# CS769: Optimization in Machine Learning

## Lecture 1: Introduction & Logistics

Ganesh Ramakrishnan

Department of Computer Science
Dept of CSE, IIT Bombay
https://www.cse.iitb.ac.in/~ganesh

January, 2025

# Outline

- Why take this course?
- Prerequisites
- Week by Week Course plan
- Course Logistics
- Continuous Optimization in Machine Learning
- Discrete Optimization in Machine Learning

# Why take this Course?

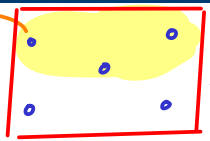
Optimization is everywhere: Big Data and Machine Learning, Scheduling and Planning, Operations Research, control theory, data analysis, simulations, almost all technology we use, search engines, computers/laptops, smart-phones, hardware/software of all kinds, ...

- Mathematical Modeling:
  - defining and modeling the problem

# Why take this Course?

Imagine replacing the yellow occupancy with data

Say: ASR for accented speech
(Eg: Tamil accented English)
Say coverage of diverse Tamil speakers



Class room
Uniform/diverse coverage of classroom
Alignment with sitting density
Minimization of cost
Based on characteristics of bulbs/tube lights

Optimization is everywhere: Big Data and Machine Learning, Scheduling and Planning, Operations Research, control theory, data analysis, simulations, almost all technology we use, search engines, computers/laptops, smart-phones, hardware/software of all kinds, ...

- Mathematical Modeling:
  - defining and modeling the problem } Regression $\min_w \|\phi w - y\|^2 + \lambda \|w\|^2$

# Why take this Course?

Optimization is everywhere: Big Data and Machine Learning, Scheduling and Planning, Operations Research, control theory, data analysis, simulations, almost all technology we use, search engines, computers/laptops, smart-phones, hardware/software of all kinds, ...

- Mathematical Modeling:
  - defining and modeling the problem
- Computational Optimization:
  - Algorithms to solve these optimization problems optimally or near optimally.

# Why take this Course?

Perspectives
1) Regularization to avoid overfitting
2) Optimization: Improve strength of convexity for convergence

Optimization is everywhere: Big Data and Machine Learning, Scheduling and Planning, Operations Research, control theory, data analysis, simulations, almost all technology we use, search engines, computers/laptops, smart-phones, hardware/software of all kinds, ...

- Mathematical Modeling:
  - defining and modeling the problem
- Computational Optimization:
  - Algorithms to solve these optimization problems optimally or near optimally.

Regression $\min_w \|\Phi w - y\|^2 + \lambda \|w\|^2$

quickly enough!!

# Why take this Course?

Machine Learning and AI are embedded in practically every sphere of our life: Gpts, Search & Ads, product search/recommendation, driverless cars, Maps, driver and route matching, Photos, Youtube, Facebook, Twitter, Office Suite, ...

- With democratization of AI, software engineers can write ML applications with a few lines of code!

# Why take this Course?

Machine Learning and AI are embedded in practically every sphere of our life: Gpts, Search & Ads, product search/recommendation, driverless cars, Maps, driver and route matching, Photos, Youtube, Facebook, Twitter, Office Suite, ...

- With democratization of AI, software engineers can write ML applications with a few lines of code!

  1) Yet we need to keep updating existing libraries such as pytorch with newer paradigms for optimization (latest coming from the realm of Generative AI)
  2) In operational/deployment scenarios, we could evolve optimization problems that are more reflective/indicative of end user requirements
  3) Even from Naive consumer perspective we need to be wise about the choice of an appropriate algo/formulation for an application

# Why take this Course?

Machine Learning and AI are embedded in practically every sphere of our life: Gpts, Search & Ads, product search/recommendation, driverless cars, Maps, driver and route matching, Photos, Youtube, Facebook, Twitter, Office Suite, ...

- With democratization of AI, software engineers can write ML applications with a few lines of code!
- Open-source libraries today offer capabilities to build products with practically zero knowledge of ML.

# Why take this Course?

Machine Learning and AI are embedded in practically every sphere of our life: Gpts, Search & Ads, product search/recommendation, driverless cars, Maps, driver and route matching, Photos, Youtube, Facebook, Twitter, Office Suite, ...

- With democratization of AI, software engineers can write ML applications with a few lines of code!
- Open-source libraries today offer capabilities to build products with practically zero knowledge of ML.

# Why take this Course?

Machine Learning and AI are embedded in practically every sphere of our life: Gpts, Search & Ads, product search/recommendation, driverless cars, Maps, driver and route matching, Photos, Youtube, Facebook, Twitter, Office Suite, ...

