

# ECE421

# Intro. to Machine Learning

Winter 2022

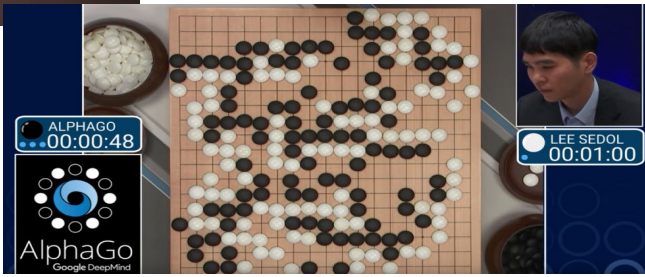
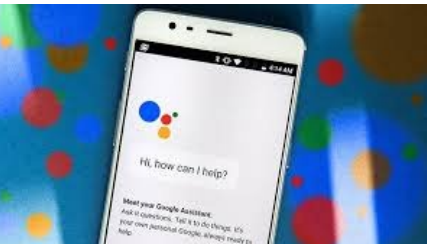
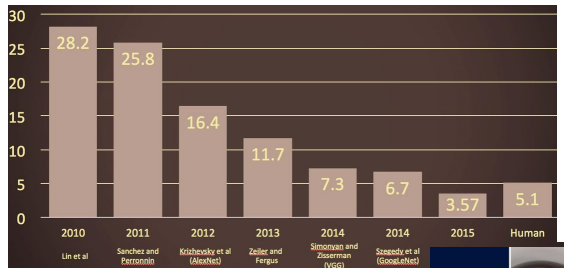
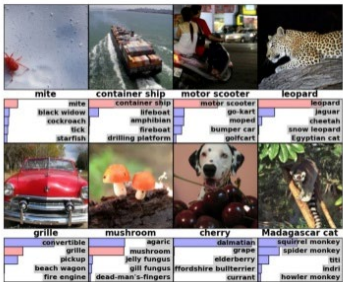
Ben Liang

Ashish Khisti

# ImageNet Challenge

## IMAGENET

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



## Natural language processing

English Spanish French Spanish - detected

English Spanish Arabic

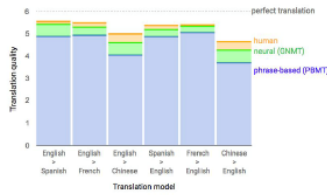
Translate

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

En un lugar de la Mancha, de cuyo nombre no quiero acordarme, no ha mucho tiempo que vivía un hidalgo de los de lanza en astillero, adarga antigua, rocín flaco y galgo corredor. Una olla de algo más vaca que carnero, salpicón las más noches, duelos y quebrantos los sábados, lantejas los viernes, algún palomino de añadidura los domingos, consumían las tres partes de su hacienda. El resto della concluían sayo de velarte, calzas de velludo para las fiestas, con sus pantuflos de lo mismo, y los días de entresemana se honraba con su vellorí 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 rocín como tomaba la podadera. Frisaba la edad de nuestro hidalgo con las cincuenta años; era de complexión recia, 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 conjeturas verosímiles, se deja entender que se llamaba Quijana. Pero esto importa poco a nuestro cuento; basta que en la narración del no se saiga un punto de la verdad.

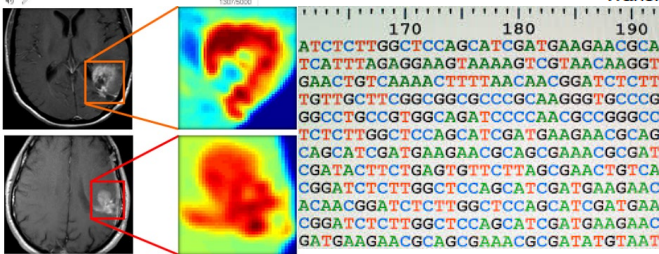
First chapter. Which deals with the condition and exercise of the famous nobleman Don Quixote de la Mancha

In a place of La Mancha, whose name I do not want to remember, there has not been a long time that lived a lord of the lance in armour, old pork, thin rocin and greyhound runner. A pot of something more cow than ram, split most nights, duels and breaks on Saturdays, gilets on Fridays, some palomino in addition to Sundays, consumed the three parts of his estate. The rest of the party concluded a velvet dress, hairy lights for the parties, with their slippers of the same, and the days of midweek were honored with their vellorí of the finest. He had a housekeeper in his house who was in his forties, and a niece who was not in his twenties, and a boy in the country and square, who saddled the rocin as he took the pruning. He emphasized the age of our hidalgo at the age of fifty. Was of a hard complexion, dry of flesh, thin of face, great early bird and friend of the hunt. They mean that he had the nickname of Quijada, or Quesada, that in this there is some difference in the authors who in this case write: Although, by plausible conjectures, it is understood that it was called Quijana. But this matters little to our story: It is enough that in the narration of him a point of truth does not come out.



