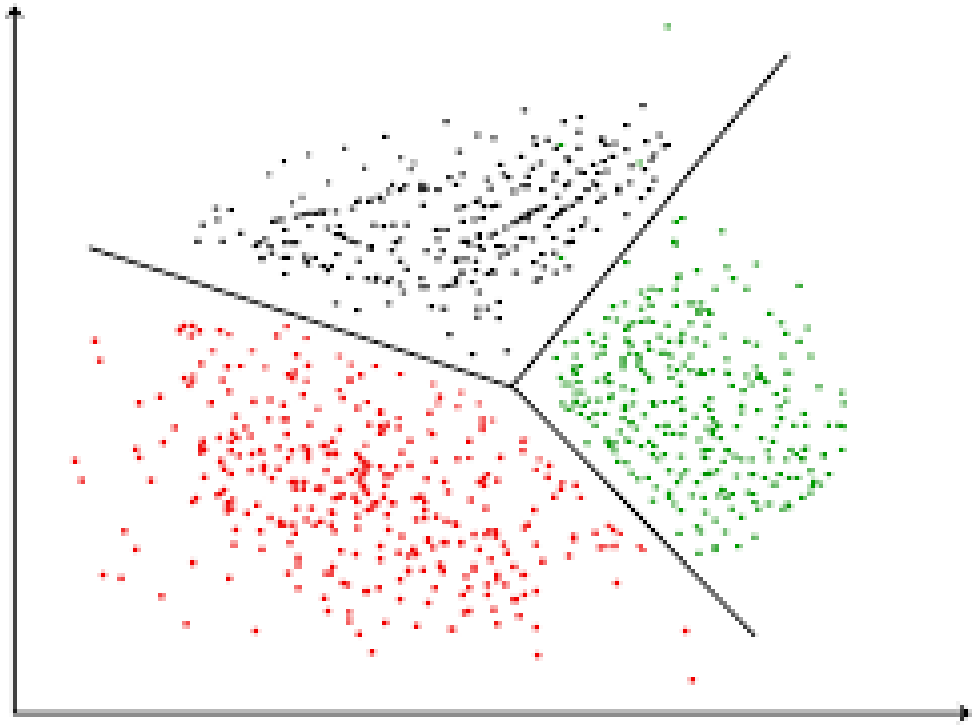
Data clustering

- https://www.ejable.com/tech-corner/ai-machine-learning-and-deep-learning/k-means-clustering/



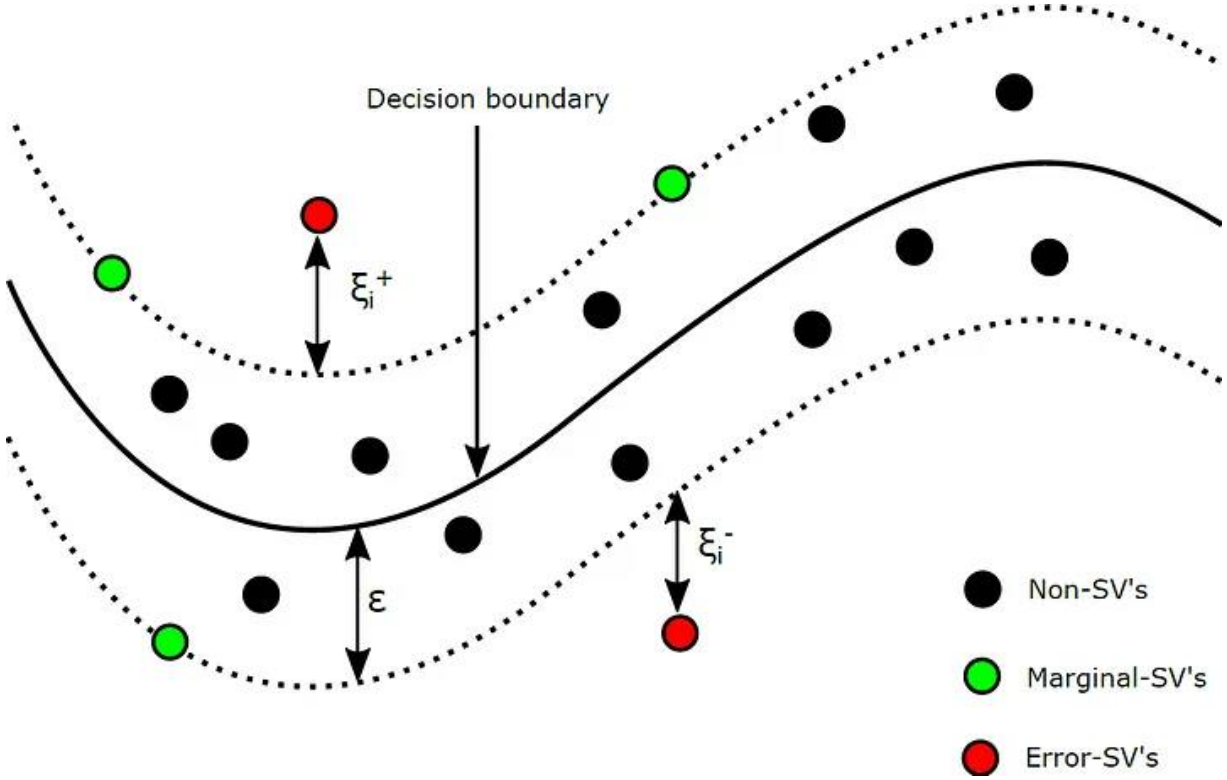
Data classification – Support vector machine