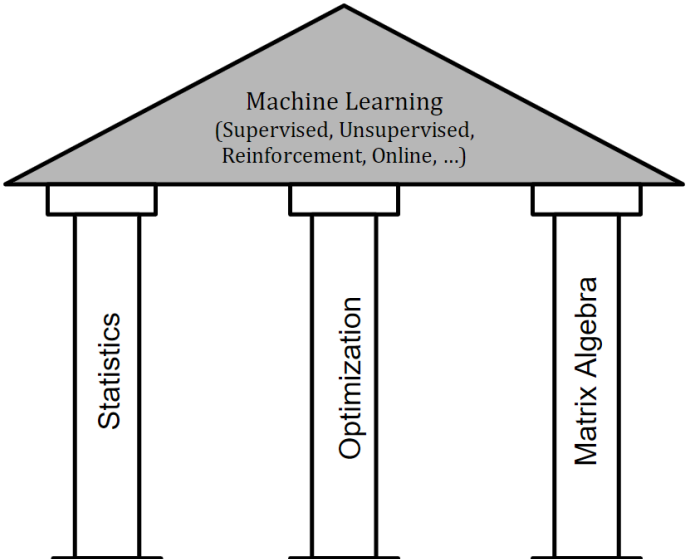
- With democratization of AI, software engineers can write ML applications with a few lines of code!
- Open-source libraries today offer capabilities to build products with practically zero knowledge of ML.
- However to push the boundaries of research and really solve problems, you need to gain hands on experience in ML!

# Why take this Course?

Machine Learning and AI are embedded in practically every sphere of our life: Gpts, Search & Ads, product search/recommendation, driverless cars, Maps, driver and route matching, Photos, Youtube, Facebook, Twitter, Office Suite, ...

- With democratization of AI, software engineers can write ML applications with a few lines of code!
- Open-source libraries today offer capabilities to build products with practically zero knowledge of ML.
- However to push the boundaries of research and really solve problems, you need to gain hands on experience in ML!
- Optimization is one of the important backbones of machine learning.

# Why take this Course?




# Why take this Course?

Optimization is one of the pillars of ML!

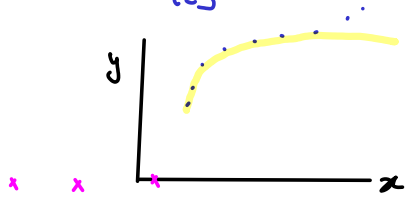
- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.

# Why take this Course?

Optimization is one of the pillars of ML!

- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.

$$\min_{w} \sum_{i \in S} (x_i^T w - y_i)^2 + \sum_{j \in S^c} \left(x_j^T w - y_j^{imp}\right)^2$$



# Why take this Course?

Optimization is one of the pillars of ML!

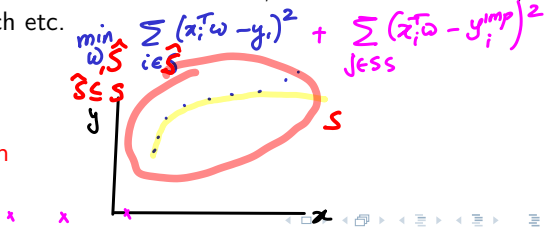
- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.
- Discrete Optimization:
  - Discrete Optimization occurs in Inference problems in structured spaces, certain learning problems and auxilliary problems such as Feature Selection, Data subset selection, Data summarization, Architechture search etc.

# Why take this Course?

Optimization is one of the pillars of ML!

- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.
- Discrete Optimization:
  - Discrete Optimization occurs in Inference problems in structured spaces, certain learning problems and auxilliary problems such as Feature Selection, Data subset selection, Data summarization, Architechture search etc.

Support vector regression
tries to estimate the subset
in continuous optimization formulation



# Why take this Course?

Optimization is one of the pillars of ML!

- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.
- Discrete Optimization:
  - Discrete Optimization occurs in Inference problems in structured spaces, certain learning problems and auxilliary problems such as Feature Selection, Data subset selection, Data summarization, Architechture search etc.
- Mixed Continuous and Discrete Optimization:
  - A growing number of problems including classical problems such as clustering, feature selection, structured sparsity occur as mixed discrete/continuous optimization problems.

# Why take this Course?

Optimization is one of the pillars of ML!

- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.
- Discrete Optimization:
  - Discrete Optimization occurs in Inference problems in structured spaces, certain learning problems and auxilliary problems such as Feature Selection, Data subset selection, Data summarization, Architechture search etc.
- Mixed Continuous and Discrete Optimization:
  - A growing number of problems including classical problems such as clustering, feature selection, structured sparsity occur as mixed discrete/continuous optimization problems.

# Why take this Course?

Optimization is one of the pillars of ML!

