Introduction to ML (Part II)

C. V. Jawahar
IIIT Hyderabad

Plans for today

A. Recap and Supervised Learning
B. Intro. to Supervised Learning algorithms
C. Worries During the Learning
D. Learning Paradigms and Optimization

Blank Slide

$\{(x_i, y_i)\}$

$i = 1, \ldots N$

$y \to$ Class

$y \to$ Res.

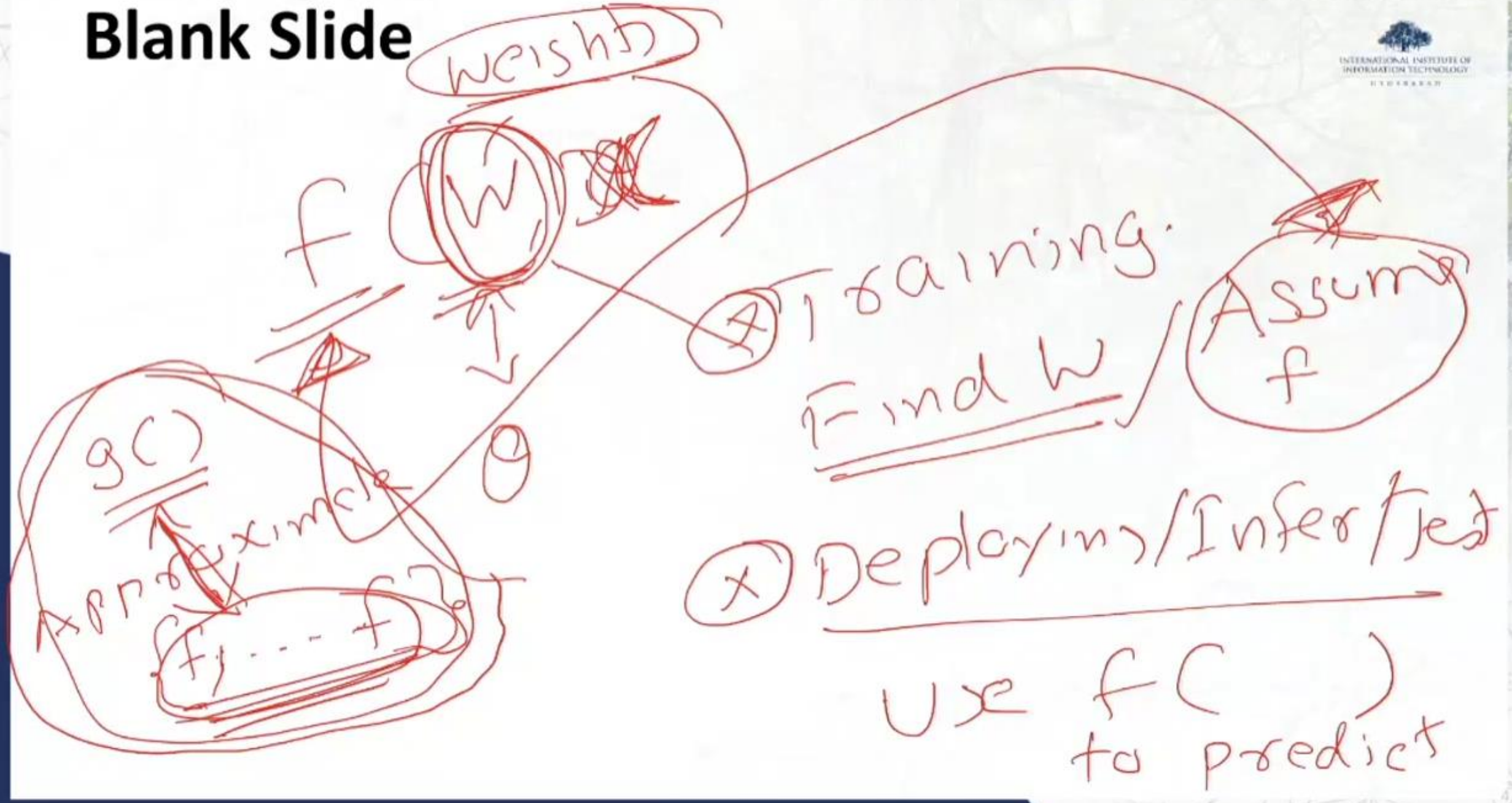
$x \in \mathbb{R}^d$

To

Goal: Find fn $f(\ )$

$f(x_i) \to y_i$ $\forall i$ To beyond N

Blank Slide


weights
W
f
θ
g()
Approximate
f
(f1 ... fn)
Training.
Find W
Assume
f
Deploying/Infer/Test
Use f( )
to predict

Variations

- Binary Classification

$$\mathbf{X} \longrightarrow l \in L = \{l_1, l_2\}$$

x Class
x Regr

- Multi Class Classification

$$\mathbf{X} \longrightarrow l \in L = \{l_1, l_2, \ldots, l_{\|C\|}\}$$

x -> y
x -> y1, y2 ...

- Multi Label Classification

$$\mathbf{X} \longrightarrow l^X = \{l_1^X, l_2^X, \ldots\} \quad l_i^X \in L = \{l_1, l_2, \ldots, l_{\|C\|}\}$$

- Structured Output Prediction

$$\mathbf{X} \longrightarrow y \in \mathcal{Y} = \{1, \ldots, \mathrm{k}\}^l$$

- $\mathcal{Y}$ are complex (structured outputs)
  - Images, text, audio, folds of protein

Problem Space

- **Feature Extraction:** Find ***X*** corresponding to an entity/item ***I*** (such as an image, web page, ECG etc.)

- **Classification:** Find a parameterized function $f_W(X)$ which can make the right predictions ***Y***.

- **End to End:** Can we learn ***Y*** directly from ***I***.

B. Intro. to Supervised Learning

Machine Learning

Popular Problems

- Classification
- Regression
- Density Estimation

Classification

- K Nearest Neigours
- Naïve Bayes Classifier
- Decision Trees
- Random Forest
- Logistic Regression
- Ensemble Learning
- Neural Networks
- Support Vector Machines

The machine learning framework

$$y = f(\mathbf{x})$$

output

prediction function

feature or representation

- **Training:** given a *training set* of labeled examples $\{(\mathbf{x}_1,y_1), \ldots, (\mathbf{x}_N,y_N)\}$, estimate the prediction function f by minimizing the prediction error.
- **Testing:** apply f to a never before seen *test example* $\mathbf{x}$ and output the predicted value $y = f(\mathbf{x})$

Slide credit: L. Lazebnik

Examples of Loss Function

- Very similar: Loss function, Objective function, Error Function

$$error = \frac{1}{N}\sum_{n=1}^{N}[y_n \neq g(x_n)]$$

- Is error the only measure?
- Many Alternatives:
  - Square error loss.
  - Ranking Loss
  - ??

Blank Slide

Consider the Following Problems

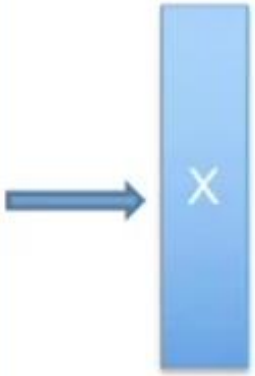
1. Given an email, decide if it is SPAM or not
2. Given the picture of a pet, decide if it is a cat or a dog

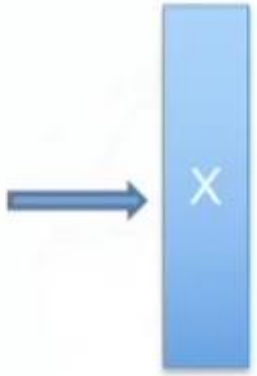
X: Text of email
Y: SPAM / HAM : {1, 0}

X: Image of the pet
Y: Cat / Dog : {1, 0}

- How do we represent the text as a vector?
- What about the image?

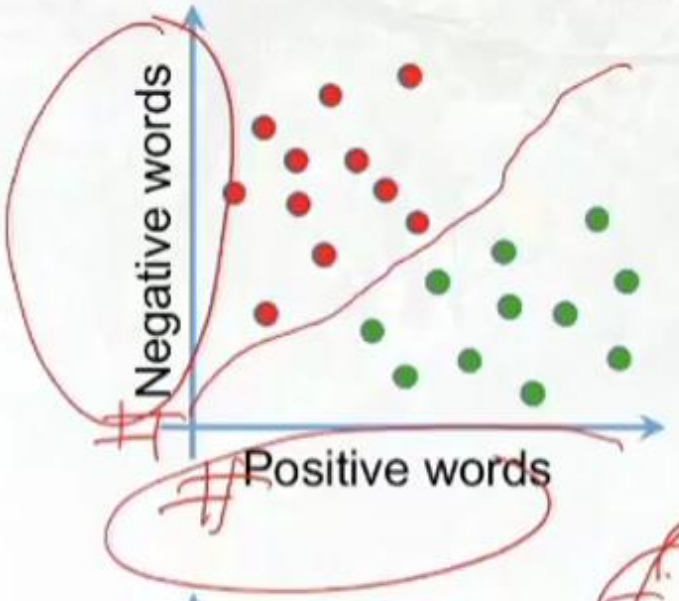
Feature Vector

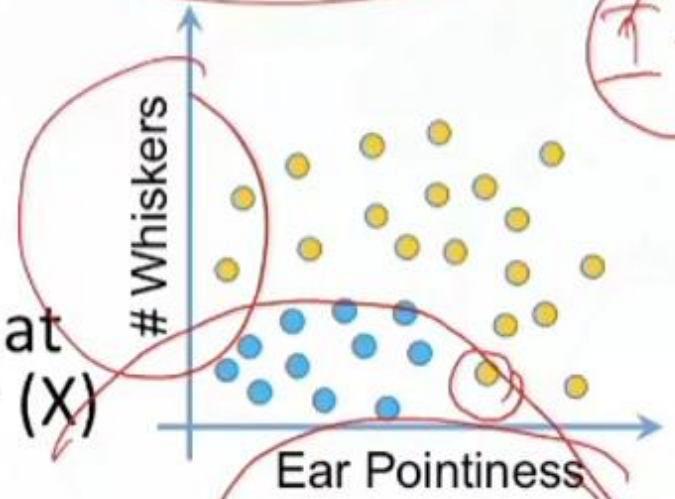




Feature Space

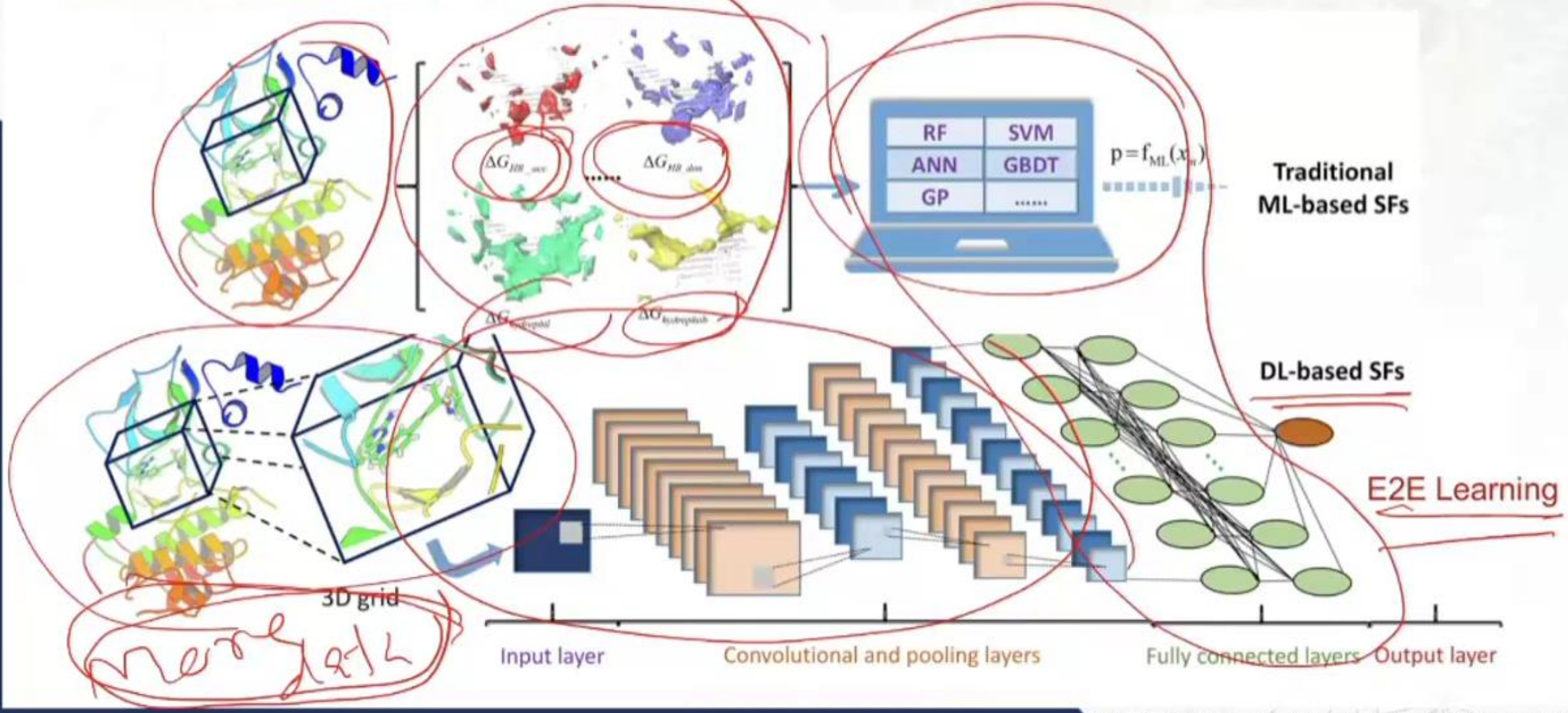
- We need to extract useful information from the input, and avoid 'noise'.
- Email: Count of positive and negative words in the email
- Pet: Pointiness of ears, Number of whiskers
- Now onwards, we assume that a sample is its feature vector (X)





