

ECE421

Intro. to Machine Learning

Winter 2022

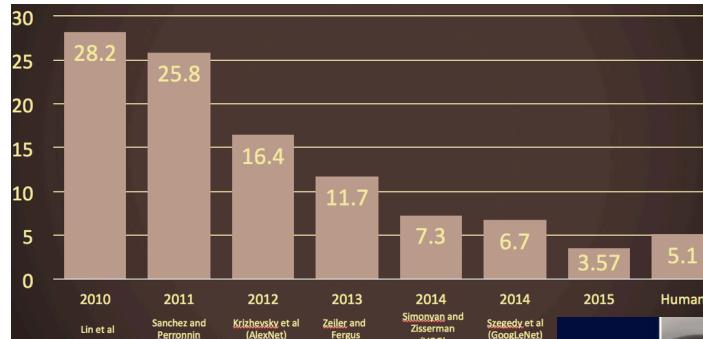
Ashish Khisti (Sec 1)

Christopher Lucasius (Sec 2)

ImageNet Challenge

IMAGENET

- 1,000 object classes (categories).
- Images:
 - 1.2 M train
 - 100k test.



DIAGNOSTICS

• Natural language processing

English Spanish French Spanish - detected -

English Spanish Arabic -

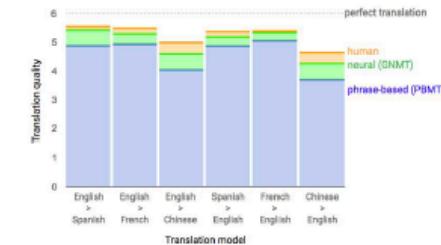
Translate

Capítulo primero. Que trata de la condición y ejercicio del famoso hidalgó don Quijote de la Mancha

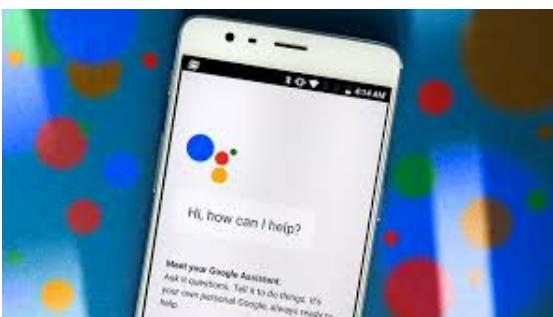
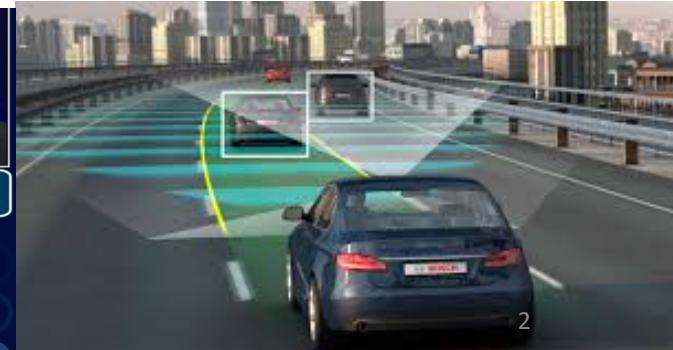
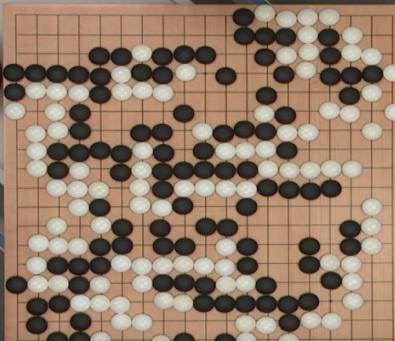
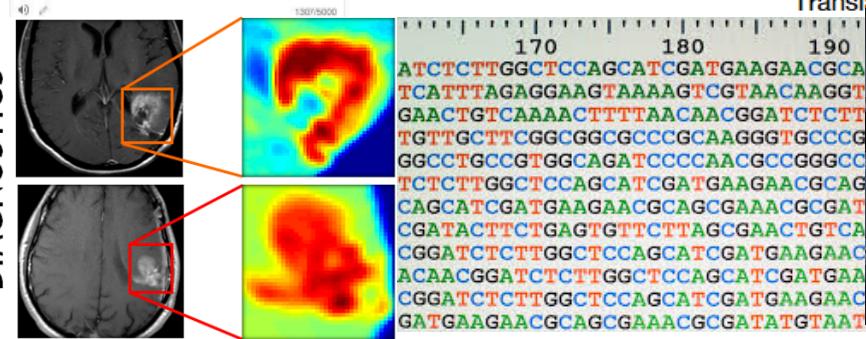
First chapter. Which deals with the condition and exercise of the famous nobleman Don Quixote de la Mancha

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgó de los de lanza en astillero, adarga antigua, rocin flaco y galgo comedor. Una olla de algo más vaca que camero, salpicón las más noches, duelos y quebrantos los sábados, lantanas los domingos, consumían las tres partes de su hacienda. El resto de la conculacian sayo de velante, calzas de velludo para las fiestas, con sus puntifllos de lo mismo, y los días de entremesadas se honraba con su vellor de lo más fino. Tenía en su casa una ama que pasaba de los cuarenta, y una sobrina que no llegaba a los veinte, y un mozo de campo y plaza, que así ensillaba el rocin como tomaba la podadera. Frisaba la edad de nuestro hidalgó con los cincuenta años; era de complección rechoncha, seco de carnes, enjuto de rostro, gran madrugador y amigo de la caza. Quieren decir que tenía el sobrenombre de Quijada, o Quesada, que en esto hay alguna diferencia en los autores que deste caso escriben; aunque, por conjecturas verosímiles, se dejá entender que se llamaba Quejana. Pero esto importa poco a nuestro cuento; basta que en la narración déjese no se salga un punto de la verdad.

Suggest an edit

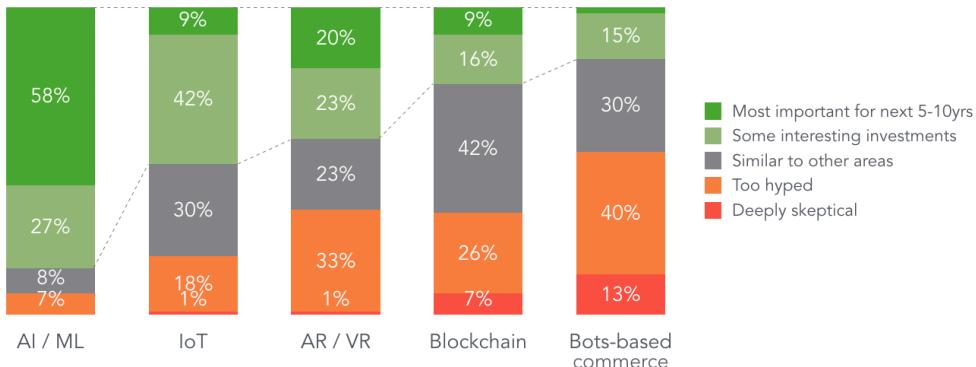


Google's Neural Machine Translation (Wu et al. 2016)



Most VCs are most excited about AI & Machine Learning as their most important investment theme for the coming 5-10 years.