- Continuous Optimization:
  - Continuous Optimization often appears as *relaxations* of risk/error minimization problem. The *Learning* problem in many parametrized models (whether supervised, semi-supervised, unsupervised, or reinforcement learning) involves **Continuous Optimization**.
- Discrete Optimization:
  - Discrete Optimization occurs in Inference problems in structured spaces, certain learning problems and auxilliary problems such as Feature Selection, Data subset selection, Data summarization, Architechture search etc.
- Mixed Continuous and Discrete Optimization:
  - A growing number of problems including classical problems such as clustering, feature selection, structured sparsity occur as mixed discrete/continuous optimization problems.

Eg: Determining locations for fitting bulbs in a classroom along with the intensities of those bulbs
Most problems are actually of this nature (in their true sense)

# Why take this Course?

- Countless number of ML libraries available which implement all kinds of optimization algorithms (Tensorflow, pytorch, scipy, sklearn, Vowpal Wabbit, several at `https://huggingface.co/`) and scalability benchmarking platforms (`https://huggingface.co/`, `https://github.com/hpcaitech/ColossalAI`)
- This course will give you the expertise to look inside these algorithms, understand how they work, why they work and how fast they work.
- Invariably in Research, you will come up with new optimization problems which you might need to implement custom algorithms or atleast the loss functions.
- Even if you don't implement new algorithms, you will have a better idea of which algorithm to use in which scneario.

# Philosophy of this Course

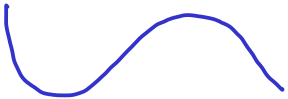
- The spirit of this course is best summarized by the quote of Thomas Cover: *Theory is only the first term of the Taylor's series of Practice*
- This course will focus on mainly the algorithmic aspects of optimization (both continuous and discrete optimization) and not so much on the modeling.
- Give a flavor of the proofs and proof techniques but will try to not make this course heavily theoretical.
- Focus extensively on implementational aspects and as a part of programming assignments, we will implement many ML loss functions and algorithms.
- The reading assignments/seminars will ground you in actual challenges in implementing.

# Why did you enroll for this Course?

Lets hear from a few of you why you took this course...

# Why did you enroll for this Course?

Going into the world beyond gradient descent!
What to do with optimization for non-convex functions



1 Convex region
of a non-convex function

2

Lets hear from a few of you why you took this course...

Diminishing returns

# Prerequisites

- Basic Linear Algebra: Matrices, Vectors
- Basics of Machine Learning (Ideally you should have taken either an undergraduate or graduate ML course)
- Algorithms course (either in undergraduate or graduate version)

# Prerequisites

- Basic Linear Algebra: Matrices, Vectors
- Basics of Machine Learning (Ideally you should have taken either an undergraduate or graduate ML course) Not a hard pre-requisite. Only some parts will be used and that too you can read up
- Algorithms course (either in undergraduate or graduate version)

# Part 1 this Course: Continuous optimization

**Lectures will be fast paced. Students will be referred to previous year lectures as and when required:** `https://bit.ly/cs769-2024` `https://bit.ly/cs769-2023` **and** `https://bit.ly/cs769-2022`

- Basics of Continuous Optimization
- Convexity
- Gradient Descent
- Projected/Proximal GD
- Subgradient Descent
- Accelerated Gradient Descent
- Newton & Quasi Newton
- Duality: Lagrange, Fenchel
- Coordinate Descent
- Frank Wolfe
- Optimization in Practice

# Part 1 this Course: Continuous optimization

**Lectures will be fast paced. Students will be referred to previous year lectures as and when required:** `https://bit.ly/cs769-2024` `https://bit.ly/cs769-2023` **and** `https://bit.ly/cs769-2022`

- Basics of Continuous Optimization
- Convexity
- Gradient Descent
- Projected/Proximal GD
- Subgradient Descent
- Accelerated Gradient Descent
- Newton & Quasi Newton
- Duality: Lagrange, Fenchel
- Coordinate Descent
- Frank Wolfe
- Optimization in Practice

More practicals
Theory in my previous CS709 Convex Opt course

# Part 2 of this Course: Discrete optimization

**Lectures will be fast paced. Students will be referred to previous year lectures as and when required:** `https://bit.ly/cs769-2023` **and** `https://bit.ly/cs769-2022`

- Linear Cost Problems
- Matroids, Spanning Trees
- s-t paths, s-t cuts
- Matchings
- Covers (Set Covers, Vertex Covers, Edge Covers)
- Optimal Transport (if time permits)
- Non-Linear Discrete Optimization
- Submodular Functions
- Submodularity and Convexity
- Submodular Minimization
- Submodular Maximization
- Optimization in Practice