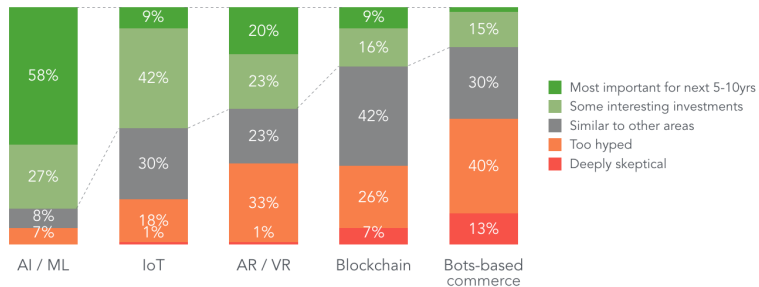
## Google's Neural Machine Translation (Wu et al. 2016)

## DIAGNOSTICS



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?

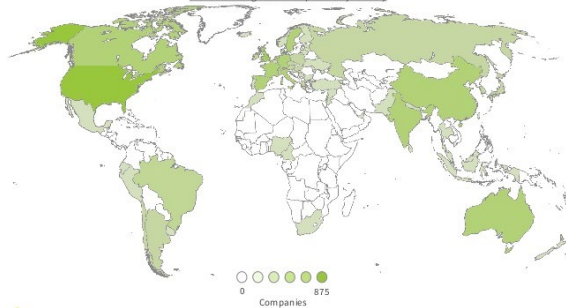


17 Source: Upfront Ventures survey of VCs (N=115), Jan 2017

upfront

Artificial intelligence startups are a global phenomenon

Artificial Intelligence Startup Count by Country

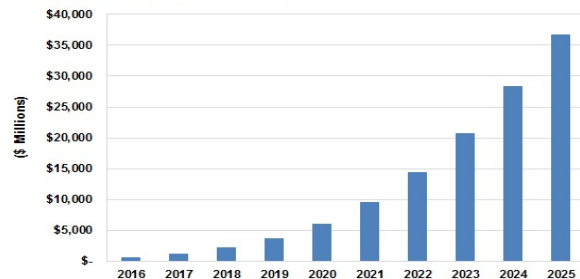


Venture Scanner

Data as of April 2017 13

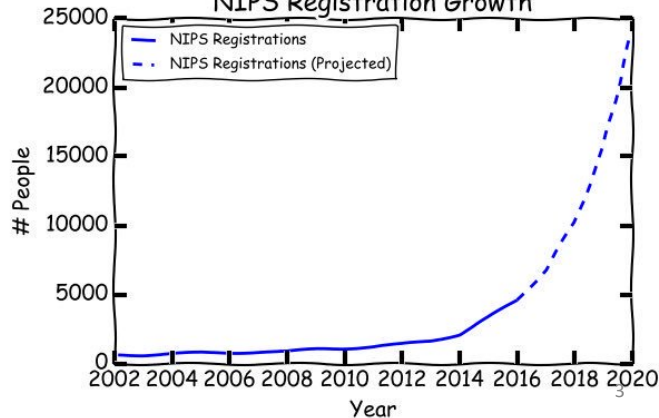
Tractica

Artificial Intelligence Revenue, World Markets: 2016-2025



Source: Tractica

NIPS Registration Growth



# Machine Learning Jargon

semi-supervised learning    **overfitting**    stochastic gradient descent    **SVM**    *Q learning*  
Gaussian processes    **deterministic noise**    **data snooping**  
*distribution-free*    *linear regression*    VC dimension    learning curves  
collaborative filtering    **decision trees**    nonlinear transformation    **sampling bias**    *neural networks*    mixture of experts  
**active learning**    *RBF*    *training versus testing*    noisy targets    *Bayesian prior*  
*ordinal regression*    **linear models**    bias-variance tradeoff    **weak learners**  
**ensemble learning**    **cross validation**    logistic regression    **data contamination**  
exploration versus exploitation    error measures    types of learning    perceptrons    **hidden Markov models**  
*clustering*    **is learning feasible?**    *kernel methods*    graphical models  
regularization    weight decay    **soft-order constraint**    *Occam's razor*    *Boltzmann machine*