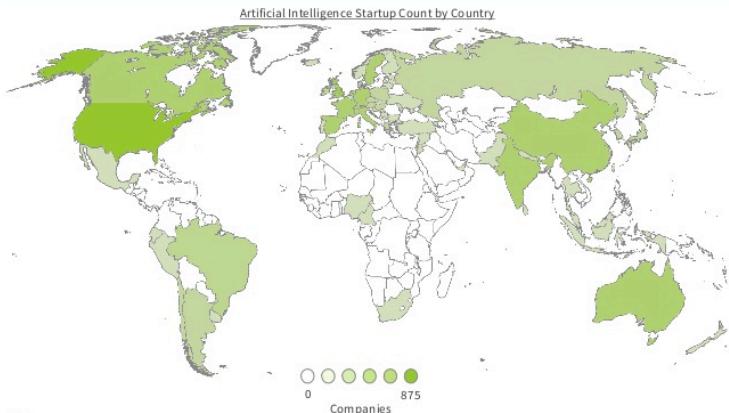
Q. How do you feel about the following investment areas?



17 Source: Upfront Ventures survey of VCs (N=115), Jan 2017

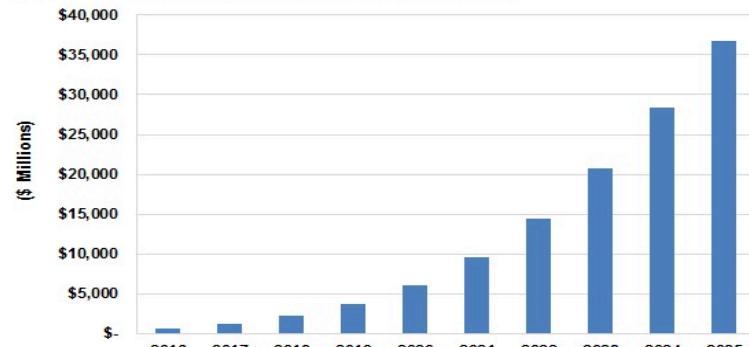
upfront
VENTURES

Artificial intelligence startups are a global phenomenon

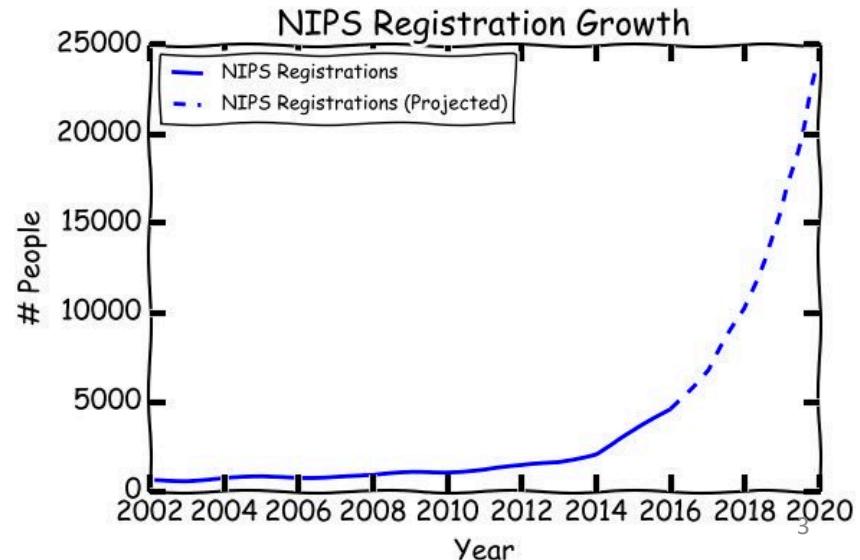


Tractica

Artificial Intelligence Revenue, World Markets: 2016-2025



Source: Tractica



Machine Learning Jargon

semi-supervised learning	overfitting	stochastic gradient descent	SVM	<i>Q</i> learning
Gaussian processes	deterministic noise			
distribution-free	<i>linear regression</i>	VC dimension	data snooping	learning curves
collaborative filtering	nonlinear transformation			mixture of experts
decision trees	RBF	<i>training versus testing</i>	neural networks	<i>no free</i>
active learning	linear models	bias-variance tradeoff	noisy targets	<i>Bayesian prior</i>
ordinal regression	cross validation	logistic regression	weak learners	
ensemble learning	error measures	types of learning	data contamination	hidden Markov models
exploration versus exploitation		kernel methods	perceptrons	graphical models
clustering	is learning feasible?		soft-order constraint	
	regularization	weight decay	Occam's razor	Boltzmann machine

Machine Learning

Develop computational systems to adaptively improve their performance with experience accumulated from the observed data.

Data Driven Approach

Overview

Learning Methods

Supervised Learning

- Linear Models
- Neural Networks
- Support Vector Machines
(Prediction)

Unsupervised Learning

- Clustering
- Density Estimation
- EM algorithm
(No Prediction)

Learning Theory

- PAC Learning
- VC Dimension
- Bias Variance Tradeoff

Learning Principles

- Regularization
- Validation

Linear Classification

- Given Training Samples: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$

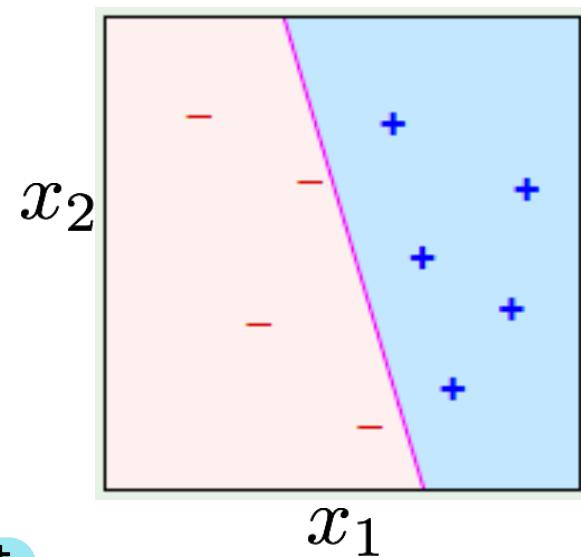
$$\mathbf{x}_i \in \mathbb{R}^d, \quad y_i \in \{\pm 1\}$$