# Part 2 of this Course: Discrete optimization

**Lectures will be fast paced. Students will be referred to previous year lectures as and when required:** `https://bit.ly/cs769-2023` **and** `https://bit.ly/cs769-2022`

- Linear Cost Problems
- Matroids, Spanning Trees
- s-t paths, s-t cuts
- Matchings
- Covers (Set Covers, Vertex Covers, Edge Covers)
- Optimal Transport (if time permits)

- Non-Linear Discrete Optimization
- Submodular Functions
- Submodularity and Convexity
- Submodular Minimization
- Submodular Maximization
- Optimization in Practice

# Parts 3 and 4 of this course: Seminar Papers and Presentations

- CS 769 Course Seminar and Project Paper List 2025: https://bit.ly/cs769-course-paper-2025
- **Description:** Team of (2 to max 3) assigned one topic at the start (They can choose from a form that we submit).
  **Expectations:**
  1. Reading assignments: Scrutinise the paper and how it connects to the broader scheme of the topic they chose.
  2. **Presentation (Teamwise):** 1 team presentation per week post-midsem, during lecture hours. [Graded]
  3. **Class Discussions:** The remaining teams MUST participate in class discussions for other paper presentations. [Graded]

# Parts 3 and 4 of this course: Seminar Papers and Presentations

- **Written Responses as follow up:** Every other team (including team who presents that week) will be expected to attend and interact and turn in a response summarising their understanding and related observations. Individual responses will be shared with the class, and will guide the class discussion. Any novelty in proposal or implementation/tried out experiments will be given bonus marks. [Graded]
- **Organization:** Paper presentations in both slots [Lectures on Topics as required to supplement]
- **Projects:** Each Team (same as seminar) will pick a group project based on the topic they chose at the start. Both theoretical and empirical investigations may be undertaken. Students may work alone or in teams (of size up to three). Each team will be guided individually through the phases of the research project, but will share its progress with the class at designated intervals. [Graded]

# Relevant Books for this Course: Most on Moodle

**Notes:**

- My notes on Convex Optimization, *etc.* (on Moodle only)
- Course notes on Optimization for Machine Learning[1]
- Convex Optimization: Algorithms and Complexity, by Sébastien Bubeck
- Convex Optimization, Stephen Boyd and Lieven Vandenberghe
- Introductory Lectures on Convex Optimization, Yurii Nesterov
- Optimization for Machine Learning, Edited by Suvrit Sra, Sebastian Nowozin and Stephen J. Wright
- A Course in Combinatorial Optimization, Alexander Schrijver
- Learning with Submodular Functions: A Convex Optimization Perspective, Francis Bach
- Zhang, Lipton, Li and Smola, Dive into Deep Learning (`http://d2l.ai/`)
- Schrijver, Alexander, Combinatorial optimization: polyhedra and efficiency, Vol. 24. Springer Science & Business Media, 2003.
- Fujishige, Satoru. Submodular functions and optimization. Vol. 58. Elsevier, 2005.

[1] `https://mathematical-tours.github.io/book-sources/optim-ml/OptimML.pdf`

# Relevant Books for this Course: Most on Moodle

**Notes:**

- My notes on Convex Optimization, *etc.* (on Moodle only)
- Course notes on Optimization for Machine Learning[1]
- Convex Optimization: Algorithms and Complexity, by Sébastien Bubeck
- Convex Optimization, Stephen Boyd and Lieven Vandenberghe
- Introductory Lectures on Convex Optimization, Yurii Nesterov
- Optimization for Machine Learning, Edited by Suvrit Sra, Sebastian Nowozin and Stephen J. Wright
- A Course in Combinatorial Optimization, Alexander Schrijver
- Learning with Submodular Functions: A Convex Optimization Perspective, Francis Bach
- Zhang, Lipton, Li and Smola, Dive into Deep Learning (`http://d2l.ai/`)
- Schrijver, Alexander, Combinatorial optimization: polyhedra and efficiency, Vol. 24. Springer Science & Business Media, 2003.
- Fujishige, Satoru. Submodular functions and optimization. Vol. 58. Elsevier, 2005.