- Labeled data of different classes



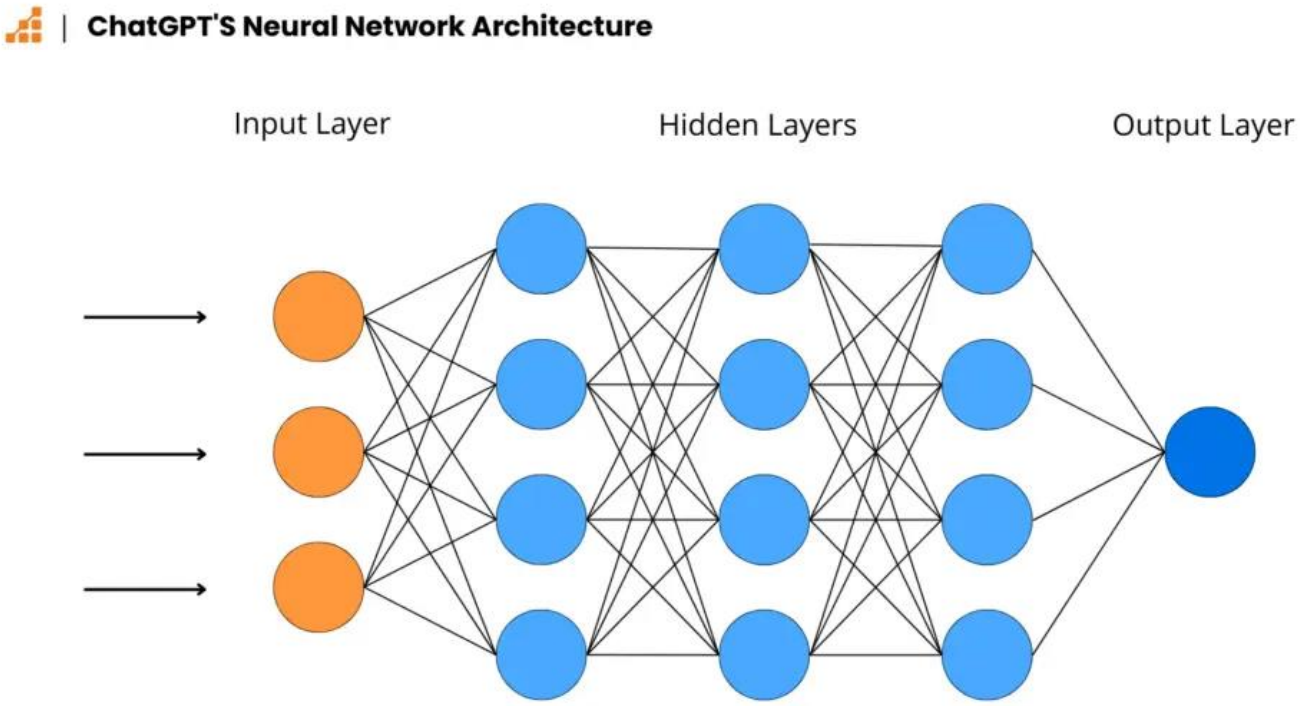
Data regression - Support vector regression

- https://medium.com/analytics-vidhya/support-vector-regression-svr-model-a-regression-based-machine-learning-approach-f4641670c5bb



Artificial neural network

- https://www.scalablepath.com/machine-learning/chatgpt-architecture-explained



Machine learning and optimization

1. Many machine learning problems are formulated as minimization of some loss function that measures discrepancy between the predictions of the model being trained and the actual problem instances, or as maximization of some reward function that affirms an expected decision.

2. One major difference between machine learning and optimization lies in the goal of generalization - optimization intends to minimize the loss/maximize the reward on a set of seen examples while machine learning is concerned with minimizing the loss/maximizing the reward on unseen samples.

Basic approaches of machine learning

I. Supervised learning for classification and prediction

- Support Vector Machines & Regression (SVM & SVR)
- Artificial Neural Networks (ANN)

II. Unsupervised learning for clustering and featuring

- Similarity Learning and Sparse Optimization

III. Reinforcement learning in dynamic environment

- Markov Decision Process and Dynamic Programming

What are essential to know about NLP?

- Problem Formulation
- Solution Analysis
  - Properties of the solutions: e.g., convexity, closeness, compactness.
  - Existence and Uniqueness: sufficient conditions and necessary conditions
  - Sensitivity Analysis
- Duality Theory
- Algorithm/Solution Method
  - Convergence Theorem
  - Efficiency/rate of convergence
  - Complexity

What do I have to know?

- Multivariate/ Advanced Calculus
- Linear Algebra/ Matrix Theory
- Introduction to Operations Research
- Linear Programming
- Reasoning and Mathematical Proving

Learning nonlinear programming

- 706 teaching adopts a process of
  - "motivation",
  - "intuition",
  - "speculation" and
  - "theorization".

The principles (basic approaches) of solving NLP problems are

Approximation, Relaxation and Reformulation