Observation Vector Binary Label

- Determine a *classification rule*

$$y = \text{sign} \left(\sum_{j=0}^d w_j x_j \right)$$

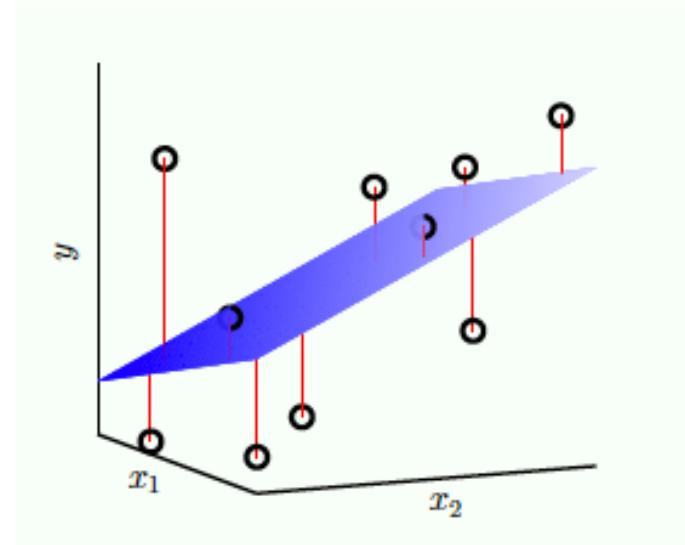
to minimize classification error



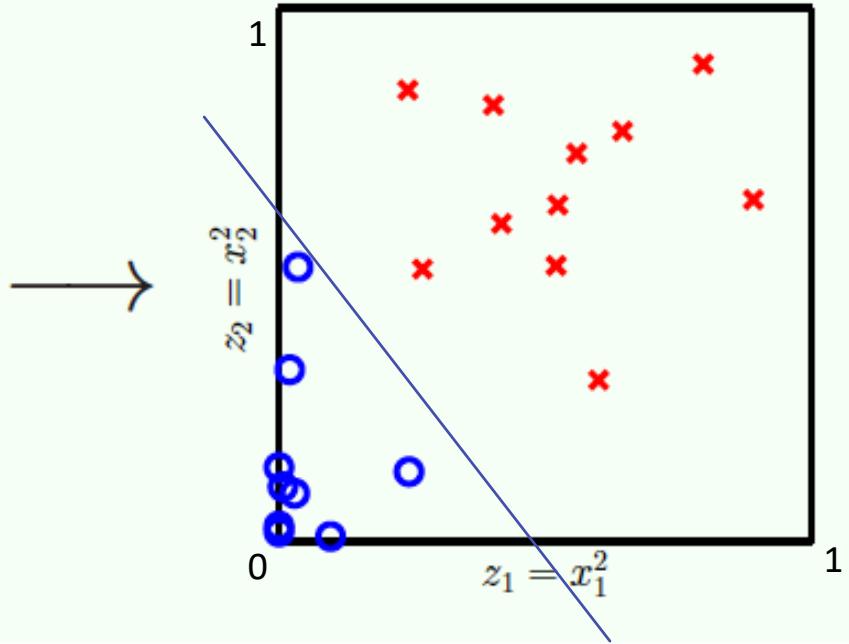
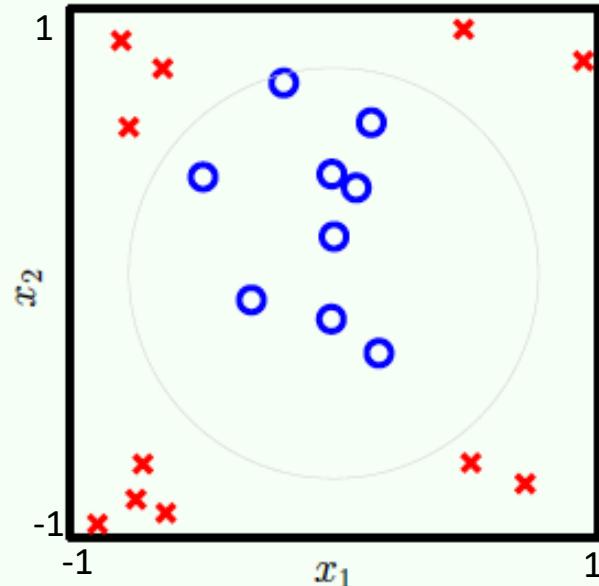
- Perceptron Learning Algorithm
- Logistic Regression and Gradient Descent

Linear Regression

- Given Training Samples: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$
 $\mathbf{x}_i \in \mathbb{R}^d, \quad y_i \in \mathbb{R}$ (Real Number)
w_i: Learning Weight
- Determine a *regression rule*: $\hat{y} = \sum_{i=0}^d w_i x_i$
 $\mathbf{x} = (x_1, \dots, x_d)$
- Minimize the prediction error:
$$\sum_{i=1}^N (y_i - \hat{y}_i)^2$$
 Least Square Error
- Least Squares and its variations



Non-Linear Transformations of Data

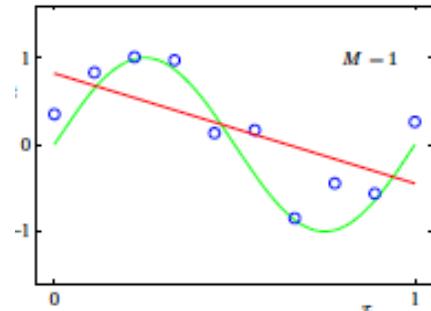


$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \implies \mathbf{z} = \begin{bmatrix} 1 \\ x_1^2 \\ x_2^2 \end{bmatrix}$$

$$\hat{y} = \text{sign}(\mathbf{w}^T \mathbf{z})$$

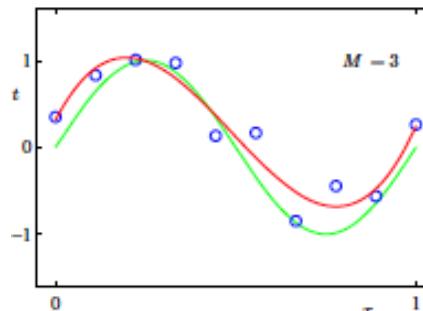
Nonlinear Transforms, Feature Vectors, Kernel Methods