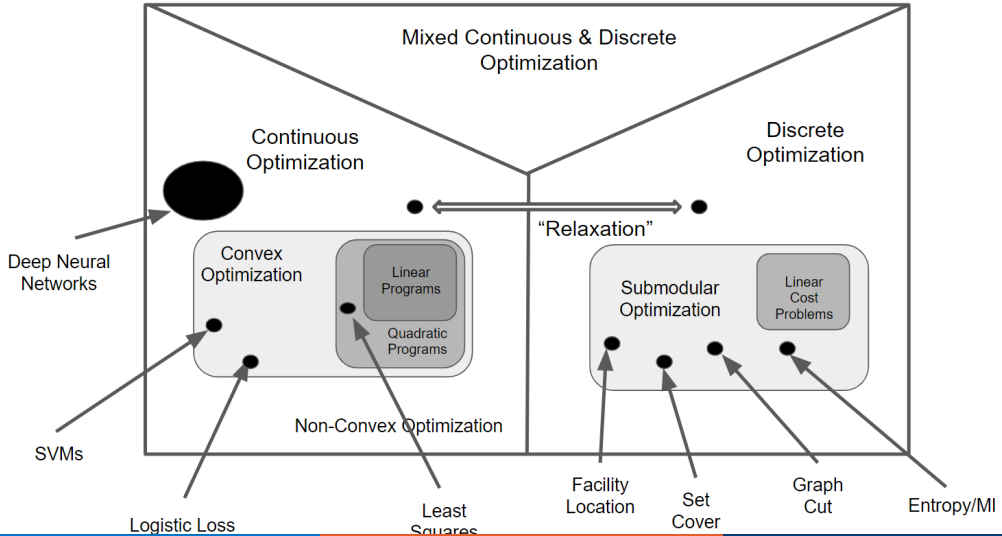
[1] `https://mathematical-tours.github.io/book-sources/optim-ml/OptimML.pdf`

# Continuous Optimization in Machine Learning

Continuous Optimization often appears as *relaxations* of empirical risk minimization problems.

- Supervised Learning: Fine tuning of Deep Models, Fine tuning of LLMs
  - Canonical examples of Logistic Regression, Least Squares, Support Vector Machines,
- Unsupervised Learning: Pre-training of Deep Models, Pre-training of LLMs,
  - k-Means Clustering, Principal Component Analysis
- Reinforcement Leaning from Human Feedback (RLHF), Agentic Systems
  - Canonical examples of Contextual Bandits and Reinforcement Learning: Soft-Max Estimators, Policy Exponential Models
  - Canonical examples of Recommender Systems: Matrix Completion, Non-Negative Matrix Factorization, Collaborative Filtering

# Big Picture: Types of Optimization Problems



# Logistics, Grading etc: Mostly moodle, some at https://www.cse.iitb.ac.in/~ganesh/cs769/

- Credit/Audit Requirements Anyone who does an exceptional course project that has the potential to be a publishable paper is eligible for a straight AA grade. Otherwise the grading breakup would be:
  - 20% Mid-semester exam
  - 25% End semester exam
  - 35% Project: A basic project will take any of the algorithms we study or any related papers, implement the algorithms in the paper, do a basic performance study and diagnose the performance. However, I would expect most projects to suggest ideas for improvement (atleast in specific settings such as multi core or multiple nodes or reasonable assumptions on matrices etc in the problem for which greater speedup is possible). A more advanced project would take a problem specification for which no solution is publicly available, figure out how to solve it, and implement the solution.
  - 20% Reading and paper presentation.
- Lectures: In Slot 9, Lectures on MS Teams (Code: **cnsjv56**)
- All lecture recordings and slides will be organized on moodle (CS 769-2024-2— Course name Optimization in Machine Learning)
- TA(s): Prateek Chand, Priya Mishra, Suraj Racha, Vedant Goswami (and possibly more TAs might get added)

# Course Project Ideas

- Let's spend a few minutes discussing some ideas for course project(s)
- Recap: CS 769 Course Seminar and Project Paper List 2025: https://bit.ly/cs769-course-paper-2025
- Encourage to jointly contribute to BharatGen (https://bharatgen.tech) and **DECILE** `https://decile.org/` python toolkit and add components to it **especially through components in GenAI** aligned with CoreSets, Curriculum, JEPA/Large Concept Models, Knowledge Distillation, HPO, NAS (computational heavy),
- Overview presentation on BharatGen and associated research landscape: `https://bit.ly/bharatgen-landscape`. Video talk followed by panel discussion: `https://youtu.be/nAEhVIT_ycA?t=4289`.
- Overview Presentation on DECILE and underlying research, as part of the Faculty Unplugged Seminar Series (FUSS): https://youtu.be/e2e3PY352BI. Impacts through Data Efficient Learning (snapshots from recent talks at MIT, CMU, Harvard, UIUC, *etc*).
- We can discuss ideas on this as the class progresses.