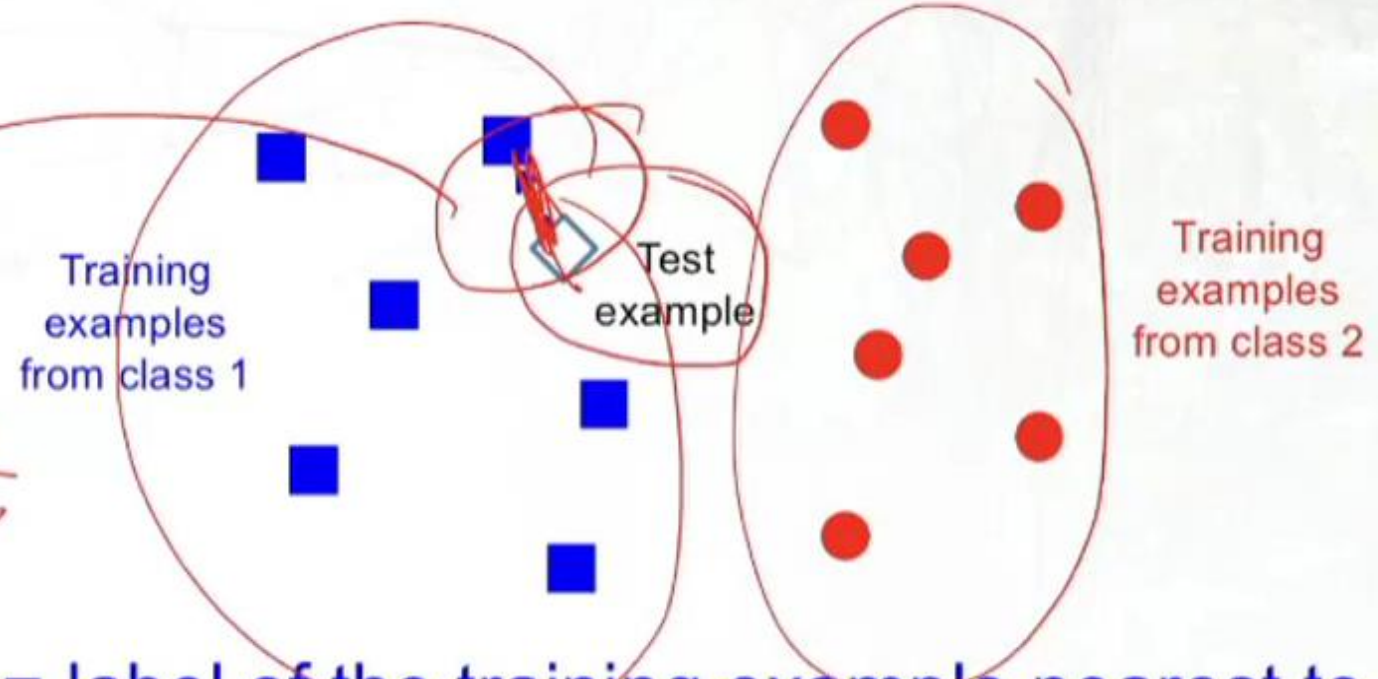
Protein-Ligand Binding Affinity Prediction

Regression: Predict Scoring Function/Scores



WIREs Comp. Mol. Sci. 2019, e1429

Classifiers: Nearest neighbor



Distance

f(x)

lazy learr

f(**x**) = label of the training example nearest to **x**

- All we need is a distance function for our inputs
- No training required!

Slide credit: L. Lazebnik

Classifiers: Linear

$w_1 = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$ $w_1 x_1 + w_2 x_2 + b$

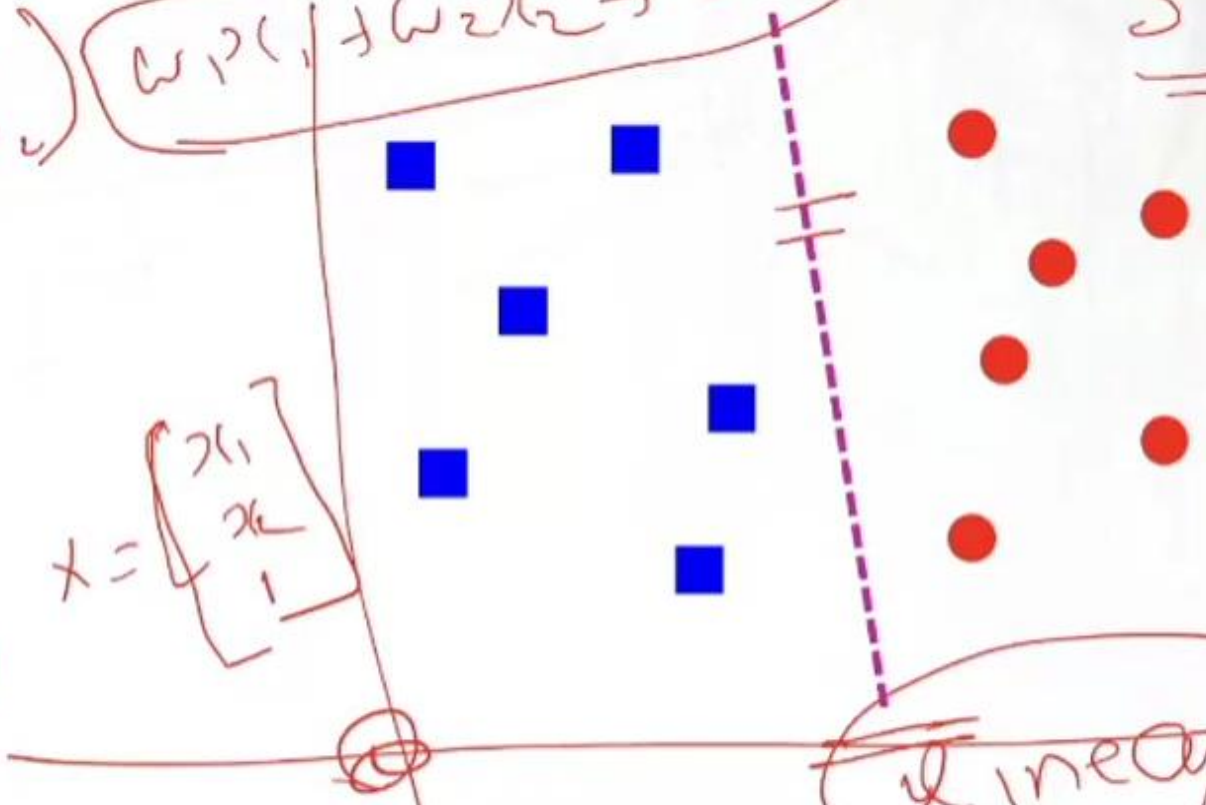
$\text{sign}(w^T x)$

$x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$

$x = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$

$w = \begin{bmatrix} w_1 \\ w_2 \\ b \end{bmatrix}$

$x^T \begin{bmatrix} x \\ 1 \end{bmatrix}^T \begin{bmatrix} w \\ b \end{bmatrix}$



linear

bias

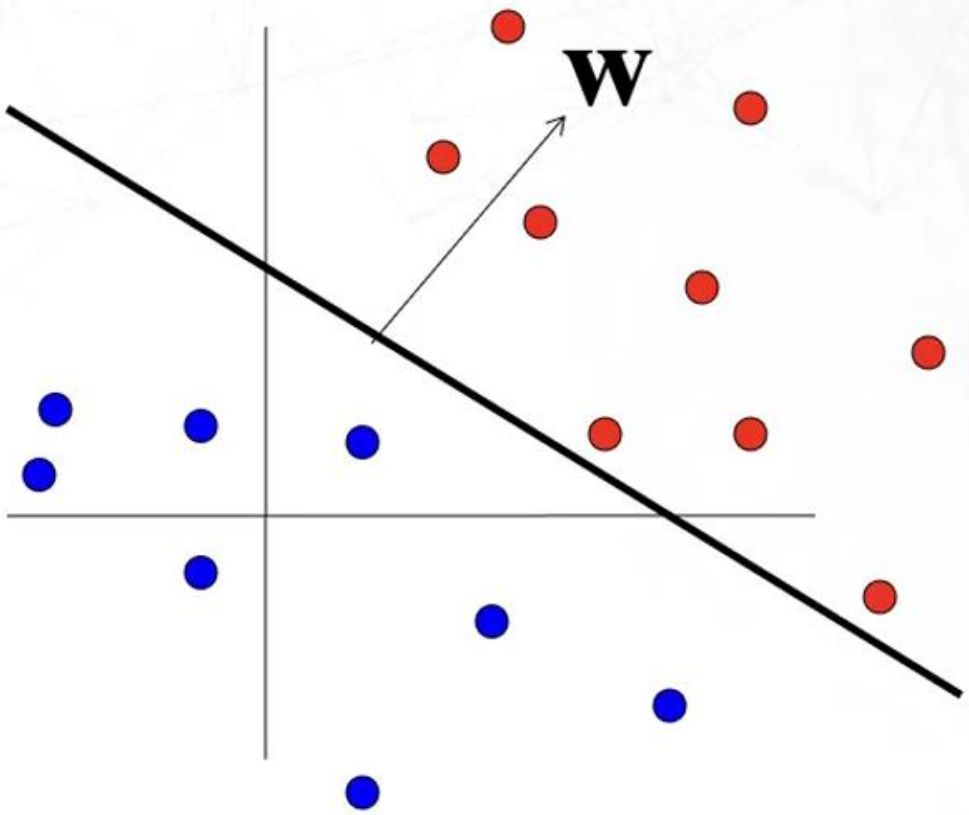
- Find a linear function to separate the classes:

$w^T x$

$$f(\mathbf{x}) = \text{sgn}(\mathbf{w}^T \cdot \mathbf{x} + b)$$

Slide credit: L. Lazebnik

Linear Classifiers

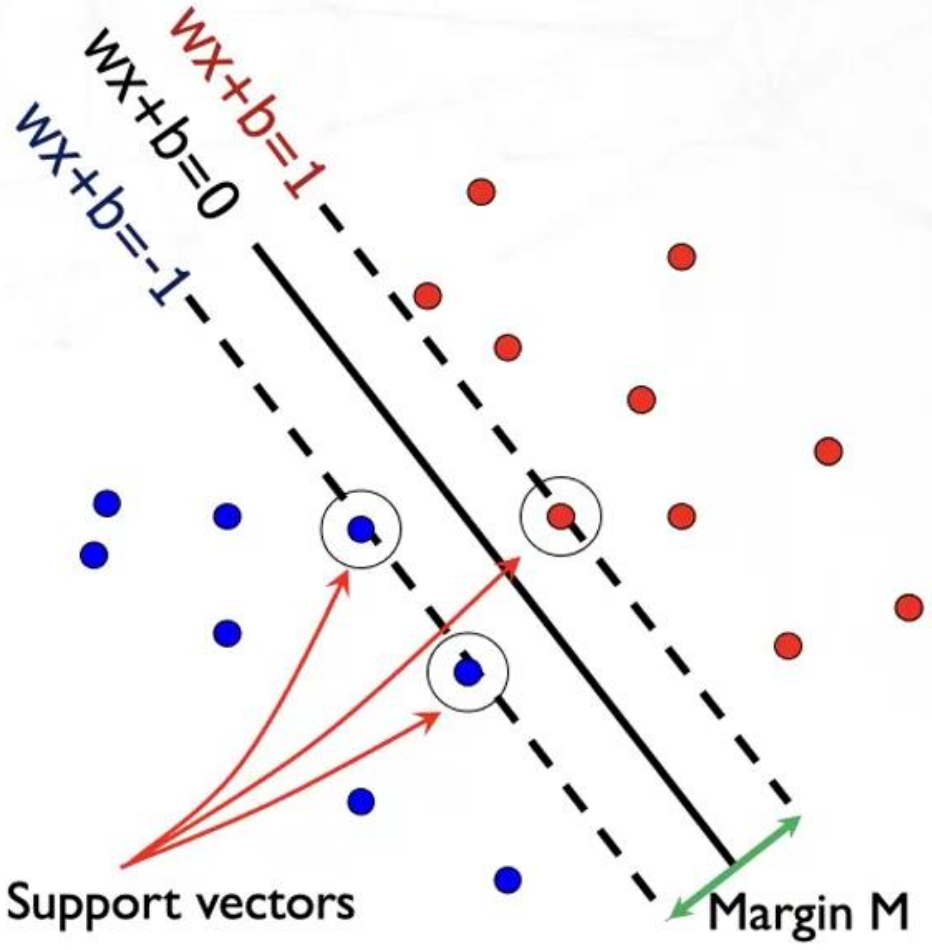


$$ax + by + cz + d = 0$$

$$\mathbf{w} \cdot \mathbf{x} + d = 0$$

$$D = \frac{|ax_0 + by_0 + cz_0 + d|}{\sqrt{a^2 + b^2 + c^2}} = \frac{\mathbf{w}^{\mathrm{T}}\mathbf{x} + d}{\|\mathbf{w}\|}$$ distance from point to plane