Non-Linear Transformations of Data



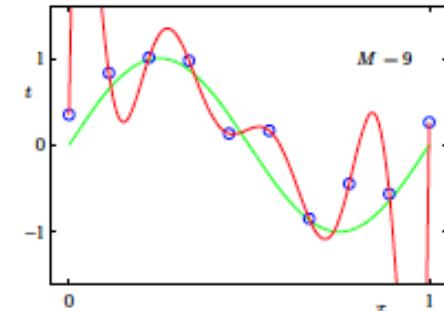
$$\mathbf{z} = [1 \quad x]^T$$

First Order Polynomial



$$\mathbf{z} = [1 \quad x \quad x^2 \quad x^3]^T$$

Third Order Polynomial



$$\mathbf{z} = [1 \quad x \quad x^2 \quad \dots \quad x^9]^T$$

9th Order Polynomial

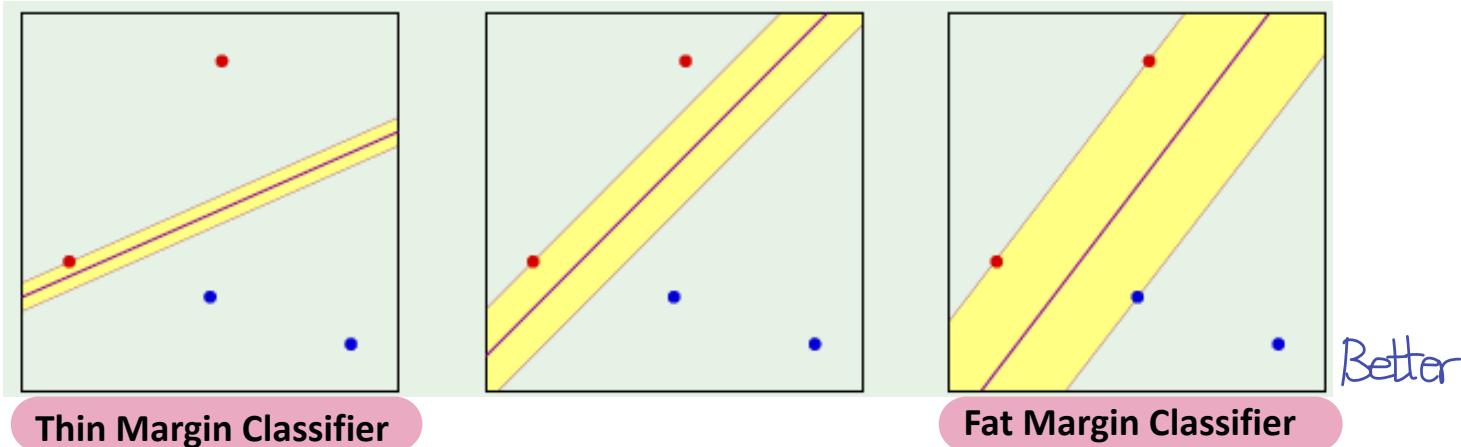
Under-fitting

Overfitting

$$\hat{y} = \mathbf{w}^T \mathbf{z}$$

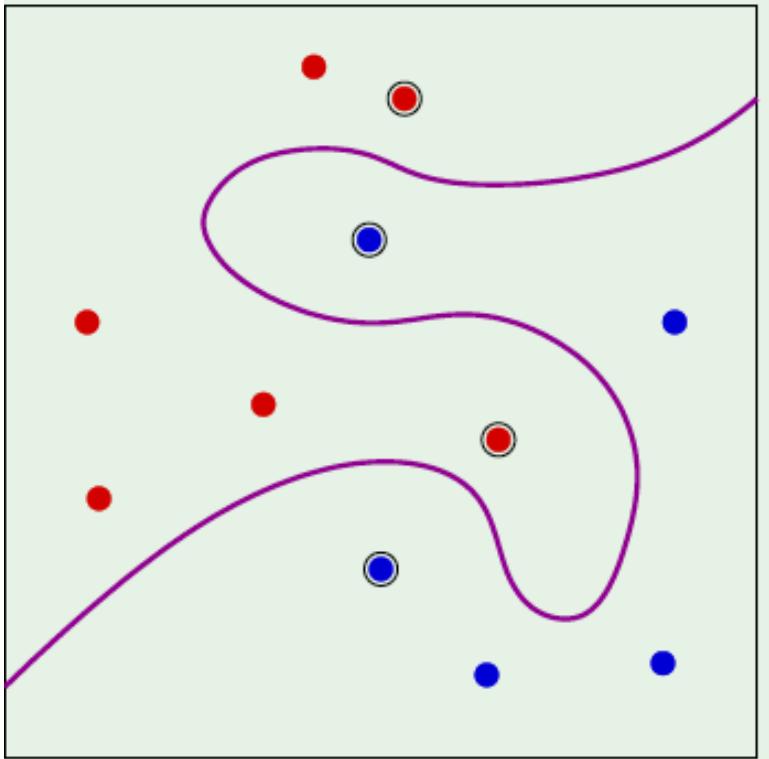
- Higher Order Non-Linearity:
 - Better Fit to Training Data
 - Less Robust to Noise (Overfitting)

Support Vector Machines (Find Maximum Margin)



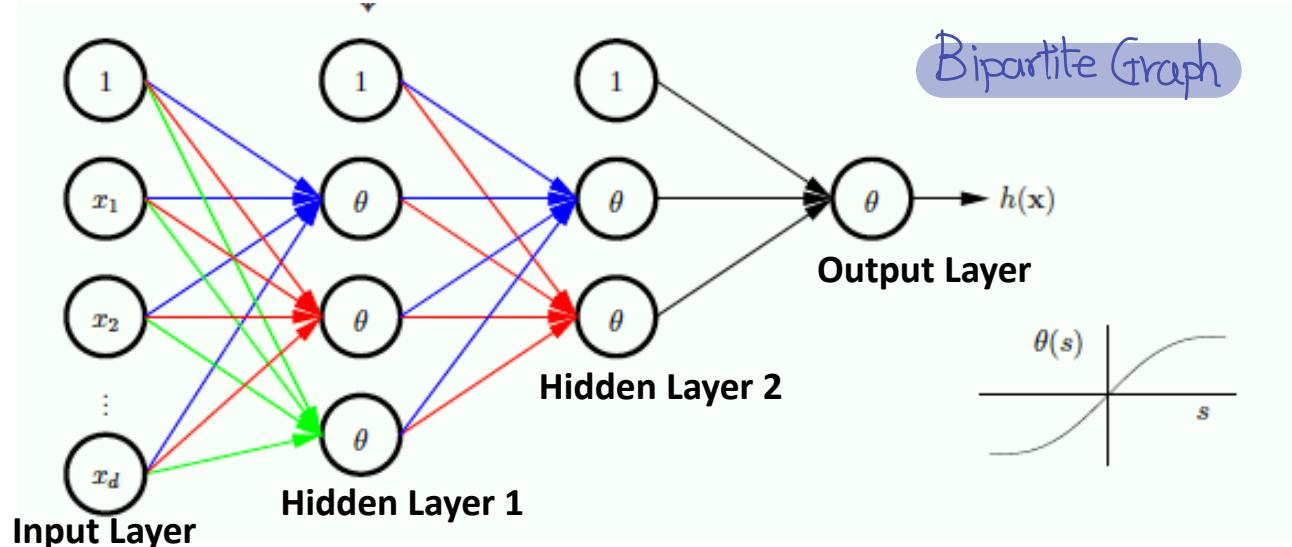
- Quadratic Programming: Maximizing Margin in Linear Classification
- Lagrange Duality Framework for identifying Support Vectors

SVMs with Non-Linear Transforms



- Radial Basis Functions (RBF)
$$\Phi(\mathbf{x}) = [\exp \{-\gamma(\|\mathbf{x} - \mathbf{x}_n\|^2)\}]_{1 \leq n \leq N}$$
- Kernel Trick for Efficient Computation
- Bounds on Generalization Error