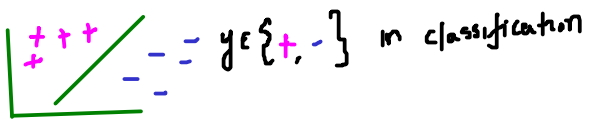
# Outline

- Why take this course? [Done]
- Prerequisites [Done]
- Course plan [Done]
- Course Logistics [Done]
- Continuous Optimization in Machine Learning: Applications include
  1. Supervised Learning
  2. Clustering
  3. Principal Component Analysis
  4. Low Rank and Non Negative Matrix Factorization
  5. Contextual Bandits and Learning from Logged Data
- Discrete Optimization in Machine Learning

# Application 1: Supervised Learning

- Data: Given training examples $\{(x_1, y_1), \cdots, x_n, y_n\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.
- Applications: Several different models depending on the applications:
  - Email Spam Filtering: Features are words, phrases, regexps in the email, Label is "+1" for Spam, "0" for Not Spam.
  - Handwritten Digit Recognition: Features are Images, Label is the Digit (say between "0" to "9").
  - Housing price Prediction: Features are House properties (square footage, # Bedrooms/Bathrooms, Location, ...) and Label is the Cost (continuous variable).

# Application 1: Supervised Learning

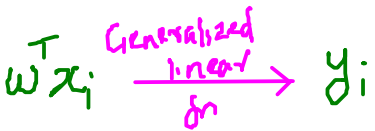



- Data: Given training examples $\{(x_1, y_1), \cdots, x_n, y_n\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.
- Applications: Several different models depending on the applications:
  - Email Spam Filtering: Features are words, phrases, regexps in the email, Label is "+1" for Spam, "0" for Not Spam.
  - Handwritten Digit Recognition: Features are Images, Label is the Digit (say between "0" to "9").
  - Housing price Prediction: Features are House properties (square footage, # Bedrooms/Bathrooms, Location, ...) and Label is the Cost (continuous variable).

# Supervised Learning: Modeling

- Data: Given training examples $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.

# Supervised Learning: Modeling




- Data: Given training examples $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.

# Supervised Learning: Modeling

- Data: Given training examples $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.
- Model: Denote the Model by $F_\theta(x)$ with $\theta$ being the parameters of the model. Model examples: $F_\theta(x) = \theta^T x$ as a simple linear model. Deep Models are recursive functions:

$$F_{\theta_1, \theta_2, \cdots, \theta_l}(x) = f_1(\theta_1^T f_2(\cdots \theta_{l-1}^T f_l(\theta_l^T x)))$$

# Supervised Learning: Modeling

$\theta = w$

- Data: Given training examples $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.
- Model: Denote the Model by $F_\theta(x)$ with $\theta$ being the parameters of the model. Model examples: $F_\theta(x) = \theta^T x$ as a simple linear model. Deep Models are recursive functions:

$$F_{\theta_1,\theta_2,\cdots,\theta_l}(x) = f_1(\theta_1^T f_2(\cdots \theta_{l-1}^T f_l(\theta_l^T x)))$$

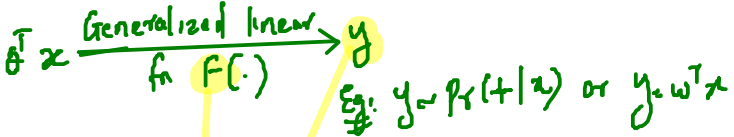
# Supervised Learning: Modeling

- Data: Given training examples $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.
- Model: Denote the Model by $F_\theta(x)$ with $\theta$ being the parameters of the model. Model examples: $F_\theta(x) = \theta^T x$ as a simple linear model. Deep Models are recursive functions:

$$F_{\theta_1, \theta_2, \cdots, \theta_l}(x) = f_1(\theta_1^T f_2(\cdots \theta_{l-1}^T f_l(\theta_l^T x)))$$

- Loss Functions: The Loss Function $L$ tries to measure the *distance* between $F_\theta(x_i)$ and $y_i$.

# Supervised Learning: Modeling




- Data: Given training examples $\{(x_1, y_1), \cdots, (x_n, y_n)\}$ where $x_i \in \mathbf{R}^m$ is the feature vectors and $y_i$ is the label.
- Model: Denote the Model by $F_\theta(x)$ with $\theta$ being the parameters of the model. Model examples: $F_\theta(x) = \theta^T x$ as a simple linear model. Deep Models are recursive functions:

$$F_{\theta_1,\theta_2,\cdots,\theta_l}(x) = f_1(\theta_1^T f_2(\cdots \theta_{l-1}^T f_l(\theta_l^T x)))$$

- Loss Functions: The Loss Function $L$ tries to measure the *distance* between $F_\theta(x_i)$ and $y_i$.