Support Vector Machines



$\mathbf{x}_i$ positive $(y_i = 1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$

$\mathbf{x}_i$ negative $(y_i = -1)$: $\quad \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$

$$\mathbf{x}_i \cdot \mathbf{w} + b = \pm 1$$

For support, vectors,

Distance between point and line: $\dfrac{|\mathbf{x}_i \cdot \mathbf{w} + b|}{\|\mathbf{w}\|}$

For support vectors:

$$\frac{\mathbf{w}^T\mathbf{x} + b}{\|\mathbf{w}\|} = \frac{\pm 1}{\|\mathbf{w}\|} \qquad M = \left| \frac{1}{\|\mathbf{w}\|} - \frac{-1}{\|\mathbf{w}\|} \right| = \frac{2}{\|\mathbf{w}\|}$$

Finding the Maximum Margin Plane

$$\mathbf{x}_i \text{ positive } (y_i = 1): \quad \mathbf{x}_i \cdot \mathbf{w} + b \geq 1$$

$$\mathbf{x}_i \text{ negative } (y_i = -1): \mathbf{x}_i \cdot \mathbf{w} + b \leq -1$$

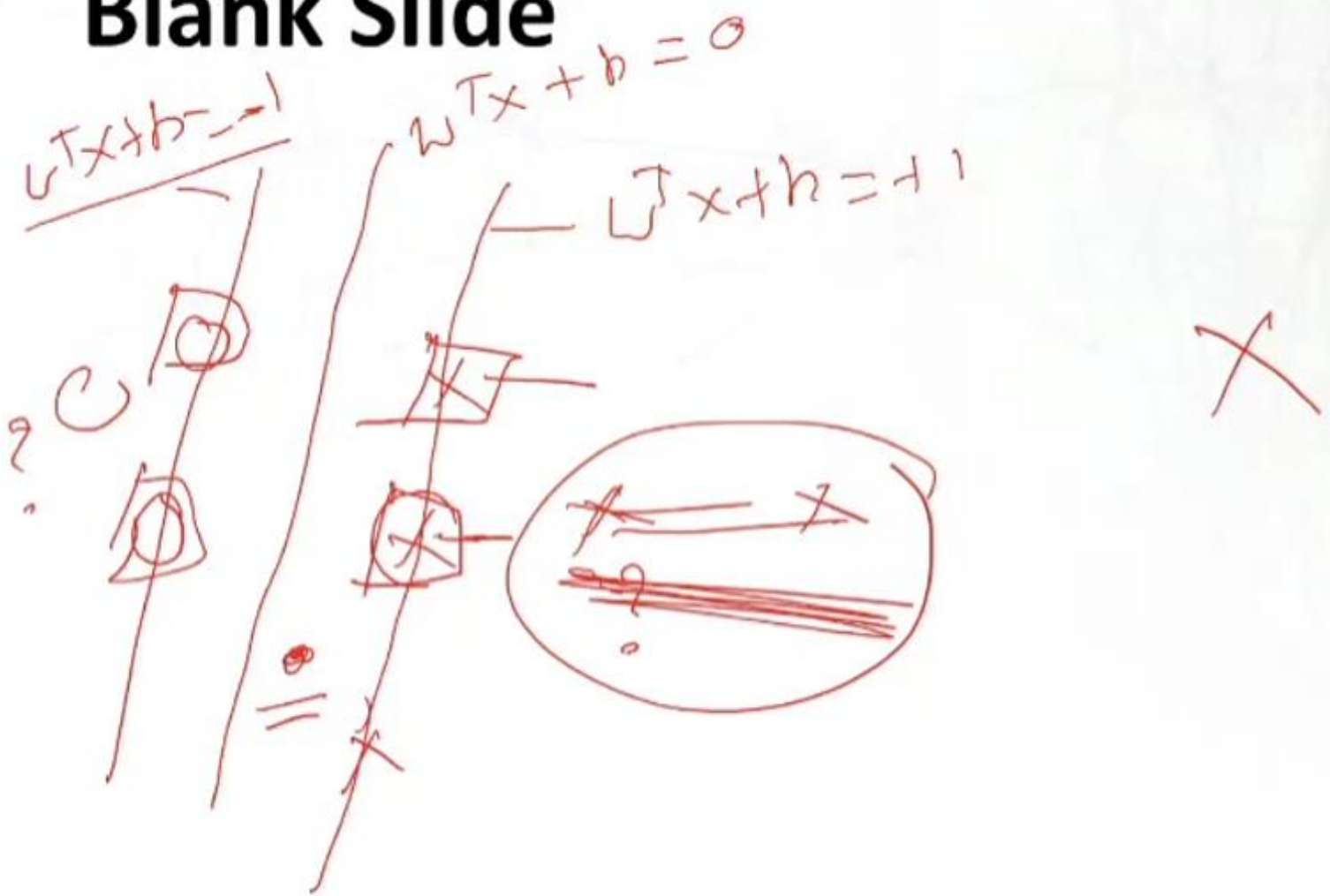
$$\text{Min } \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{M} \xi_i$$

$$\text{subject to: } y_i(w^T x_i) \geq \mathbf{1} - \xi_i, \xi_i \geq \mathbf{0}$$

One constraint for each training point.

Note sign trick.

Blank Slide

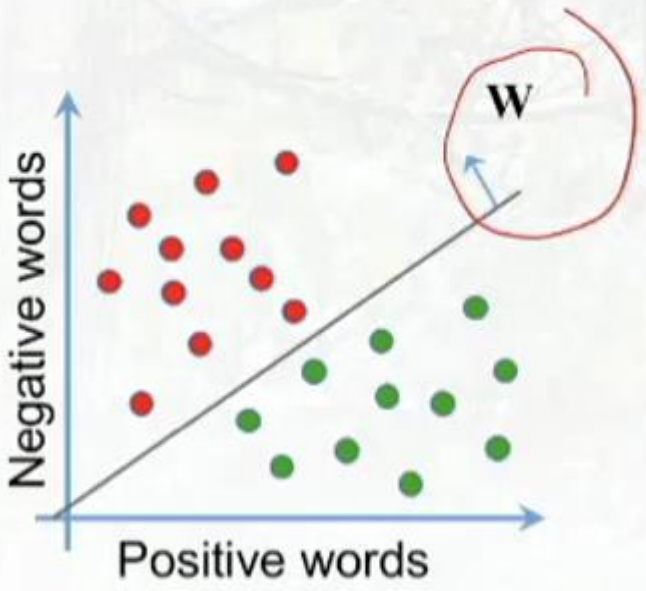


Blank Slide

Creating a SPAM Filter

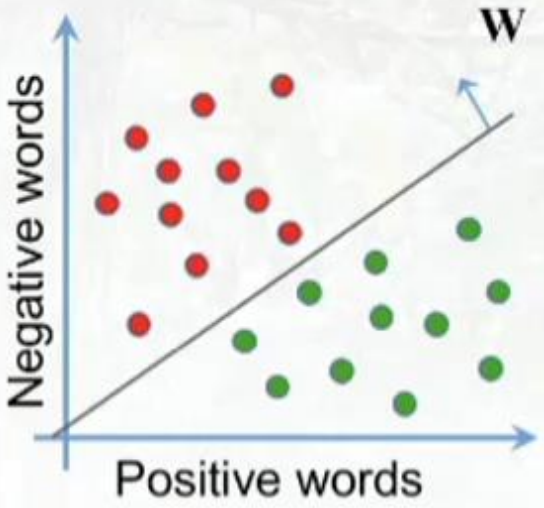
- Can we separate SPAM from Non-SPAM ?

1. Collect a set of example emails (training data)
2. Convert the emails to vector representation
3. Label the examples as SPAM or not
4. Learn a separator in the feature space

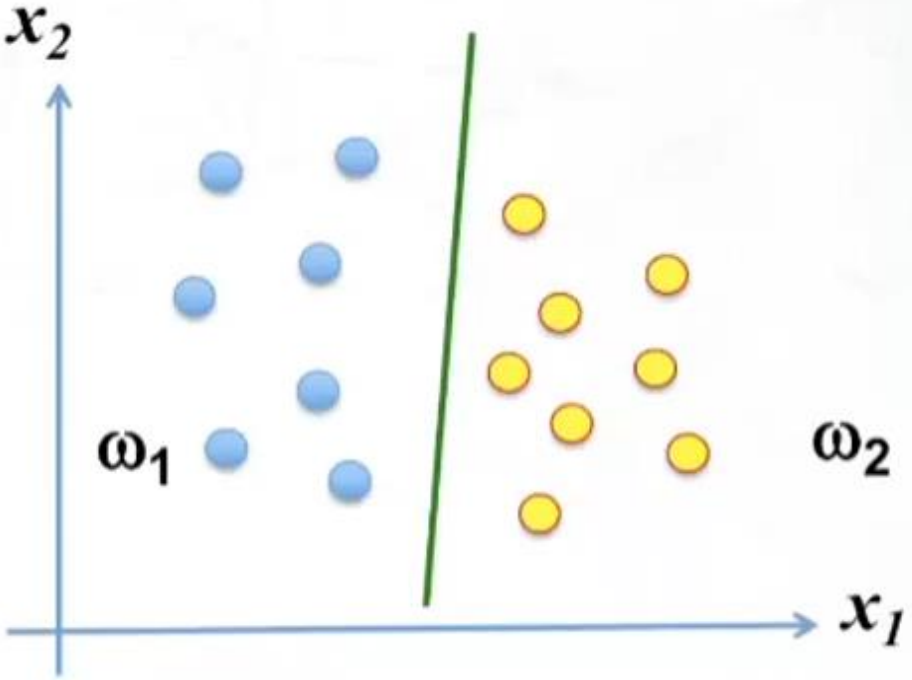


Training vs Testing

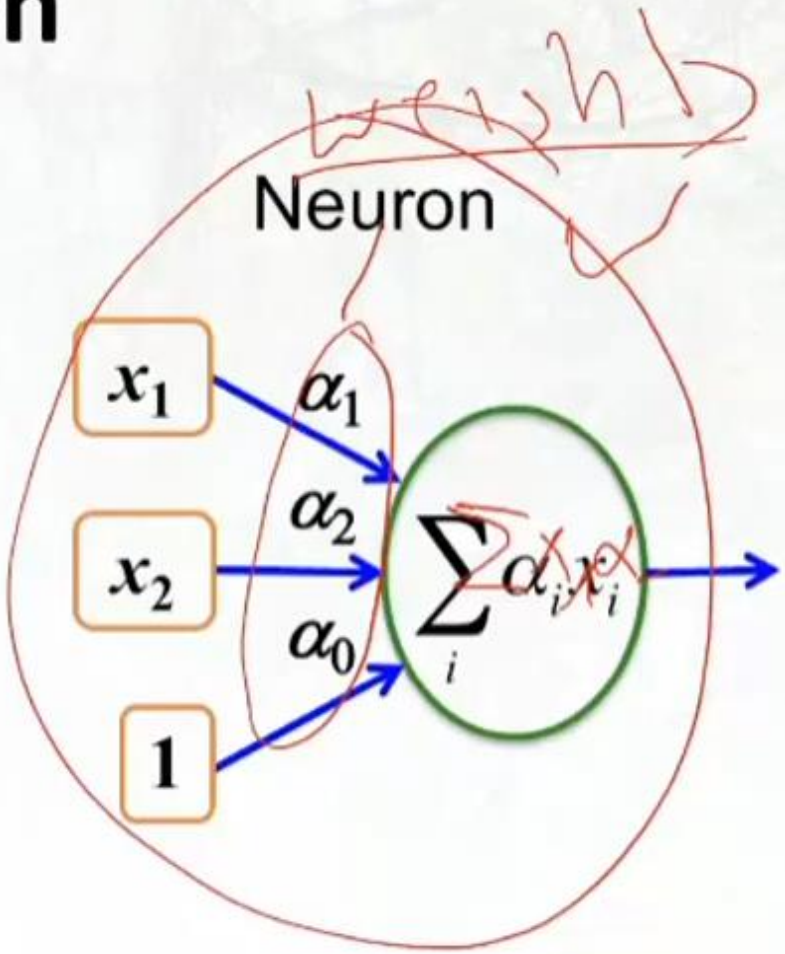
- Training
  - Using examples to 'learn' a classifier model
  - $y = f(\mathbf{W}, \mathbf{X})$: We learn the parameters $\mathbf{W}$ during training
- Testing
  - Predict the y values for new samples
  - Why **new** samples?
    - Using training samples to test may give a false sense of capability



Linear Classifier: A Neuron



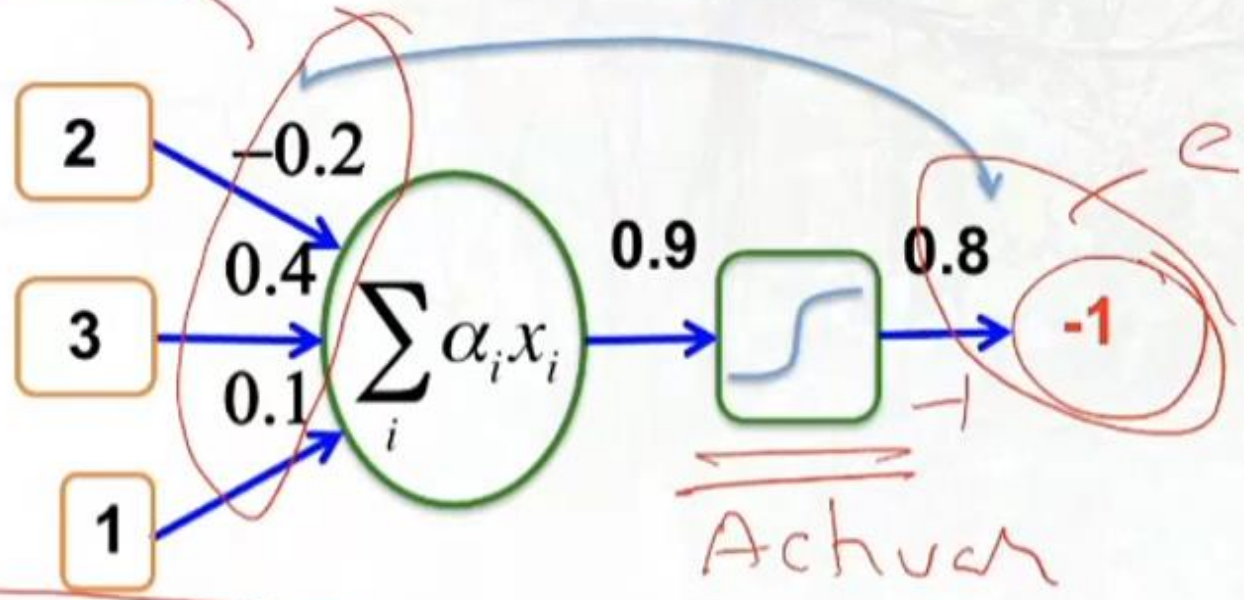
$$\alpha_1 x_1 + \alpha_2 x_2 + \alpha_0 = 0$$



- A linear boundary separates two classes.
- Learning the classifier means learning the weights, α_i.