# Machine Learning

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

# Overview

## Learning Methods

### Supervised Learning

- Linear Models
- Neural Networks
- Support Vector Machines

### Unsupervised Learning

- Clustering
- Density Estimation
- EM algorithm

## Learning Theory

- PAC Learning
- VC Dimension
- Bias Variance Tradeoff

## Learning Principles

- Regularization
- Validation

# Linear Classification

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

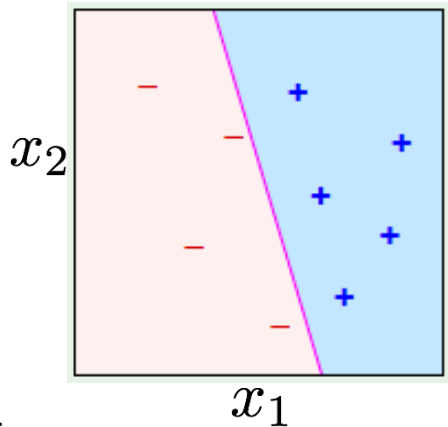
$$\mathbf{x}_i \in \mathbb{R}^d, \quad y_i \in \{\pm 1\}$$

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

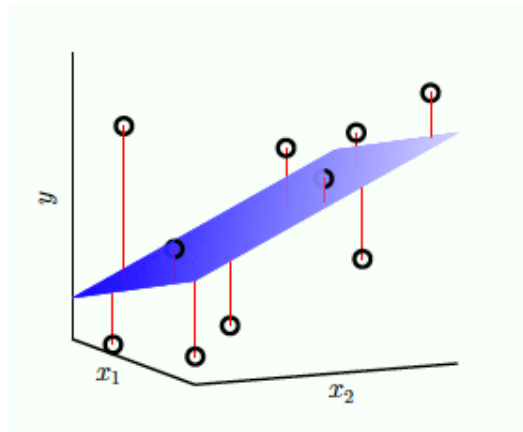
$$\mathbf{x}_i \in \mathbb{R}^d, \quad y_i \in \mathbb{R}$$

- **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:

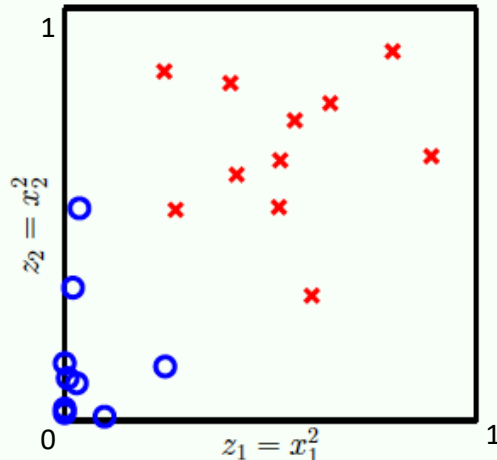
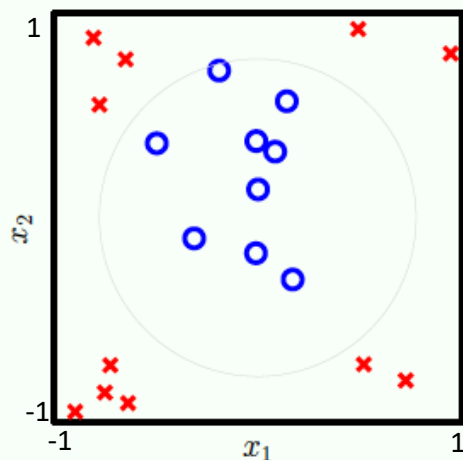
$$\sum_{i=1}^N (y_i - \hat{y}_i)^2$$

- Least Squares and its variations





# Non-Linear Transformations of Data

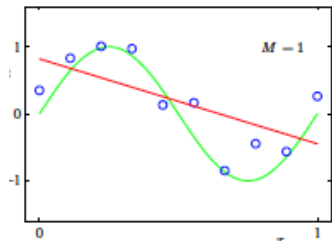


$$\mathbf{x} = \begin{bmatrix} 1 \\ x_1 \\ x_2 \end{bmatrix} \Rightarrow \mathbf{z} = \begin{bmatrix} 1 \\ x_1^2 \\ x_2^2 \end{bmatrix}$$

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

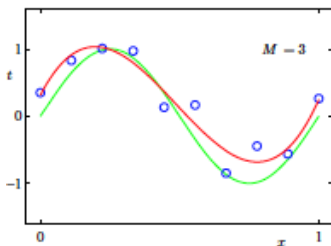
Nonlinear Transforms, Feature Vectors, Kernel Methods