Distances could be probabilitistic

ML1-LinearRegression.ipynb :
https://colab.research.google.com/drive/13ILGLUU_k4VS_tCpsMivLc9nnvOIuxAN#scrollTo=PyeQbbkjdE6v

# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda \Omega(\theta)$$

# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda\Omega(\theta)$$

Purposes of regularizer:
1) Increasing the bias of the model to yield $\theta$ that are simpler (say smaller norms)
<==> Adding a prior to the model (see extra optional slides at the end)
2) In this course: We see how the regularizer/prior makes the optimization problem well behaved (through properties such as strong local/global convexity) resulting in faster convergence rates of optimization algorithms

# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda \Omega(\theta)$$

- $L$: *Loss function*, $\Omega$: *Regularizer*. Example $F_\theta(x) = \theta^T x$

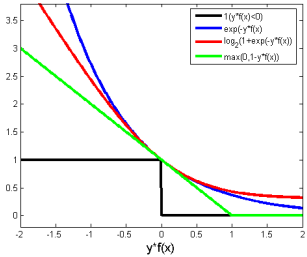
# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda\Omega(\theta)$$

- $L$: *Loss function*, $\Omega$: *Regularizer*. Example $F_\theta(x) = \theta^T x$

- Examples of $L$:
  - Logistic Loss: $\log(1 + \exp(-y_i F_\theta(x_i)))$
  - Hinge Loss: $\max\{0, 1 - y_i F_\theta(x_i)\}$
  - Softmax Loss: $-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^{k} \exp(F_{\theta_c}(x_i)))$
  - Absolute Error: $|F_\theta(x_i) - y_i|$
  - Least Squares: $(F_\theta(x_i) - y_i)^2$



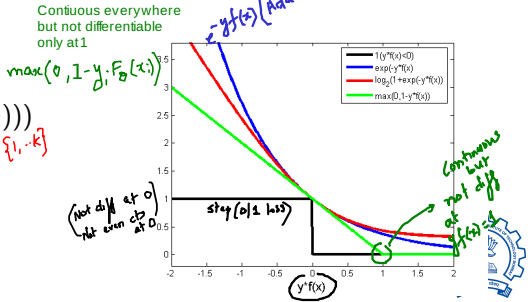
# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda\Omega(\theta)$$

- $L$: *Loss function*, $\Omega$: *Regularizer*. Example $F_\theta(x) = \theta^T x$
- Examples of $L$:
  - Logistic Loss: $\log(1 + \exp(-y_i F_\theta(x_i)))$
  - Hinge Loss: $\max\{0, 1 - y_i F_\theta(x_i)\}$
  - Softmax Loss: $-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^{k} \exp(F_{\theta_c}(x_i)))$
  - Absolute Error: $|F_\theta(x_i) - y_i|$
  - Least Squares: $(F_\theta(x_i) - y_i)^2$

Purer regression losses



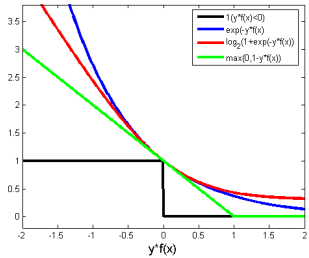
# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda\Omega(\theta)$$

- $L$: *Loss function*, $\Omega$: *Regularizer*. Example $F_\theta(x) = \theta^T x$

- Examples of $L$:
  - Logistic Loss: $\log(1 + \exp(-y_i F_\theta(x_i)))$
  - Hinge Loss: $\max\{0, 1 - y_i F_\theta(x_i)\}$
  - Softmax Loss: $-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^{k} \exp(F_{\theta_c}(x_i)))$
  - Absolute Error: $|F_\theta(x_i) - y_i|$
  - Least Squares: $(F_\theta(x_i) - y_i)^2$
- Examples of $\Omega$:
  - L1 Regularizer: $\sum_{i=1}^{m} |\theta[i]|$
  - L2 Regularizer: $\sum_{i=1}^{m} \theta[i]^2$



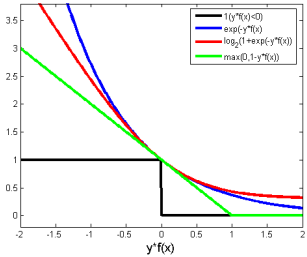
# Supervised Learning: Optimization Problem

- "Loss plus Regularizer" Framework:

$$\min_{\theta} G(\theta) = \sum_{i=1}^{n} L(F_\theta(x_i), y_i) + \lambda\Omega(\theta)$$

$\min_\theta L(F, y)$ s.t. $\Omega(\theta) \le r$

- $L$: *Loss function*, $\Omega$: *Regularizer*. Example $F_\theta(x) = \theta^T x$
- Examples of $L$:
  - Logistic Loss: $\log(1 + \exp(-y_i F_\theta(x_i)))$
  - Hinge Loss: $\max\{0, 1 - y_i F_\theta(x_i)\}$
  - Softmax Loss: $-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^{k} \exp(F_{\theta_c}(x_i)))$
  - Absolute Error: $|F_\theta(x_i) - y_i|$
  - Least Squares: $(F_\theta(x_i) - y_i)^2$
- Examples of $\Omega$:
  - L1 Regularizer: $\sum_{i=1}^{m} |\theta[i]|$ → sparsity (corners)
  - L2 Regularizer: $\sum_{i=1}^{m} \theta[i]^2$

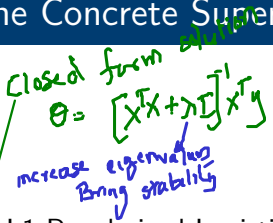


# Some Concrete Supervised Learning Instances

Q: Which of these has/have close form solution(s)?