Blank Slide

Learning



weight update

error

Actual

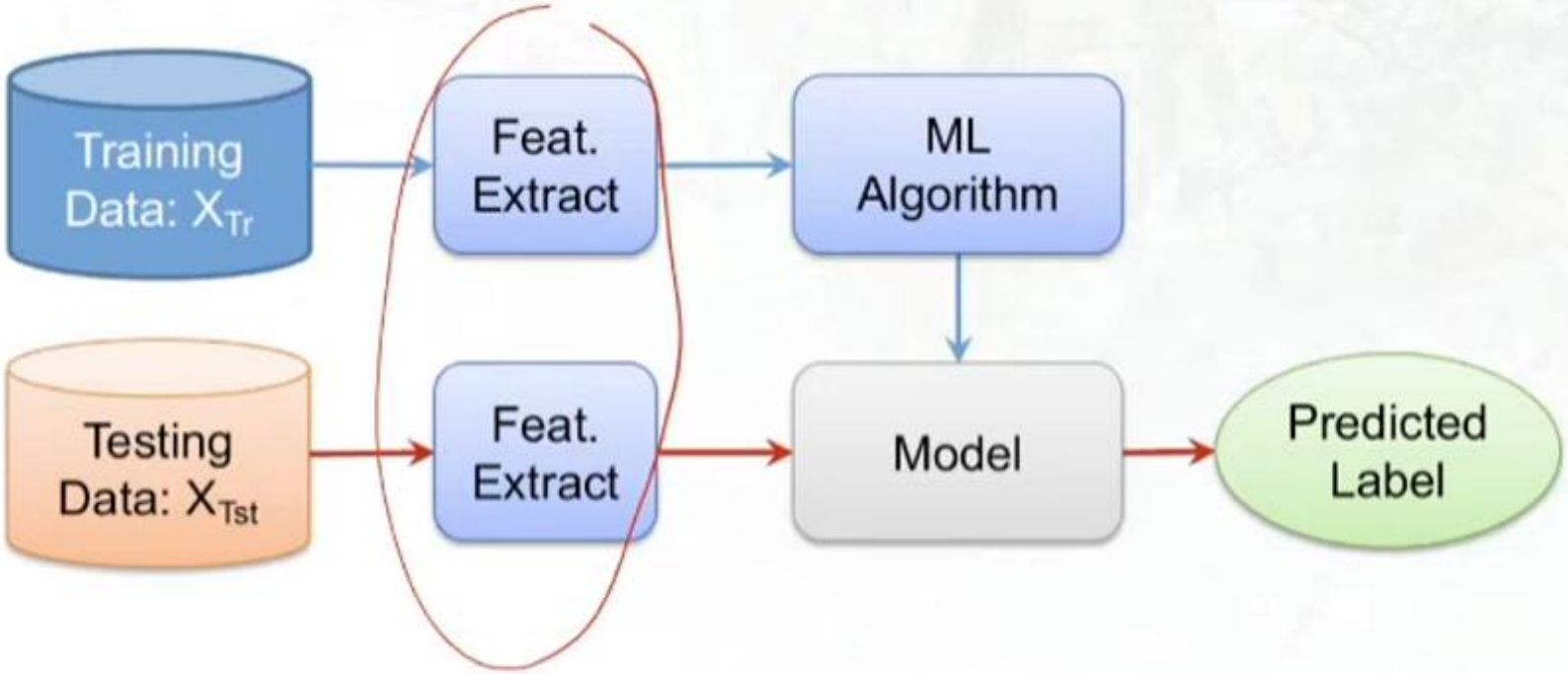
error is small

Repeat until weights stabilize

1. Take a training sample and pass it through the neuron
2. Find the difference between the output and expected values (loss function)
3. Update the weights to reduce this loss

Blank Slide

Summary: The ML Pipeline



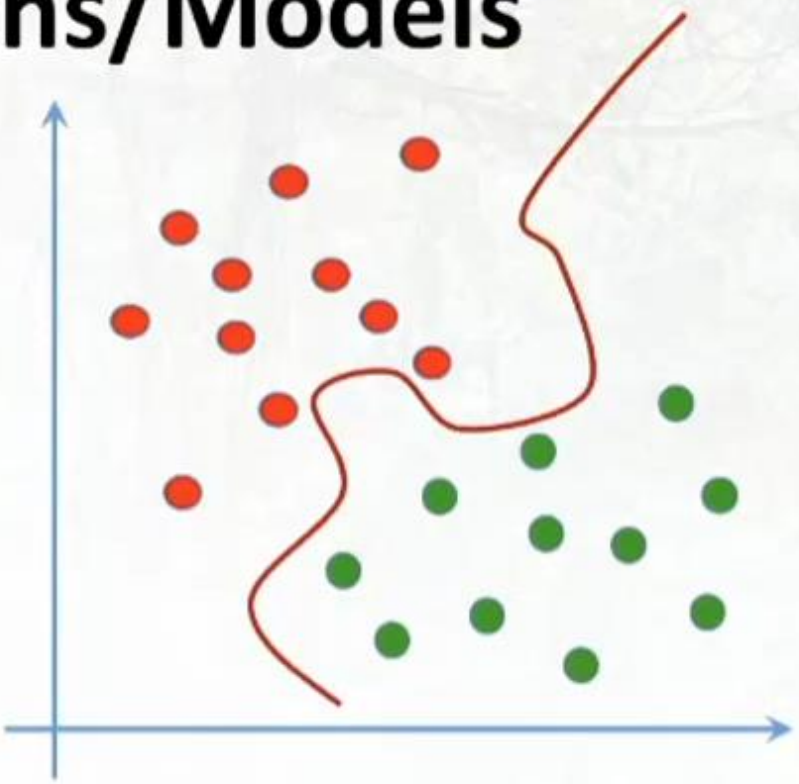
In traditional AI models,
the features used to be engineered and learning was limited to the classifier

C. Worries During Learning

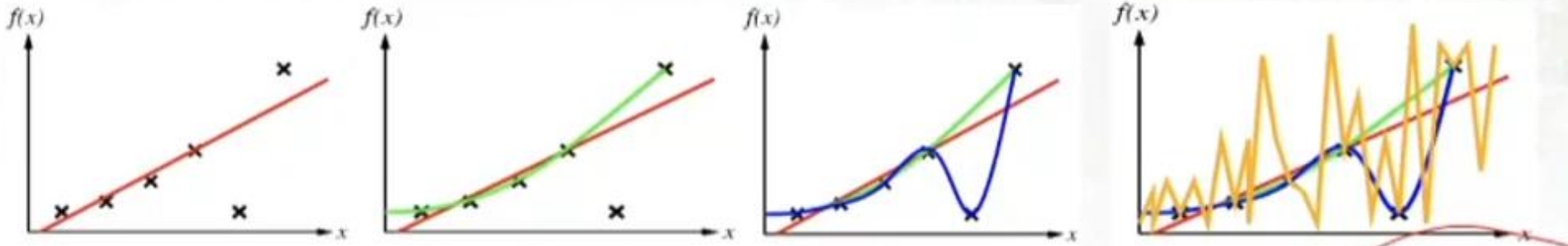
Wide Ranges of Functions/Models

Models for Learning

- If-Else: Axis-Parallel (Decision Trees)
- Linear Classifier
- Distance based (Nearest Neighbor)
- Non-linear (Neural Network)



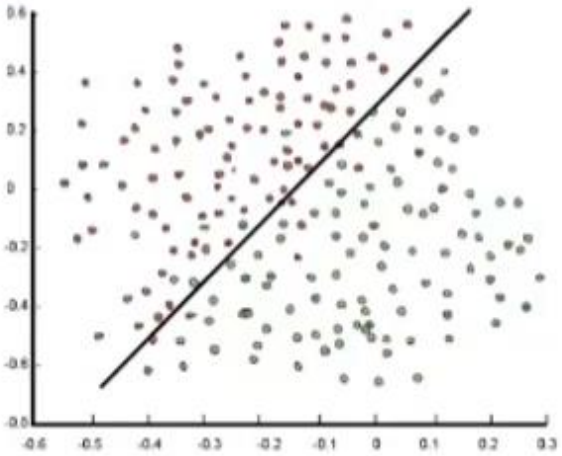
Remember: Generalization and Overfitting

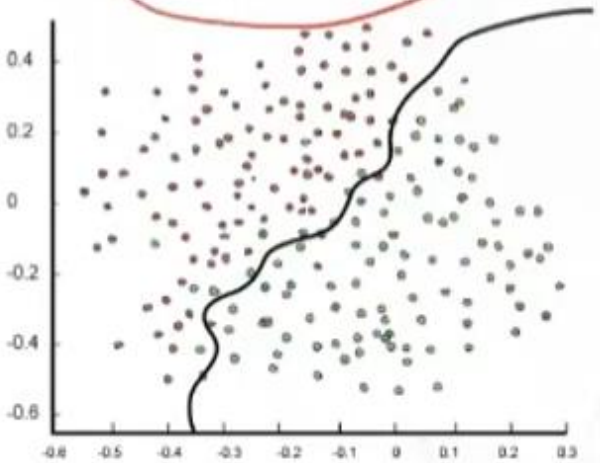


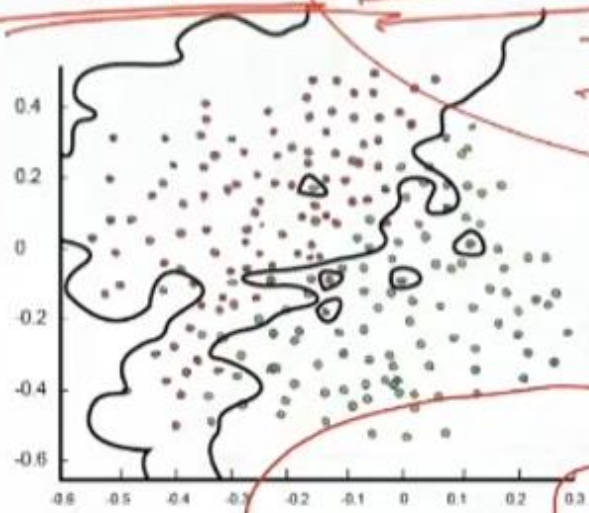
Comp ↓ Tr ↑

?