# Non-Linear Transformations of Data



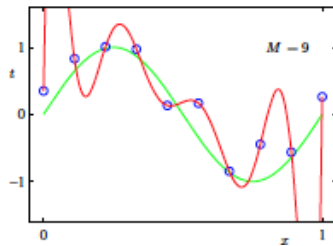
$$\mathbf{z} = [1 \quad x]^T$$

First Order Polynomial



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

Third Order Polynomial



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

9<sup>th</sup> Order Polynomial

Under-fitting

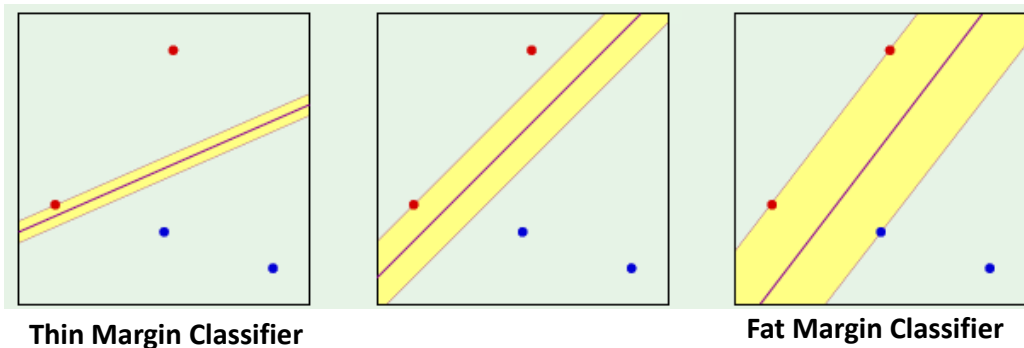
$$\hat{y} = \mathbf{w}^T \mathbf{z}$$

Overfitting

- Higher Order Non-Linearity:

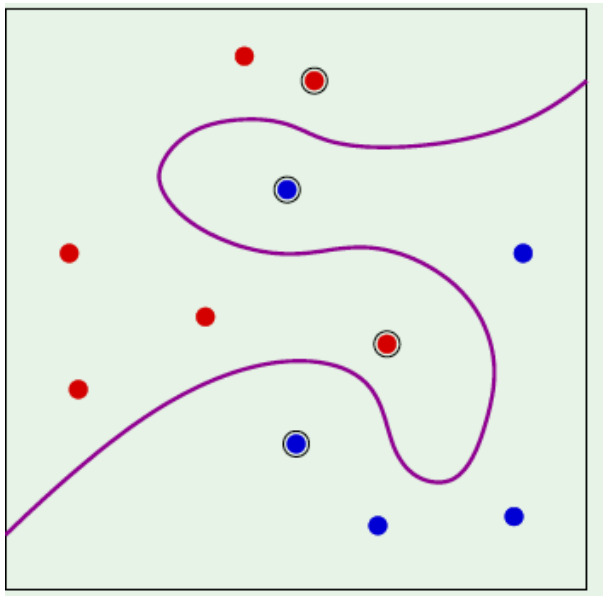
- Better Fit to Training Data
- Less Robust to Noise (Overfitting)

# Support Vector Machines



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

# SVMs with Non-Linear Transforms

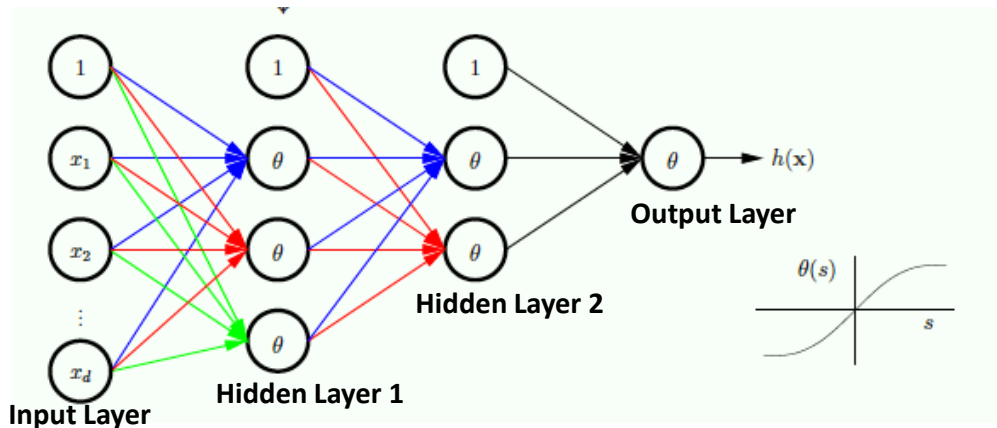


- Radial Basis Functions

$$\Phi(\mathbf{x}) = [\exp \{ -\gamma (||\mathbf{x} - \mathbf{x}_n||^2) \}]_{1 \leq n \leq N}$$