Neural Networks - Architecture



$$h_i^{(1)} = \theta \left(\sum_{j=0}^d w_{ji}^{(1)} \cdot x_j \right)$$

Output 'i'
(Hidden Layer 1)
Weights: Layer 1
Inputs

$$h_i^{(2)} = \theta \left(\sum_{j=0}^d w_{ji}^{(2)} \cdot h_j^{(1)} \right)$$

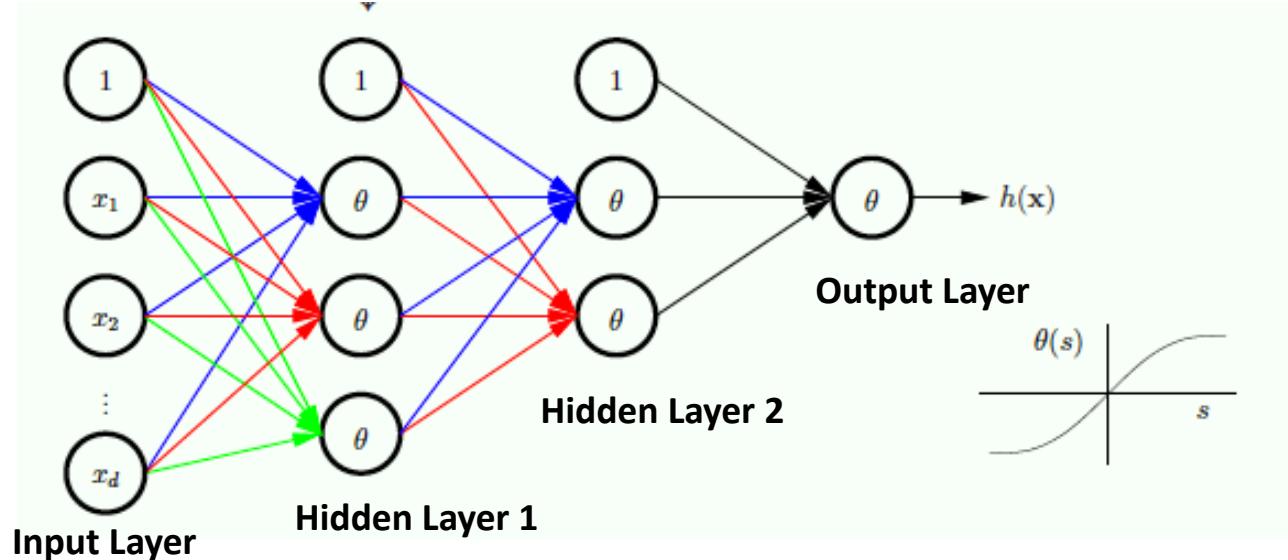
Output 'i'
(Hidden Layer 2)
Weights: Layer 2

$$y = sign \left(\sum_{j=0}^d w_{ji}^{(3)} \cdot h_j^{(2)} \right)$$

Output

Θ : Non Linear Activation Function

Neural Networks - Architecture



Vector Form

$$y = \text{sign} [W_3^T \left\{ \theta \left(W_2^T \left\{ \theta (W_1^T \cdot \mathbf{x}) \right\} \right) \right\}]$$

Universal Approximation Theorem: A Neural Network with **one hidden layer** can approximate any “reasonable” non-linear function with sufficiently many hidden units.

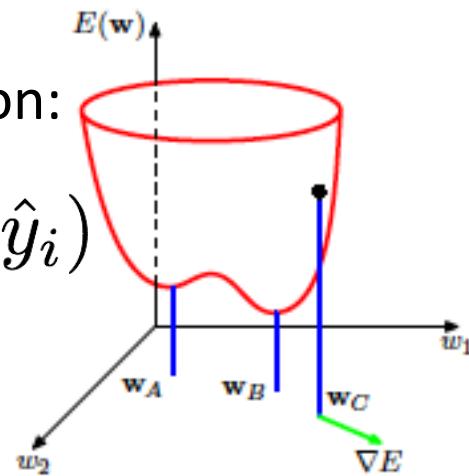
Neural Network - Learning

- Given Training Samples: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$
- Fix:

- Number of Hidden Units per layer
- Nonlinearity: θ

- Learn: Weights: $w_{i,j}^k$
- Minimize Loss Function:

$$E(\mathbf{w}) = \sum_{i=1}^n \ell_{\mathbf{w}}(y_i, \hat{y}_i)$$

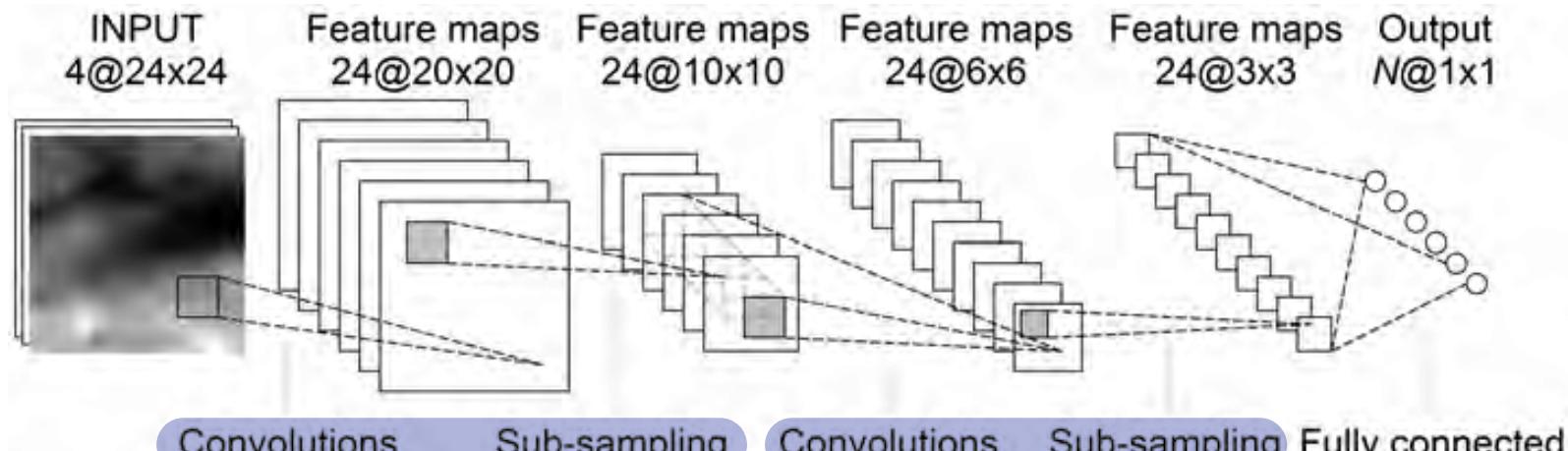


Training Procedure

SGD

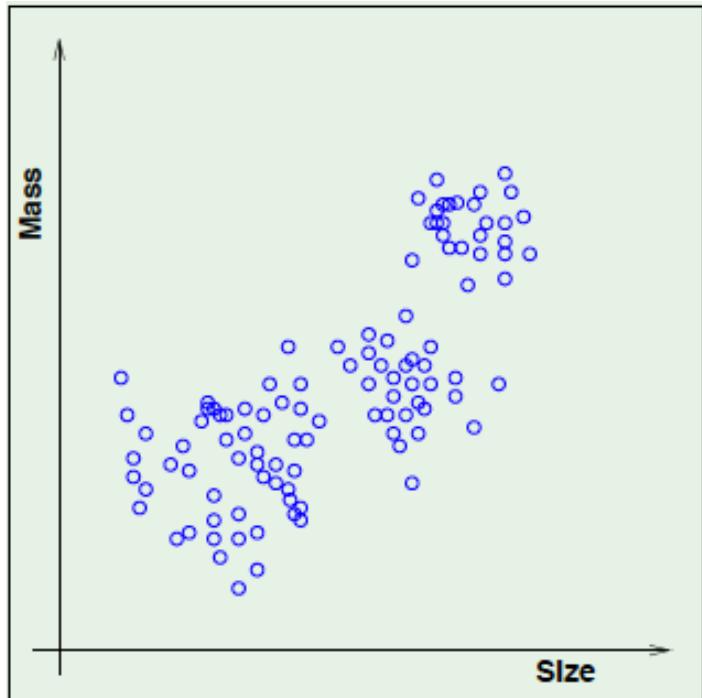
- Stochastic Gradient Descent
- Backpropagation Algorithm
- Early Stopping Rule
- Dropout and Regularization
- Convolutional Neural Networks. CNN
Computer Vision Task