Comp ↑ Tr err = 0







X

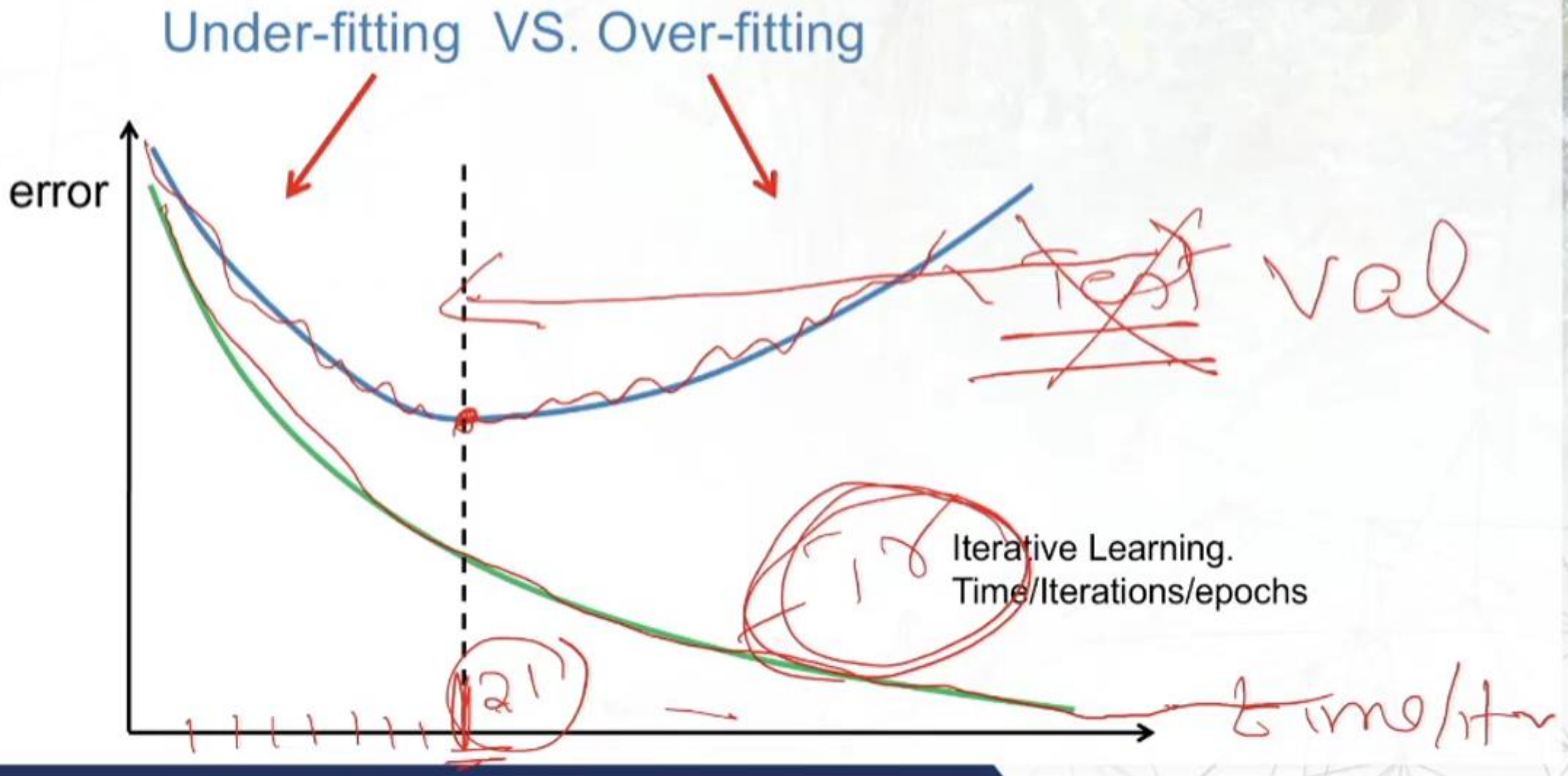
overfitting

Overfitting

- Performance on the training data is too good
- Performance on the test data is bad.

- Why?
  - Algorithm has remembered the data/solutions.

What are we seeking?

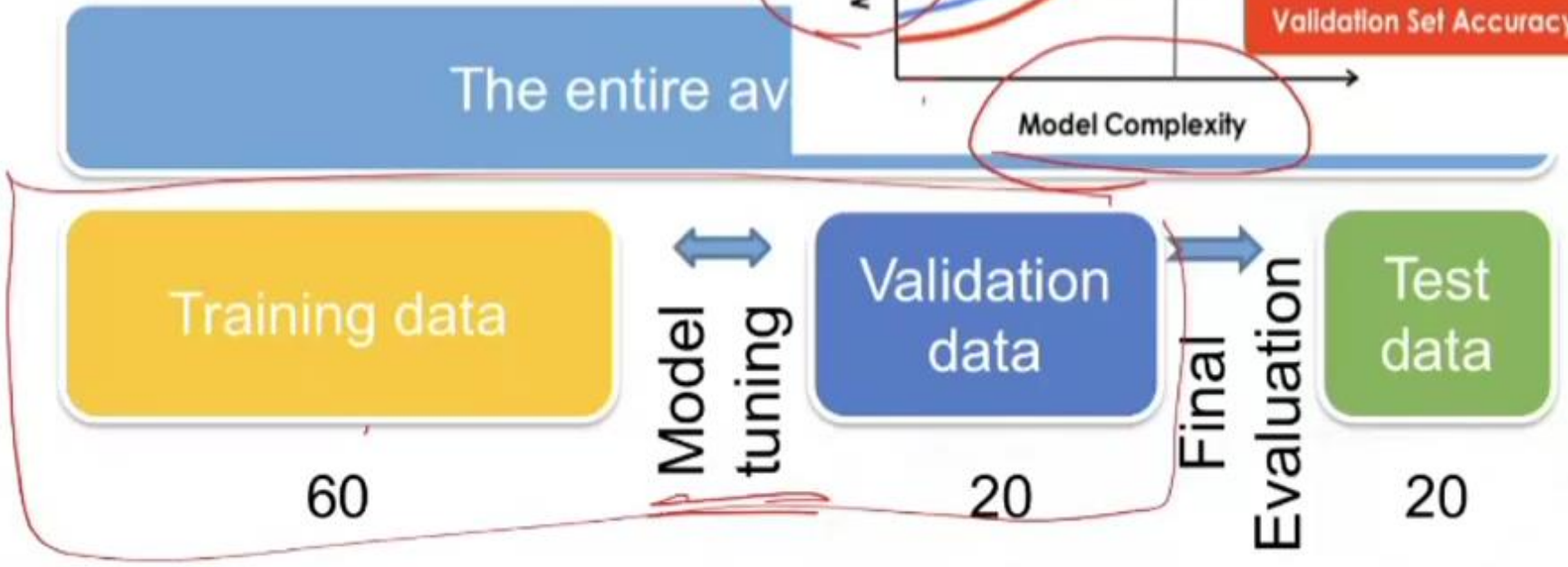


Blank Slide

Blank Slide

Detecting Overfitting

- When do we overfit?
  - Model is powerful/complex
  - Limited training samples
  - Large number of features
- Empirical Detection

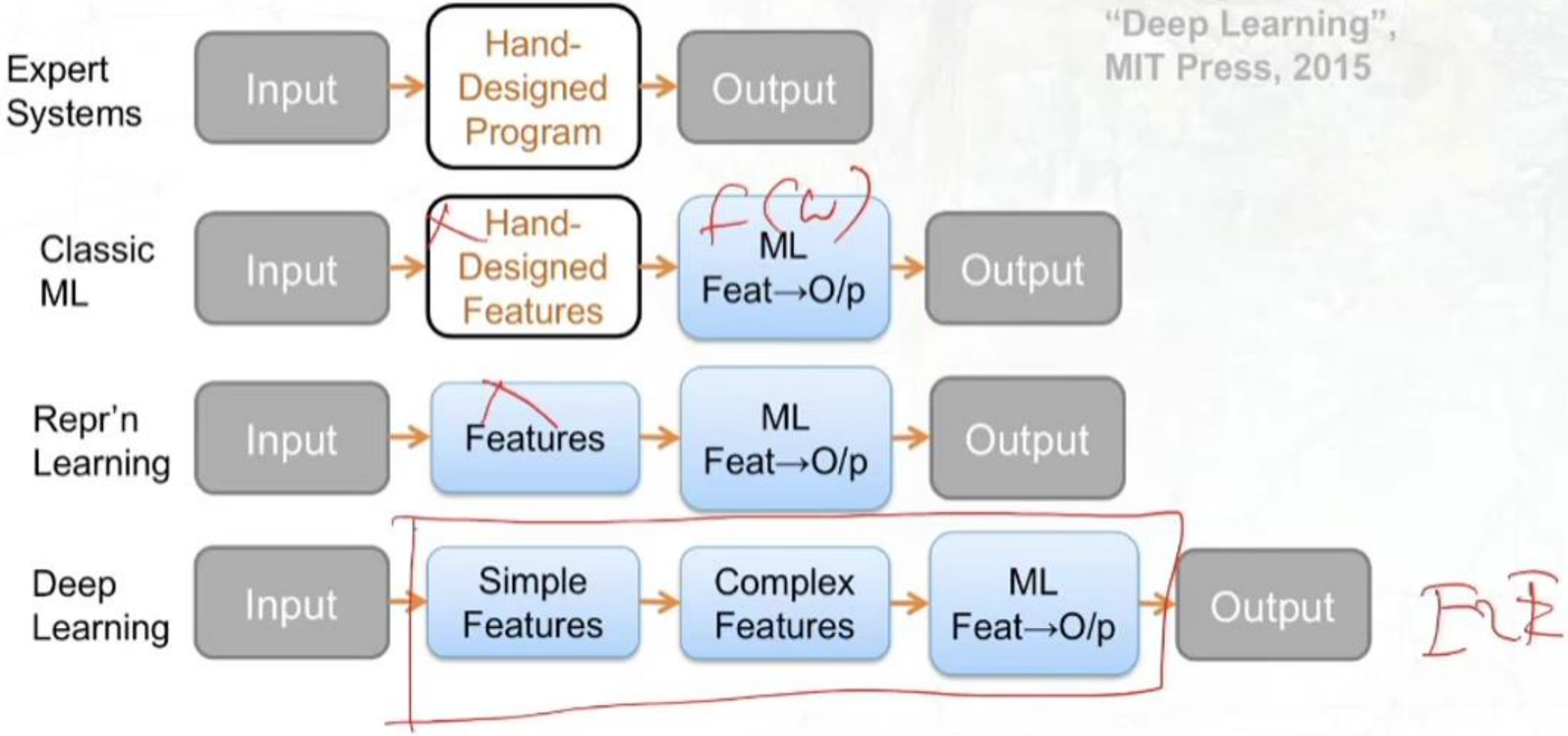


Overfitting: Solutions

- Reduce the model complexity
  - Regularization
  - Ensemble Techniques
- Increase the number of training samples
  - Data Augmentation
- Reduce the number of features
  - Dimensionality Reduction
- Model-specific methods
  - Early Stopping
  - Pruning
  - Large-margin methods

Evolution of Learning

Y. Bengio et al,
"Deep Learning",
MIT Press, 2015

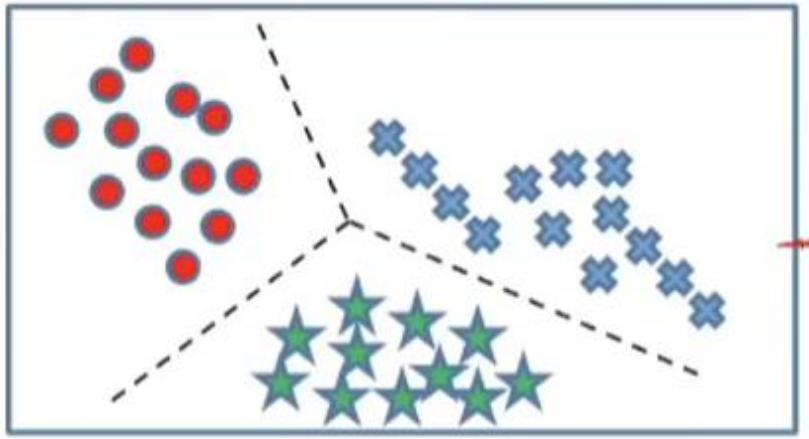


Summary

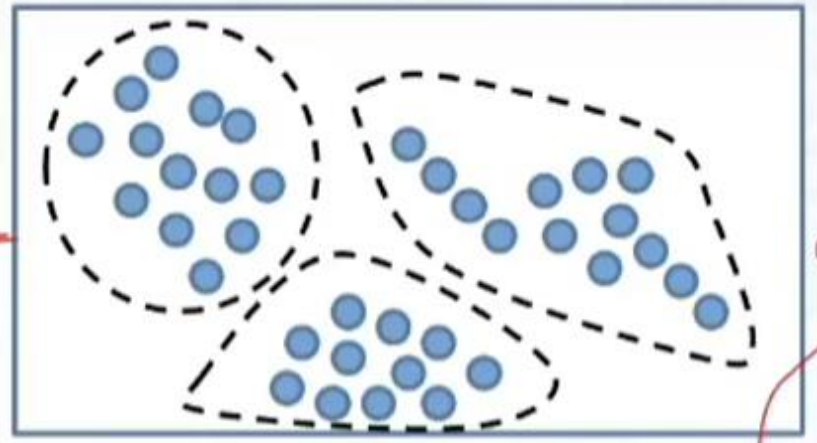
- Data driven problem solving
  - Modelling vs Learning/Generalization
  - Classification, Regression, Structured Prediction
  - Many other formulations/problems
- Function Class
  - Simple Models: DT, Nearest Neighbour, Linear
  - Powerful, Popular Neural Networks
- Problem of Learning
  - Optimization: Convex and non-convex
  - Learning as "finding the W"
  - Deep Architectures (more next lecture)