- 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

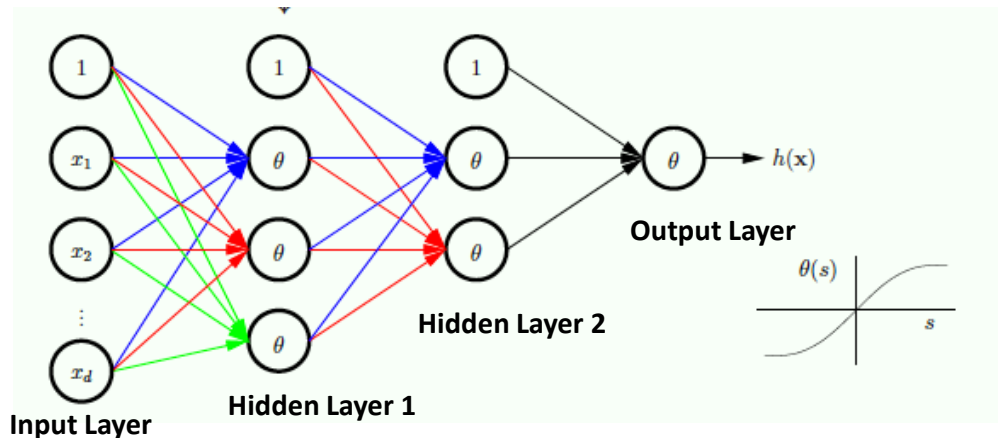
$$h_i^{(2)} = \theta \left( \sum_{j=0}^d w_{ji}^{(2)} \cdot h_j^{(1)} \right)$$

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

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

Output

# Neural Networks - Architecture



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

**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:**

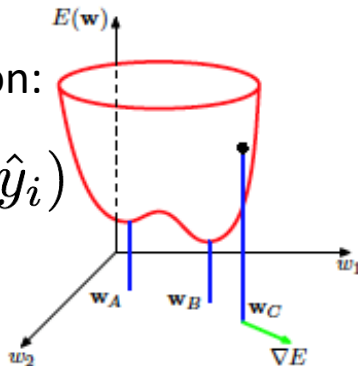
- Number of Hidden Units per layer

- Nonlinearity:  $\theta$

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

- **Minimize** Loss Function:

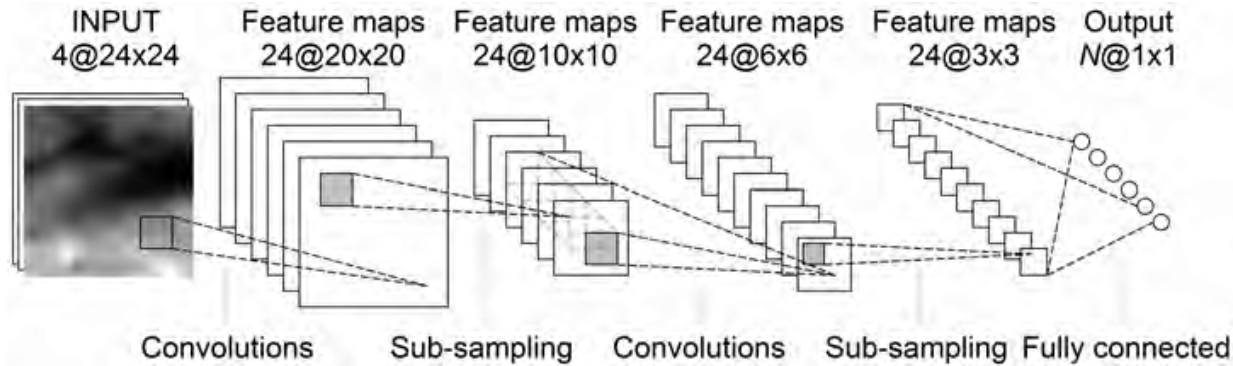
$$E(\mathbf{w}) = \sum_{i=1}^n \ell_{\mathbf{w}}(y_i, \hat{y}_i)$$



## Training Procedure

- Stochastic Gradient Descent
- Backpropagation Algorithm
- Early Stopping Rule
- Dropout and Regularization
- *Convolutional Neural Networks.*