Convolutional Neural Networks



- Image statistics are translation invariant (objects and viewpoint translates)
- Expect low-level features to be local (e.g. edge detector)
- Expect high-level features learned to be coarser

Unsupervised Learning



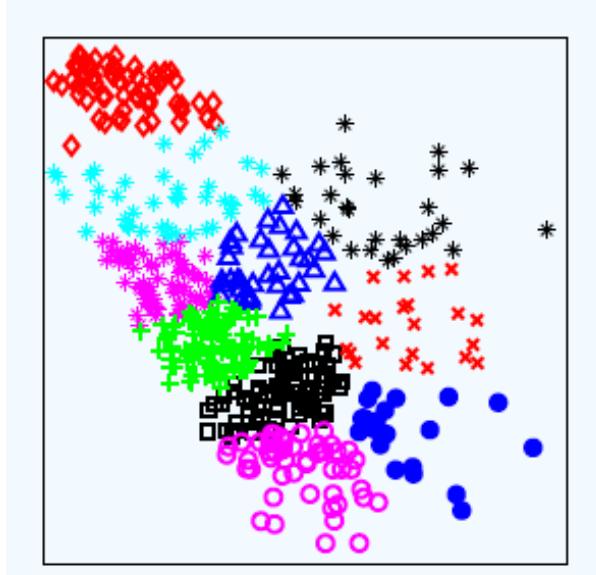
Training Data:

$$\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$$

No Labels!

Still Need to do Learning!

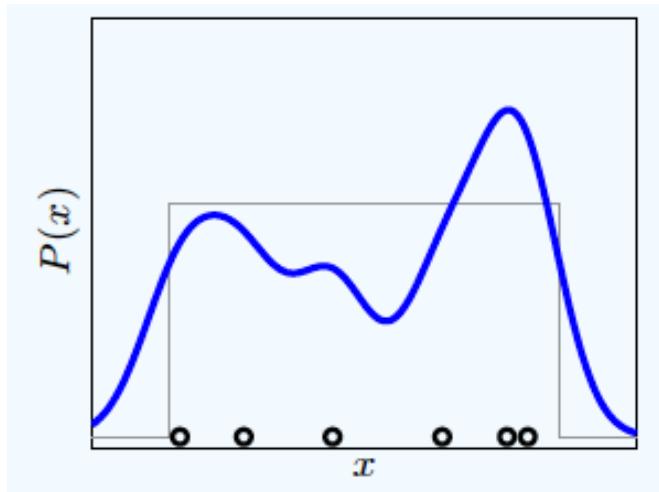
k-Means Clustering



- Cluster Centers: $\mu_1, \mu_2, \dots, \mu_k$
- Partitions: S_1, S_2, \dots, S_k
- Minimize :
$$\sum_{j=1}^N \|\mathbf{x}_j - \mu(\mathbf{x}_j)\|^2$$

Lloyd's Algorithm

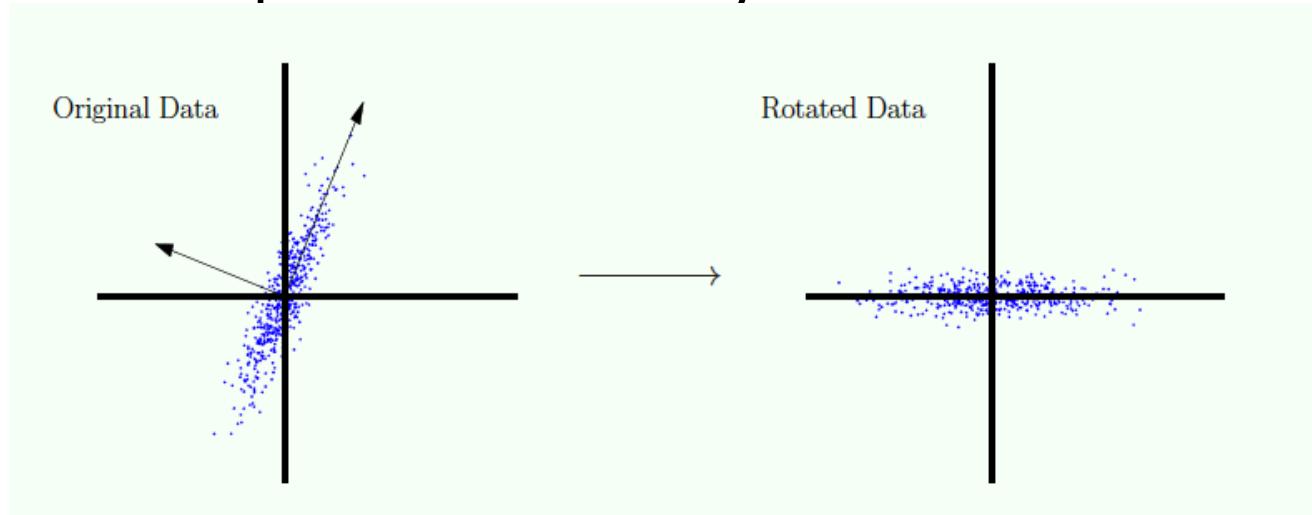
Density Estimation - Gaussian Mixture Models



$$f(\mathbf{x}) = \sum_{j=1}^k w_j \cdot \mathcal{N}(\mathbf{x}; \mu_j, \Sigma_j)$$

- Estimating Probability Density Function
- Expectation Maximization Algorithm (EM) for GMMs
- General EM (time permitting) for Maximum Likelihood solution

Principal Component Analysis



- Dimensionality Reduction
- Identify directions with large variance
- Singular Value Decomposition
- Non-Linear Extension: AutoEncoders

Theory

Training and Testing Errors

- Training Data: $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)$
- Ground Truth (Not Known): $y = f(\mathbf{x})$
- Output of Learning: $\hat{y} = h(\mathbf{x})$ Prediction
- Training (In Sample) Error

$$E_{\text{in}}(h) = \frac{1}{N} \sum_{i=1}^N \mathbb{I}[y_i \neq h(\mathbf{x}_i)]$$