D. Learning Paradigms and Optimization

Class of ML Algorithms

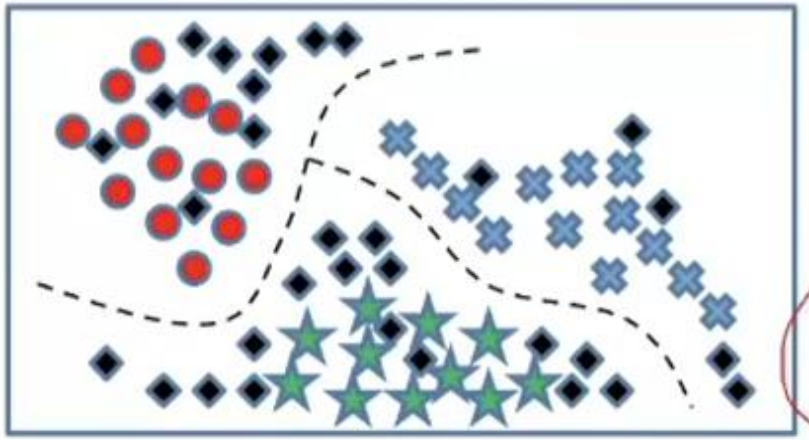


Supervised learning



cluster

Unsupervised learning



W. Sup
Self sup.

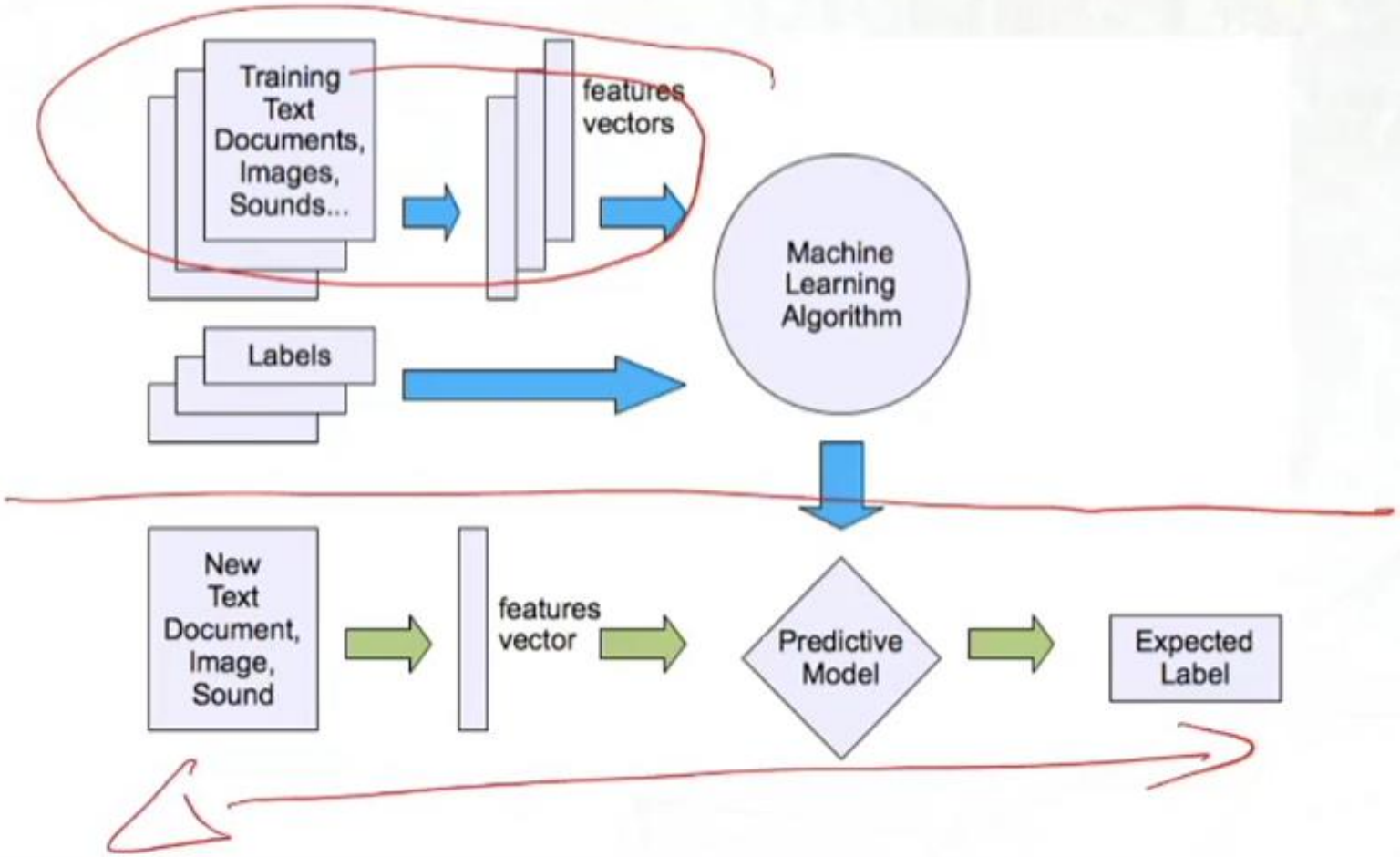
Semi-supervised learning

Algorithms

- **Supervised learning** ($\{x_n \in R^d, y_n \in R\}_{n=1}^N$)
  - Prediction
  - Classification (discrete labels), Regression (real values)
- **Unsupervised learning** ($\{x_n \in R^d\}_{n=1}^N$)
  - Clustering
  - Probability distribution estimation
  - Finding association (in features)
  - Dimension reduction
- **Semi-supervised learning**
- **Reinforcement learning**
  - Decision making (robot, chess machine)

Machine learning structure

- **Supervised learning**



Optimization: Find the *best* W

$$\text{Min } \frac{1}{2} \| w \|^2 + C \sum_{i=1}^{M} \xi_i$$

$$\text{subject to: } y_i(w^T x_i) \geq \mathbf{1} - \xi_i, \xi_i \geq \mathbf{0}$$

Eg. SVM

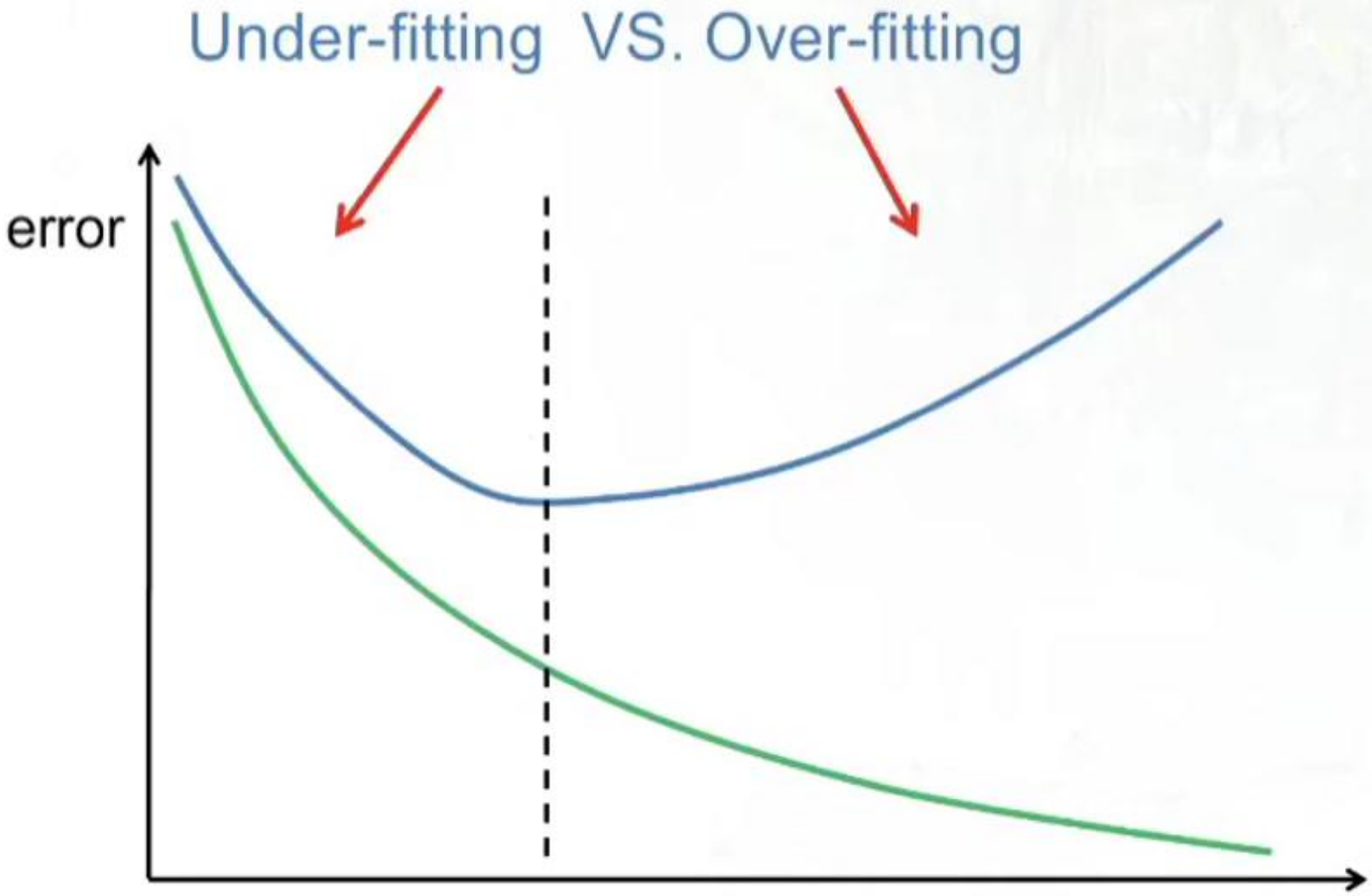
Often problem is formulated as an optimization problem on the set of known examples

Some More Key Words

- Training: Find f and W
- Testing: Evaluate f on a specific example
- Training, Testing and Validation splits of the data
- Generalization: Goal is to do well on "unseen data"
- Error, Loss, Objective Functions
- Complexity of the solution (eg. Number of free parameters)
- Generative classifiers try to model the data. Discriminative classifiers try to predict the label.

V.C. Dimension

What are we seeking?



Example: Linear Classification

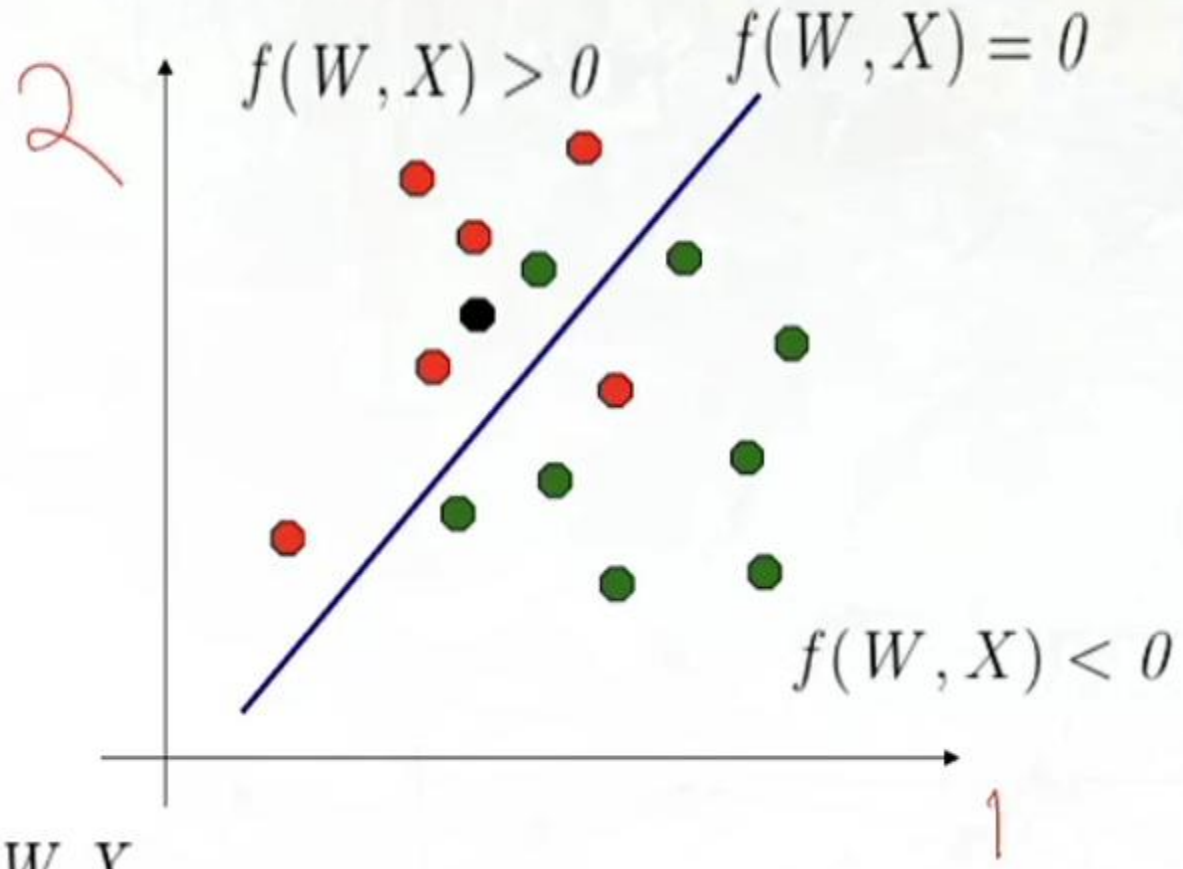
$$f(W, X) = w_1 x_1 + w_2 x_2 + \ldots + w_d x_d$$

=1

d - dimensi.