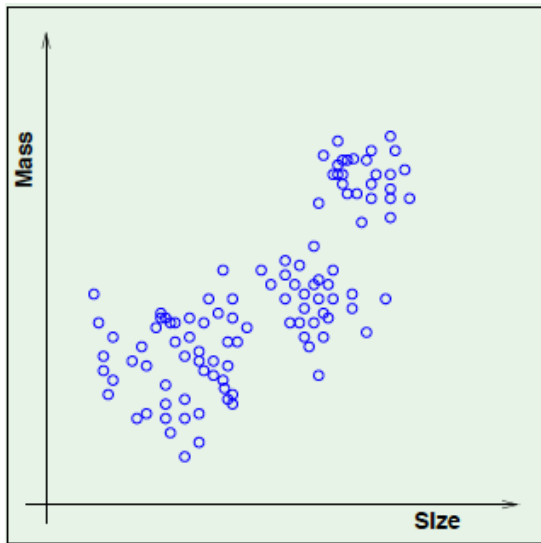
# Convolutional Neural Networks



- 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



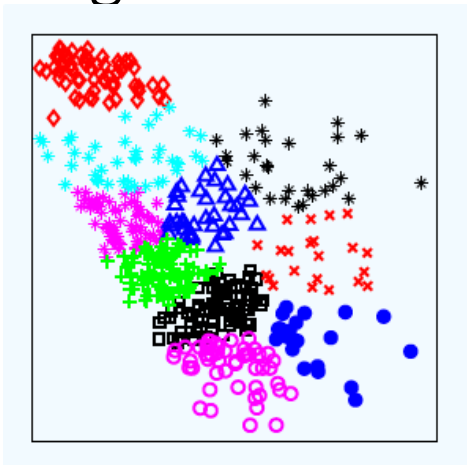
Training Data:

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

No Labels!

Still Need to do Learning!

# k-Means Clustering



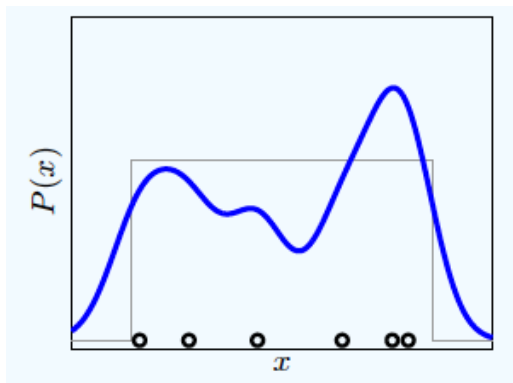
- Cluster Centers:  $\mu_1, \mu_2, \dots, \mu_k$

- Partitions:  $S_1, S_2, \dots, S_k$

- Minimize : 
$$\sum_{j=1}^N \|\mathbf{x}_j - \mu(\mathbf{x}_j)\|^2$$

Lloyd's Algorithm

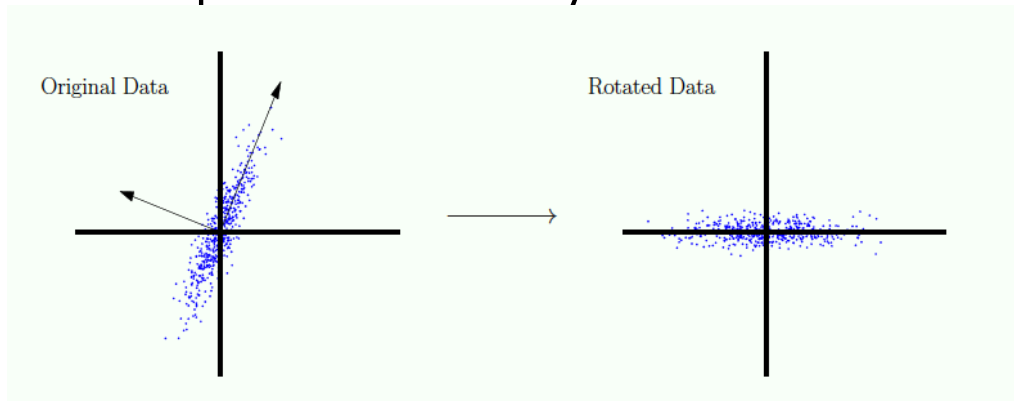
# Density Estimation - Gaussian Mixture Models



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

- 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:  $\hat{y} = h(\mathbf{x})$

- 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

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


# 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:  $1 - \delta$

- $$E_{\text{out}}(h) \leq E_{\text{in}}(h) + \sqrt{\frac{1}{2N} \log \frac{2M}{\delta}}$$