- Testing (Out of Sample) Error

New Data Point

$$E_{\text{out}}(h) = E_{\mathbf{x}, y} [\mathbb{I}[y \neq h(\mathbf{x})]]$$

Probably Approximately Correct (PAC) Learning

- Fixed Hypothesis Class

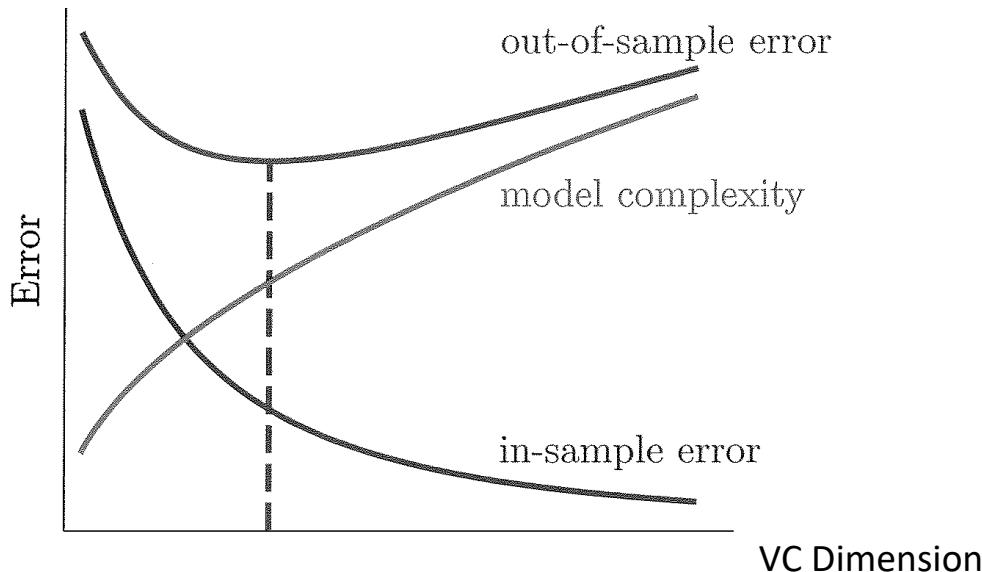
$$\mathcal{H} = \{h_1, h_2, \dots, h_M\}, \quad h \in \mathcal{H}$$

- The following bound holds with probability: $1 - \delta$
- $E_{\text{out}}(h) \leq E_{\text{in}}(h) + \sqrt{\frac{1}{2N} \log \frac{2M}{\delta}}$
 - Complexity of Hypothesis Class
 - Number of Training Examples

$$|\mathcal{H}| = \infty \implies M \leftarrow \text{VC Dimension}(\mathcal{H})$$

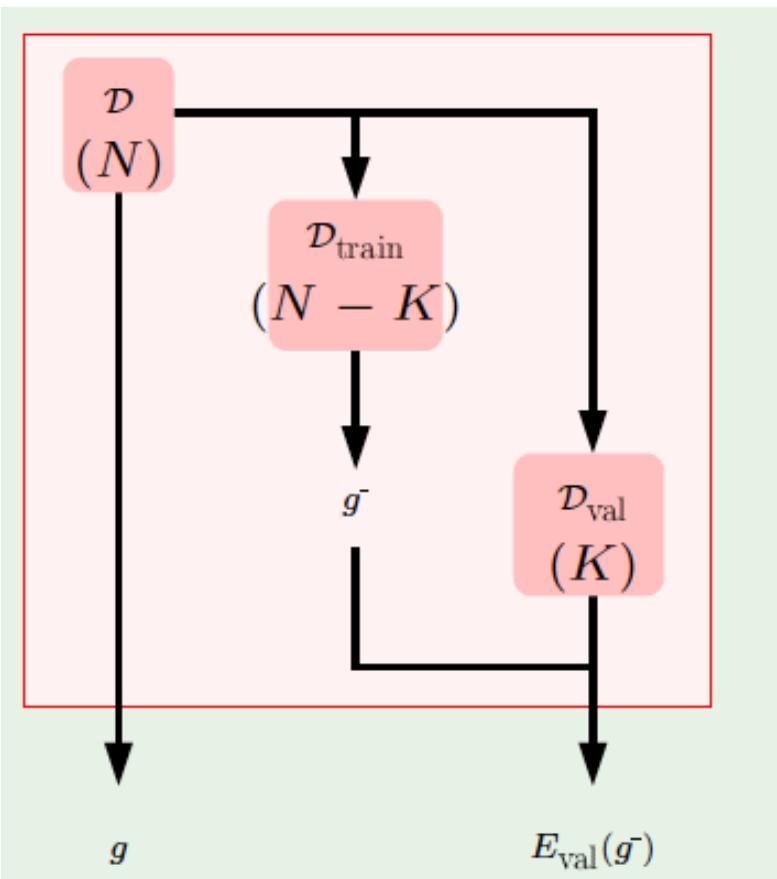
Vapnik Chervonenkis (VC) dimension

- VC Dimension provides a natural measure for complexity of class
- Linear Models: $\text{VC Dimension} = \text{dimension} + 1$
- Neural Networks (roughly) = # of Weight Parameters



Techniques

Validation



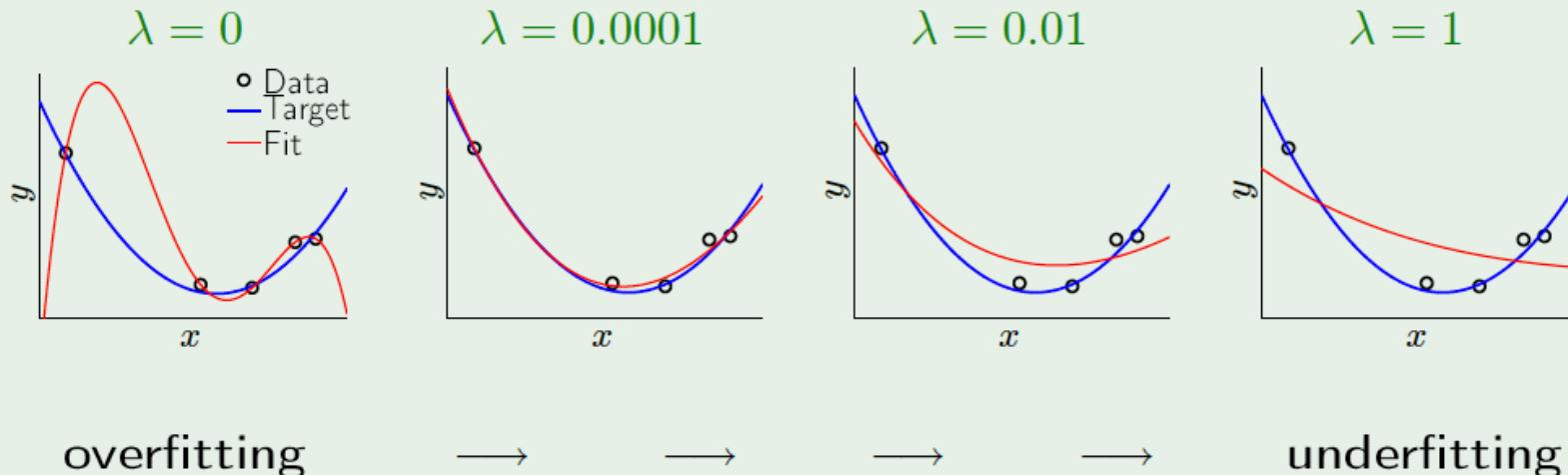
- Divide Training Set into 2 Parts
 - Learning Set
 - Validation Set
- Learning Set: Train Model
- Validation Set: Estimate Test Error
- Applications of Validation
 - Model Selection
 - Selection of Hyperparameters (e.g., regularization coefficient)
 - Number of Training Steps (Early Stopping)
- Cross Validation