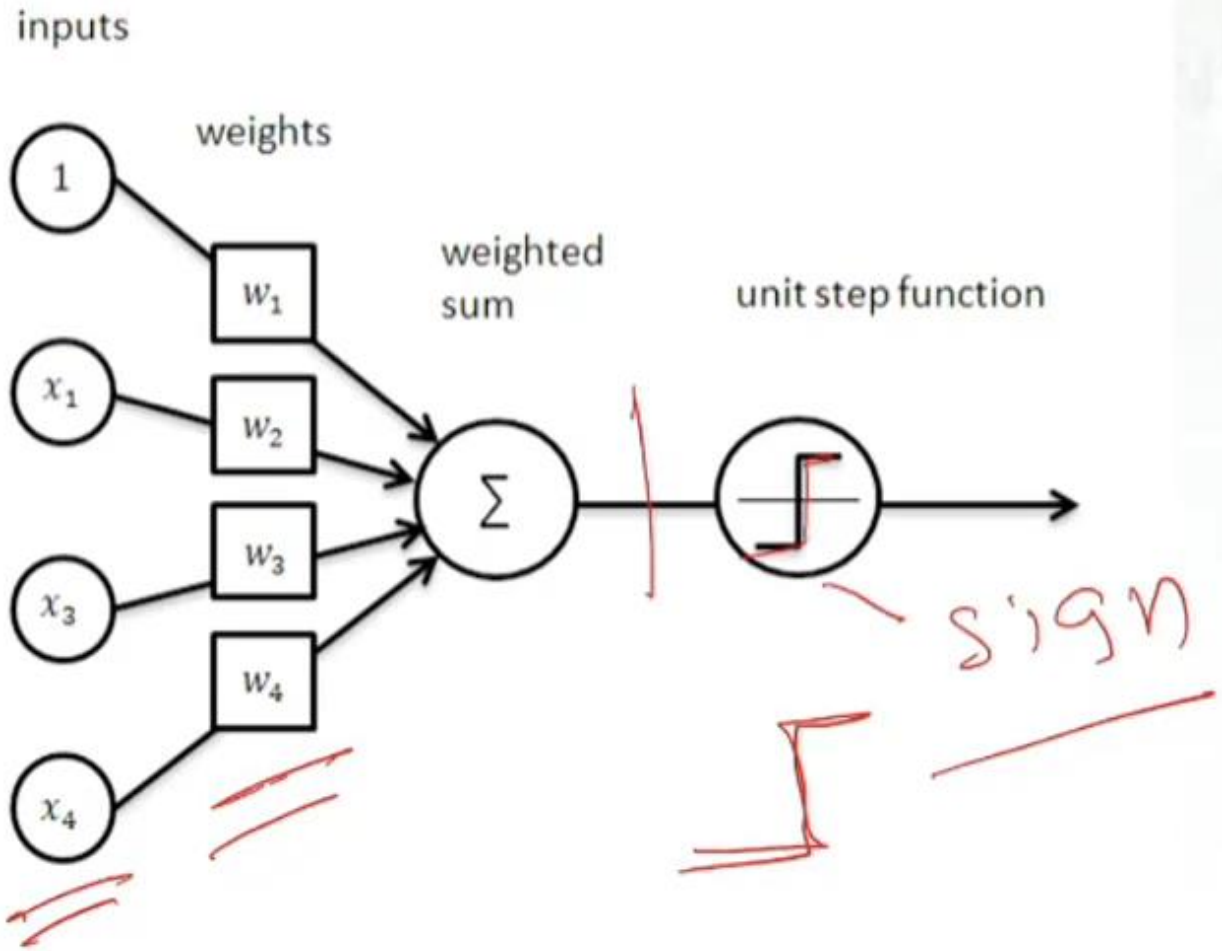
Feature Vector $X = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$

Parameters to be learned $W = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}$



Compactly: $f(W, X) = W^T X$ or $W.X$

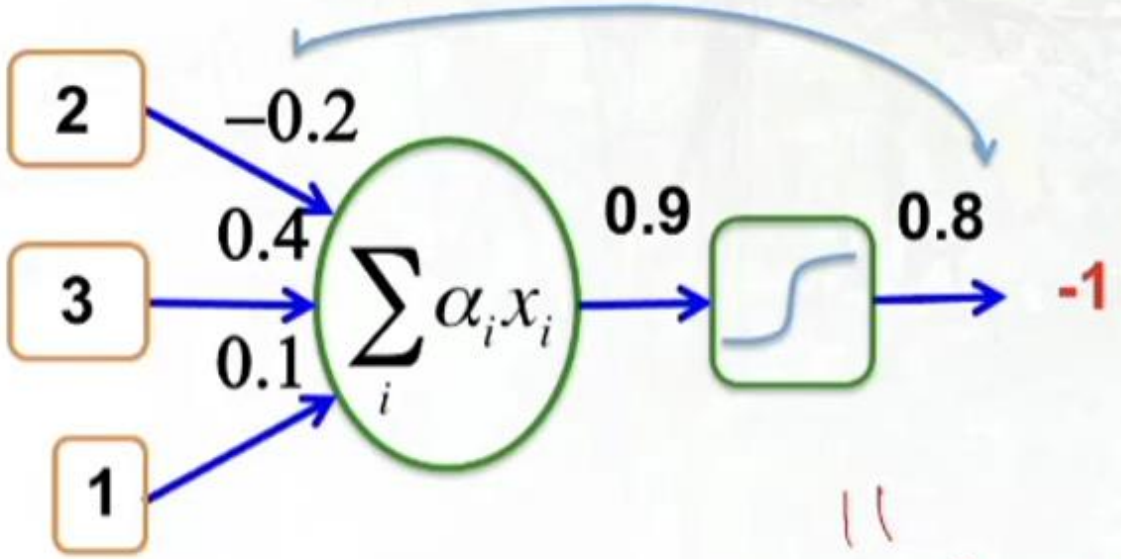
Example: A "Neural Network" Perspective



Predicted Label:

Sign ($\mathbf{w}^T\mathbf{x}$)

Basics: Learning



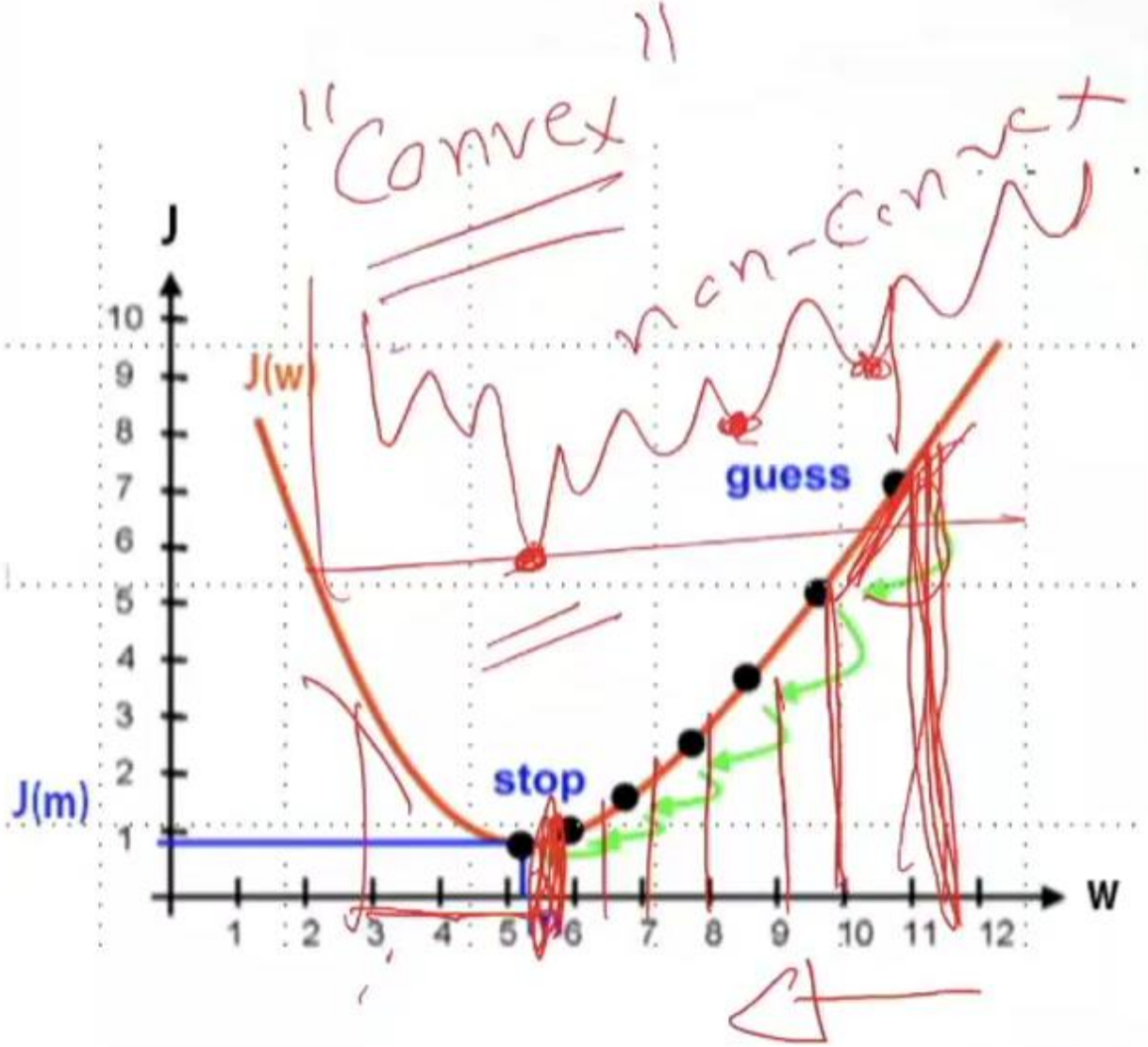
weight update

"Gradient Desc"

Repeat until weights stabilize

1. Take a training sample and pass it through the neuron
2. Find the difference between the output and expected values (loss function)
3. Update the weights to reduce this loss

Basics: Gradient Descent: Intuition



Problem: Minimize J(w).

1. Start with an initial guess.
2. Update your guess by moving along the negative slope (gradient).

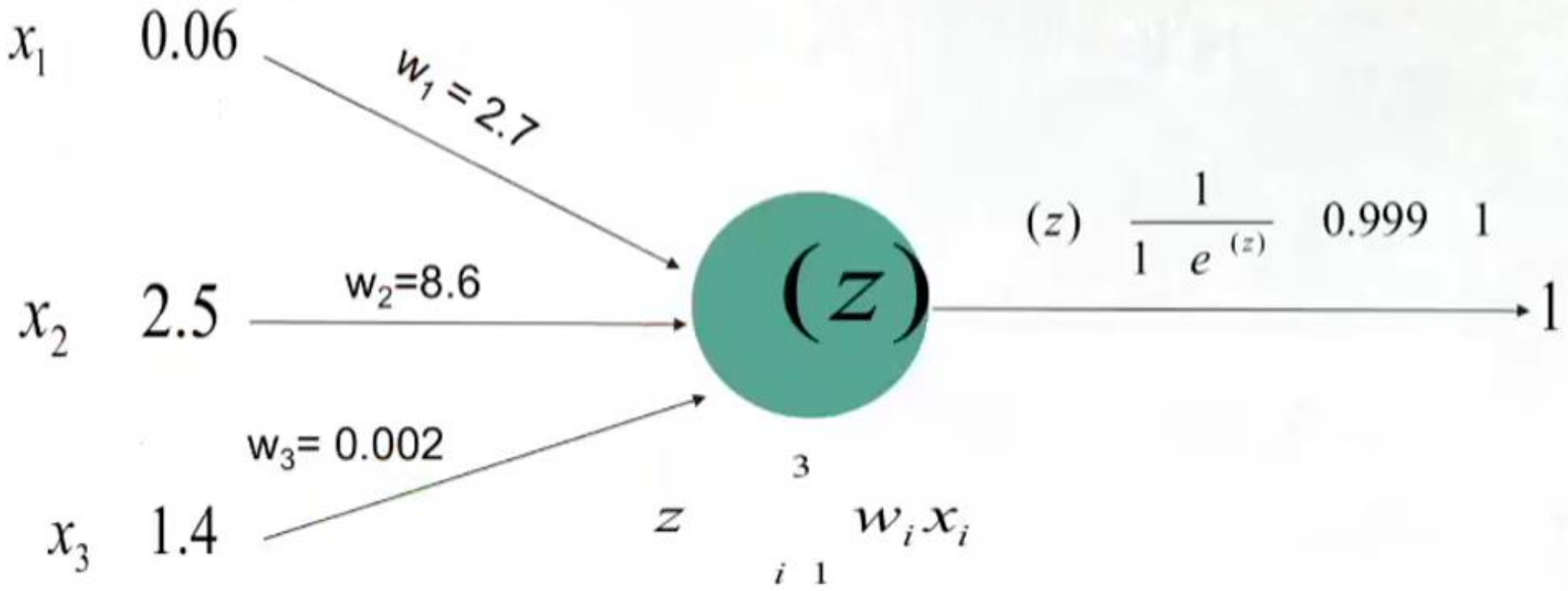
$$\mathbf{w_{n+1}} = \mathbf{w_n} + \eta \Delta \mathbf{w}$$

Notes:

1. We do not have an analytic form of J(w) to differentiate and find the optima/best/minima.
2. We can differentiate "locally" at the samples and update the weights/parameters.