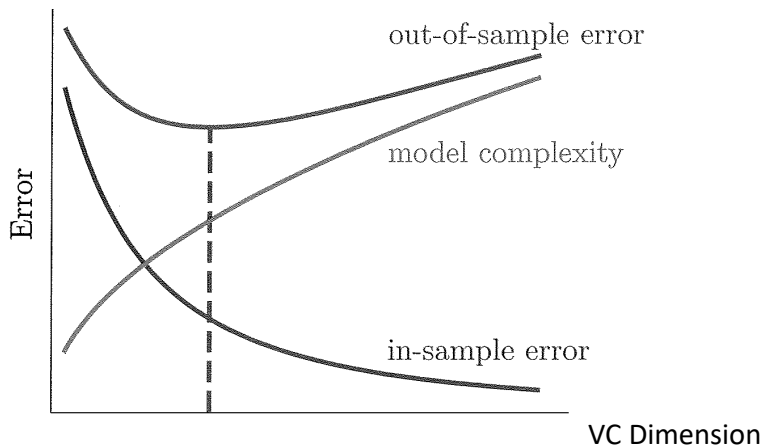
  
Complexity of Hypothesis Class

  
Number of Training Examples

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

# Vapnik Chervonenkis (VC) dimension

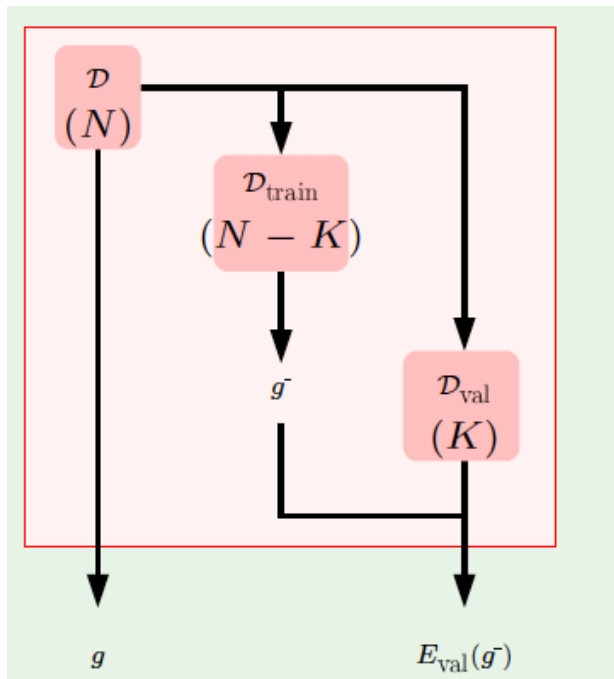
- VC Dimension provides a natural measure for complexity of class
- Linear Models: VC Dimension = dimension + 1
- Neural Networks (roughly) = # of Weight Parameters





# Techniques

# Validation

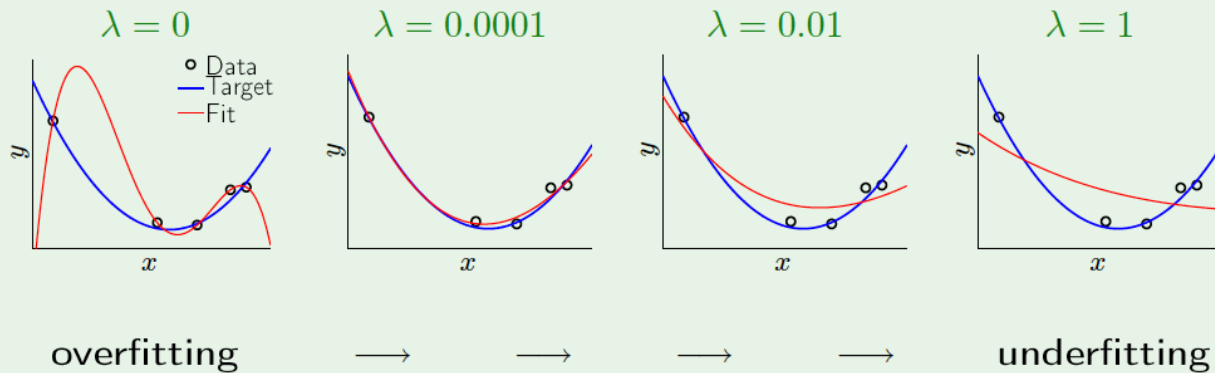


- Divide Training Set into 2 Parts
  - Learning Set
  - Validation Set
- Learning Set: Train Model
- Validation Set: Estimate Test Error
- Applications of Validation
  - Model Selection
  - Selection of Hyperparameters (e.g., regularization coefficient)
  - Number of Training Steps (Early Stopping)
-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