- L1 Regularized Logistic Regression: $\min_\theta \sum_{i=1}^n \log(1 + \exp(-y_i F_\theta(x_i))) + \lambda \sum_{i=1}^m |\theta[i]|$
- L2 Regularized Logistic Regression: $\min_\theta \sum_{i=1}^n \log(1 + \exp(-y_i F_\theta(x_i))) + \lambda \sum_{i=1}^m \theta[i]^2$
- L2 Regularized SVMs: $\min_\theta \sum_{i=1}^n \max\{0, 1 - y_i F_\theta(x_i)\} + \lambda \sum_{i=1}^m \theta[i]^2$
- L2 Regularized Multi-class Logistic Regression: $\min_{\theta_1,\cdots,\theta_k} \sum_{i=1}^n \{-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^k \exp(F_{\theta_c}(x_i)))\} + \sum_{i=1}^c \lambda \sum_{j=1}^m \theta_i[j]^2$
- L1 Regularized Least Squares (Lasso): $\min_\theta \sum_{i=1}^n (F_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^m |\theta[i]|$
- L2 Regularized Least Squares (Ridge): $\min_\theta \sum_{i=1}^n (F_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^m \theta[i]^2$

# Some Concrete Supervised Learning Instances




- L1 Regularized Logistic Regression: $\min_\theta \sum_{i=1}^n \log(1 + \exp(-y_i F_\theta(x_i))) + \lambda \sum_{i=1}^m |\theta[i]|$
- L2 Regularized Logistic Regression: $\min_\theta \sum_{i=1}^n \log(1 + \exp(-y_i F_\theta(x_i))) + \lambda \sum_{i=1}^m \theta[i]^2$
- L2 Regularized SVMs: $\min_\theta \sum_{i=1}^n \max\{0, 1 - y_i F_\theta(x_i)\} + \lambda \sum_{i=1}^m \theta[i]^2$
- L2 Regularized Multi-class Logistic Regression: $\min_{\theta_1, \cdots, \theta_k} \sum_{i=1}^n \{-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^k \exp(F_{\theta_c}(x_i)))\} + \sum_{i=1}^c \lambda \sum_{j=1}^m \theta_i[j]^2$
- L1 Regularized Least Squares (Lasso): $\min_\theta \sum_{i=1}^n (F_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^m |\theta[i]|$
- L2 Regularized Least Squares (Ridge): $\min_\theta \sum_{i=1}^n (F_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^m \theta[i]^2$

# Some Concrete Supervised Learning Instances

These are only some combinations. You can also have L1 variants for SVM and Logistic Regression

The rest are solved using iterative algorithms (discussed in this course)

- L1 Regularized Logistic Regression: $\min_\theta \sum_{i=1}^n \log(1+\exp(-y_i F_\theta(x_i))) + \lambda \sum_{i=1}^m |\theta[i]|$
- L2 Regularized Logistic Regression: $\min_\theta \sum_{i=1}^n \log(1+\exp(-y_i F_\theta(x_i))) + \lambda \sum_{i=1}^m \theta[i]^2$
- L2 Regularized SVMs: $\min_\theta \sum_{i=1}^n \max\{0, 1-y_i F_\theta(x_i)\} + \lambda \sum_{i=1}^m \theta[i]^2$
- L2 Regularized Multi-class Logistic Regression: $\min_{\theta_1,\cdots,\theta_k} \sum_{i=1}^n \{-F_{\theta_{y_i}}(x_i) + \log(\sum_{c=1}^k \exp(F_{\theta_c}(x_i)))\} + \sum_{i=1}^c \lambda \sum_{j=1}^m \theta_i[j]^2$
- L1 Regularized Least Squares (Lasso): $\min_\theta \sum_{i=1}^n (F_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^m |\theta[i]|$
- L2 Regularized Least Squares (Ridge): $\min_\theta \sum_{i=1}^n (F_\theta(x_i) - y_i)^2 + \lambda \sum_{i=1}^m \theta[i]^2$

The last one is the only one to have a closed form solution.