Useful Technique In Practice

Regularization

$$y = \mathbf{w}^T \cdot \mathbf{z}, \quad \mathbf{z} = [1 \ x \ \dots, x^M]^T$$

Minimizing $E_{\text{in}}(\mathbf{w}) + \frac{\lambda}{N} \mathbf{w}^T \mathbf{w}$ for different λ 's:



- Weight Decay Method
- Neural Networks: Drop Out Method

Logistics

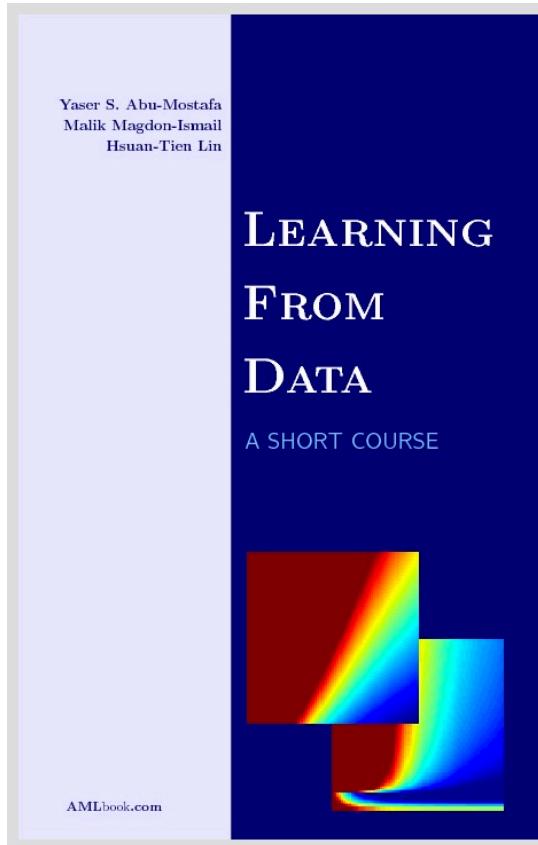
Course Staff

Instructors:

- Ashish Khisti (Section 1)
- Christopher Lucasius (Section 2)

- Tutorials (Mandatory)
 - See calendar timing

Course Textbook



- **Required Textbook:**
- “Learning from Data” Available at U of T bookstore
 - <http://www.amlbook.com>
 - Supplementary Chapters and Appendices
 - Slides from other courses (Caltech, RPI etc)
 - Video Lectures (Prof. Abu-Mostafa)
 - Discussion Forum
- Deep Learning by Goodfellow et.al (Free Online book)
- Recommended Textbook
 - Machine Learning and Pattern Recognition by Christopher Bishop (Text for CSC411)

Tentative Schedule

Week	Description
Week 1(Jan 9)	Intro/Linear Classification
Week 2 (Jan 16)	Linear Regression, Regularization
Week 3 (Jan 23)	Logistic Regression
Week 4 (Jan 30)	Gradient Descent
Week 5 (Feb 6)	Multiplayer Perceptron, Backpropagation
Week 6 (Feb 13)	Deep Learning
Week 7 (Feb 27)	Unsupervised Learning: Clustering and Density Estimation
Week 8 (Mar 6)	EM Algorithm
Week 9 (Mar 13)	Support Vector Machine
Week 10 (Mar 20)	Support Vector Machine, PAC Learning
Week 11 (Mar 27)	PAC Learning, Bias-Variance Tradeoff
Week 12 (Apr 3)	Validation, Cross-Validation

Pre-requisites

- Undergraduate Course in Probability (Official Pre-Req.)
 - Bayes Theorem, Union Bound, Gaussian Distributions
- Linear Algebra (Strongly Recommended)
 - Vector Space Concepts, Matrices
- Programming
 - We will use Python and Tensor Flow Package for our assignments

Grade Composition

- Homework: 10%

Midterm: 25% *March 6th, 2023*

Programming assignments: $7.5\% * 4 = 30\%$

Final exam: 35%

Learning Outcomes

- Fundamentals: basic theory and the fundamental algorithms
- Analysis: ML algorithms for classification, regression and unsupervised learning.
- Algorithm Design: using computational toolboxes of machine learning

ECE421 Course Description

An Introduction to the basic theory, the fundamental algorithms, and the computational toolboxes of machine learning. The focus is on a balanced treatment of the practical and theoretical approaches, along with hands on experience with relevant software packages. Supervised learning methods covered in the course will include: the study of linear models for classification and regression, neural networks and support vector machines. Unsupervised learning methods covered in the course will include: principal component analysis, k-means clustering, and Gaussian mixture models. Theoretical topics will include: bounds on the generalization error, bias-variance tradeoffs and the Vapnik-Chervonenkis (VC) dimension. Techniques to control overfitting, including regularization and validation, will be covered

Machine Learning

Computational system whose performance improves with accumulated data

Ex, recommendation systems (Netflix)

Given a list of movies and partial ratings given by users, you want to predict how a user will rate a movie he/she has not seen

Approach 1:

Based on human experts

An expert view all movies & categorize them

1. Actors
2. Blockbusters
3. Comedy / Actions

Interview each users

1. Like comedy/actions?
2. Favorite actors
3. Like blockbusters?

Tedious process, expensive, user preferences can change.

Approach 2:

Machine Learning

Visualize Dataset

users	1	2	3	4	5
1		r_{12}	r_{13}		
2	r_{21}	r_{22}			r_{25}
3	r_{31}			r_{34}	
4			r_{43}		
5					

r_{ij} , rating given by user i for movie j (if available)

S = set of available rating

$$= \{(i,j) : r_{ij} \text{ is available in our dataset}\}$$

$$S = \{(1,2), (1,3), (2,1), (2,2), (2,5), \dots\}$$

Model (Cannot interpret the weights)

1) for each user i , user preference vector

$$x_i = (x_{i1}, x_{i2}, \dots, x_{id}) \in \mathbb{R}^d \xrightarrow{\text{Hyperparameter}}$$

2) for each movie j , movie attribute vector

$$y_j = (y_{j1}, y_{j2}, \dots, y_{jd}) \in \mathbb{R}^d$$

Function to compute an estimate for the rating

given x_i & y_j

Estimated Rating for $\hat{r}_{ij} = x_i^\top y_j = \sum_{l=1}^d x_{il} y_{jl}$

User i and Movie j

Model Parameters (Ω)

$$\Omega = \{\{x_i\}_{i \in U}, \{y_j\}_{j \in M}\}$$

U : set of users

M : set of movies

Training

Learn model parameters Ω given the training set

Define loss function (Training Error)

$$E_{in}(\Omega) = \sum_{(i,j) \in S} (r_{ij} - \hat{r}_{ij})^2$$

Error In Sample

$$x_i^T y_j$$

We will select Ω to minimize $E_{in}(\Omega)$

$$\Omega^* = \underset{\Omega}{\operatorname{argmin}} E_{in}(\Omega)$$

Optimal Model Parameter (Ω^*)

Training Algorithm, Gradient Descent

EM Algorithm

Prediction Function, $(i,j) \notin S$, $\hat{r}_{ij} = x_i^T y_j$

$$\Omega^* = \{\{x_i\}_{i \in U}, \{y_j\}_{j \in M}\}$$