# Regularization

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

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



- Weight Decay Method
- Neural Networks: Drop Out Method

# Logistics

# Course Staff

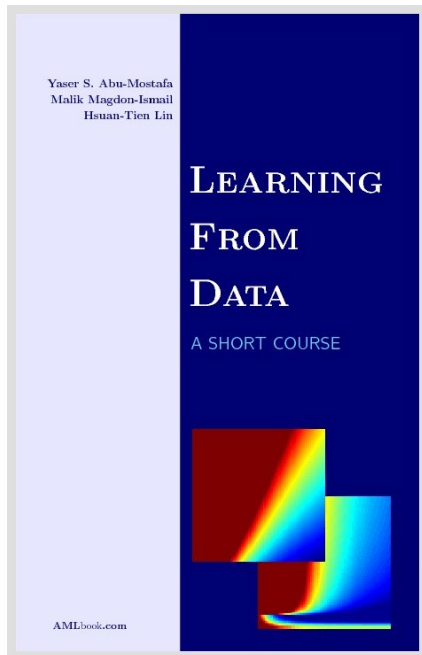
## Instructors:

- Ben Liang (Section 1)
- Ashish Khisti (Section 2)

## • Tutorials

- Tutorial 1: Fridays 9 - 11 am
- Tutorial 2: Fridays 10 - 12 pm
- Tutorial 3: Mondays 12 - 2 pm
- Tutorial 4: Tuesdays 9 - 11 am

# 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

<b>Week of Jan 10</b>	Intro/Linear Classification		
<b>Jan 17</b>	Linear Regression, Regularization	Tut 1	Assignment 1 posted
<b>Jan 24</b>	Logistic Regression	Tut 2	
<b>Jan 31</b>	Gradient Descent	Tut 3	<b>Assignment 1 due</b> , Assignment 2 posted
<b>Feb 7</b>	Multiplayer Perceptron, Backpropagation	Tut 4	
<b>Feb 14</b>	Deep Learning	Tut 5	
<b>Feb 21</b>	<b>Reading Week</b>		
<b>Feb 28</b>	Unsupervised Learning: Clustering and Density Estimation	Tut 6	<b>Assignment 2 due</b> , Assignment 3 posted
<b>Mar 7</b>	EM Algorithm	Tut 7	<b>Midterm Exam</b>
<b>Mar 14</b>	Support Vector Machine	Tut 8	
<b>Mar 21</b>	Support Vector Machine, PAC Learning	Tut 9	
<b>Mar 28</b>	PAC Learning	Tut 10	
<b>Apr 4</b>	PAC Learning, Bias-Variance Tradeoff	Tut 11	<b>Assignment 3 due</b>
<b>Apr 11</b>	Validation, Cross-Validation		Last day of class: Apr 14

# 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

- Mid Term: 20% (March 7<sup>th</sup>)
- Homeworks: 10%
- Programming Assignment ( $3 \times 15\% = 45\%$ )
- Final Exam: 25% in Exam Week
  
- Assignments will be done individually.

# 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

# Tensor flow Assignments

- **Python based** ML Library released by Google in 2015
- Automatic Training for Neural Networks
- GPU Support (Not Required for Assignments in this courses)
- Installation through Anaconda Environment is Recommended (See Installation Guide on Course Webpage)
- Tons of Resources!
  - Tensorflow.Org Tutorials
  - CS231n Stanford Tutorial (<http://cs231n.stanford.edu/>)
  - See Course Webpage for a simple tutorial (Updated, Use Chrome Browser)

## Tensor flow Example (<https://www.tensorflow.org>)

```
import tensorflow as tf
x = tf.placeholder(tf.float32, [None, 784])
W = tf.Variable(tf.zeros([784, 10]))
b = tf.Variable(tf.zeros([10]))
y = tf.nn.softmax(tf.matmul(x, W) + b)
```

Initialize Computational Graph

```
cross_entropy = tf.reduce_mean(-tf.reduce_sum(y_ * tf.log(y), reduction_indices=[1]))
train_step = tf.train.GradientDescentOptimizer(0.5).minimize(cross_entropy)
```

Loss Function  
and Optimizer

```
sess = tf.InteractiveSession()
tf.global_variables_initializer().run()

for _ in range(1000):
    batch_xs, batch_ys = mnist.train.next_batch(100)
    sess.run(train_step, feed_dict={x: batch_xs, y_: batch_ys})
```

Training Routing