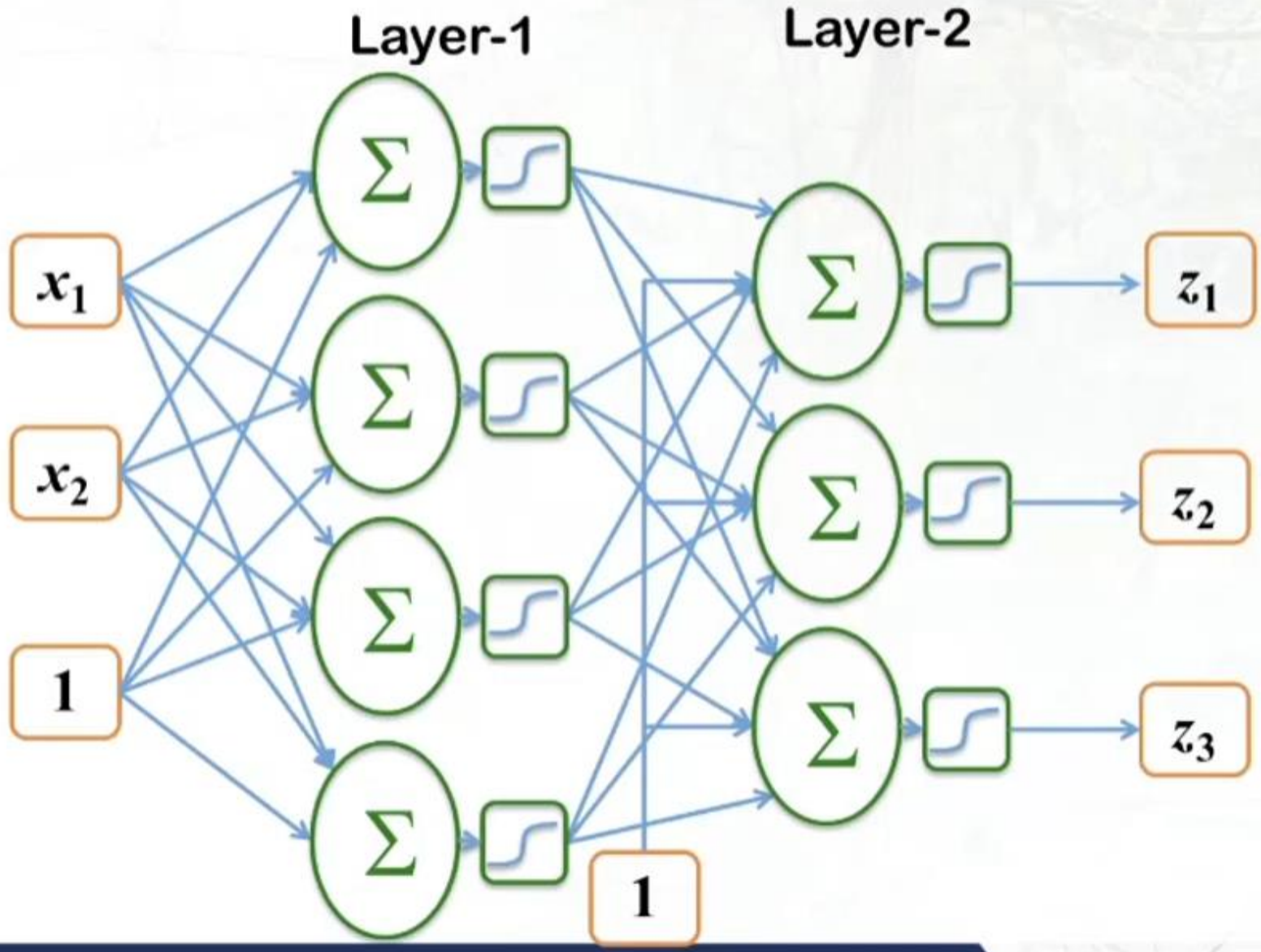
Learning in Neural Networks

Learning = Finding the best or optimal weights

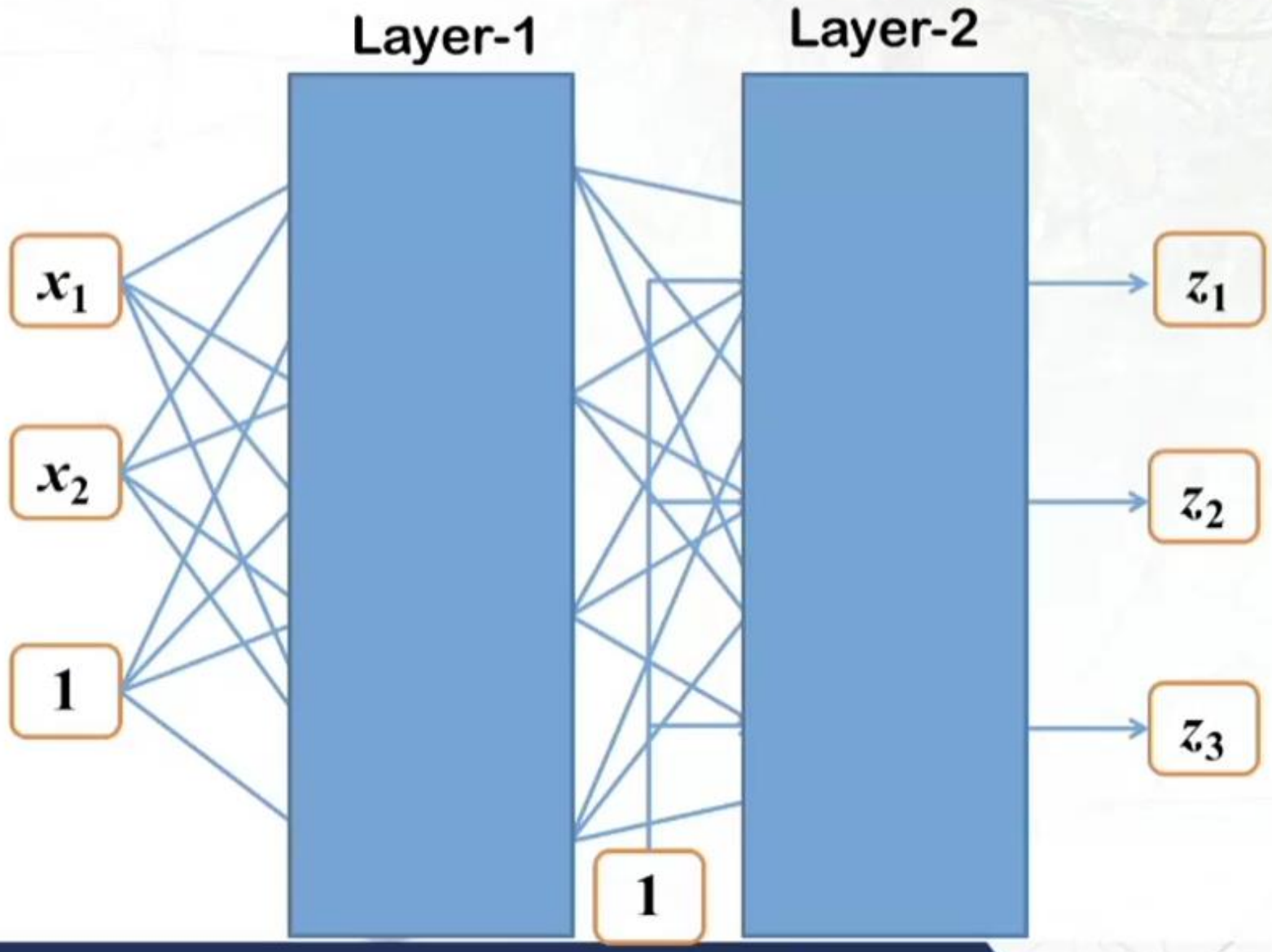


z -0.06 2.7 2.5 8.6 1.4 0.002 21.34

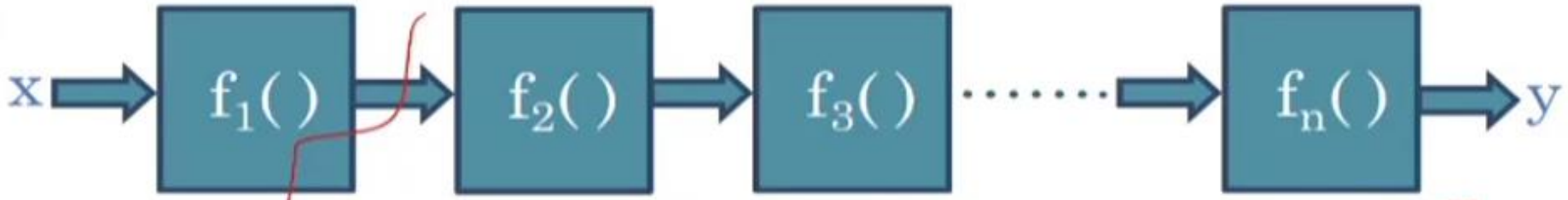
Neural Network

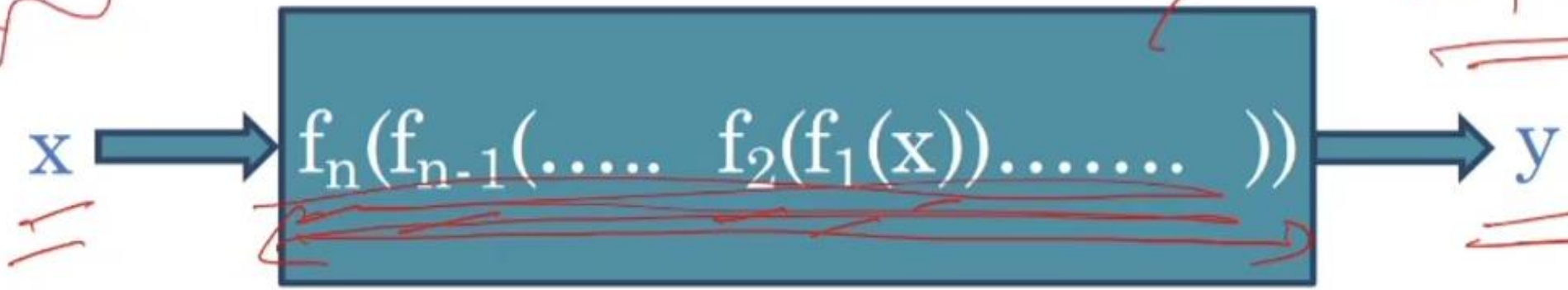


Neural Network

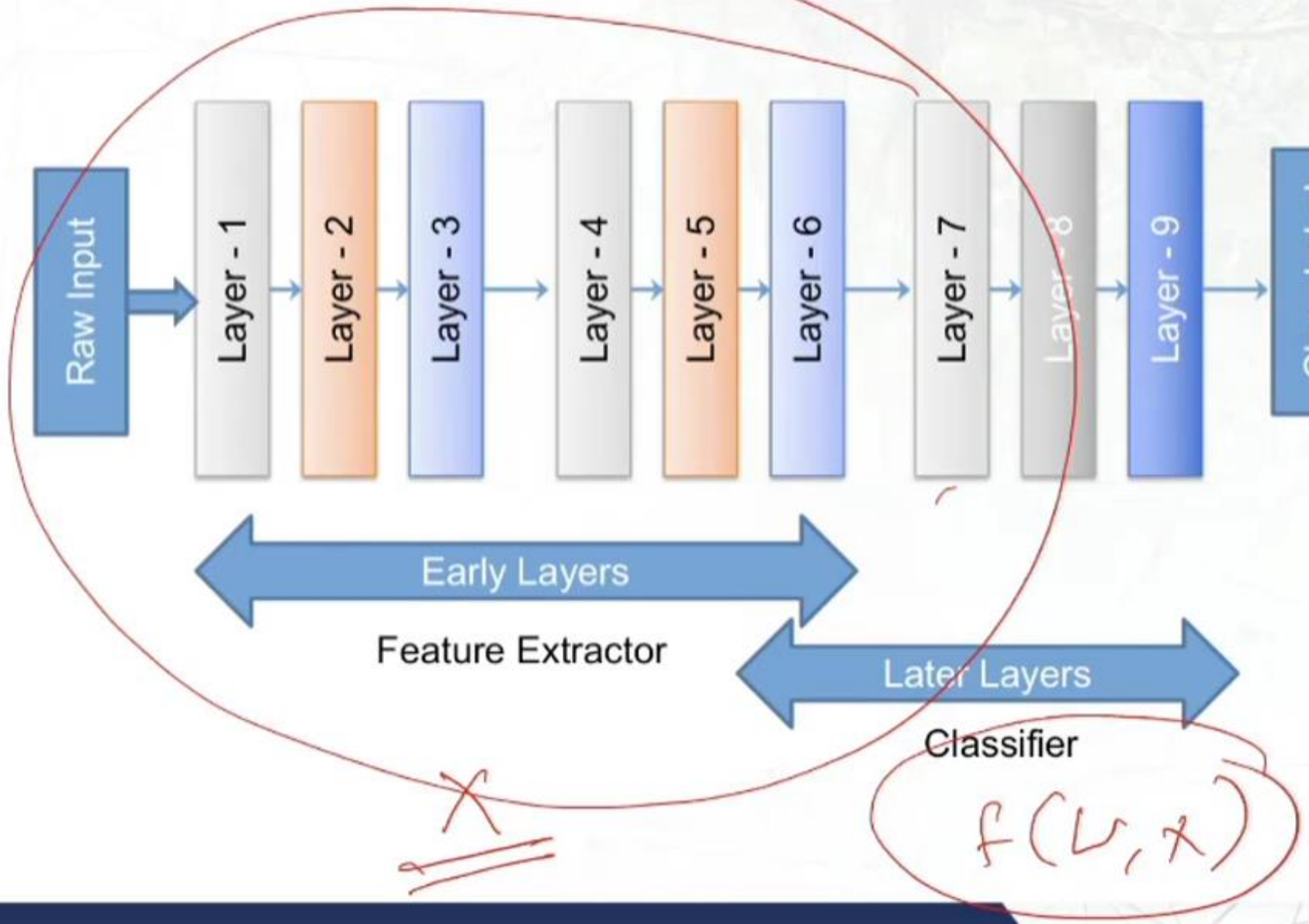


Complex Functions and Richer Features

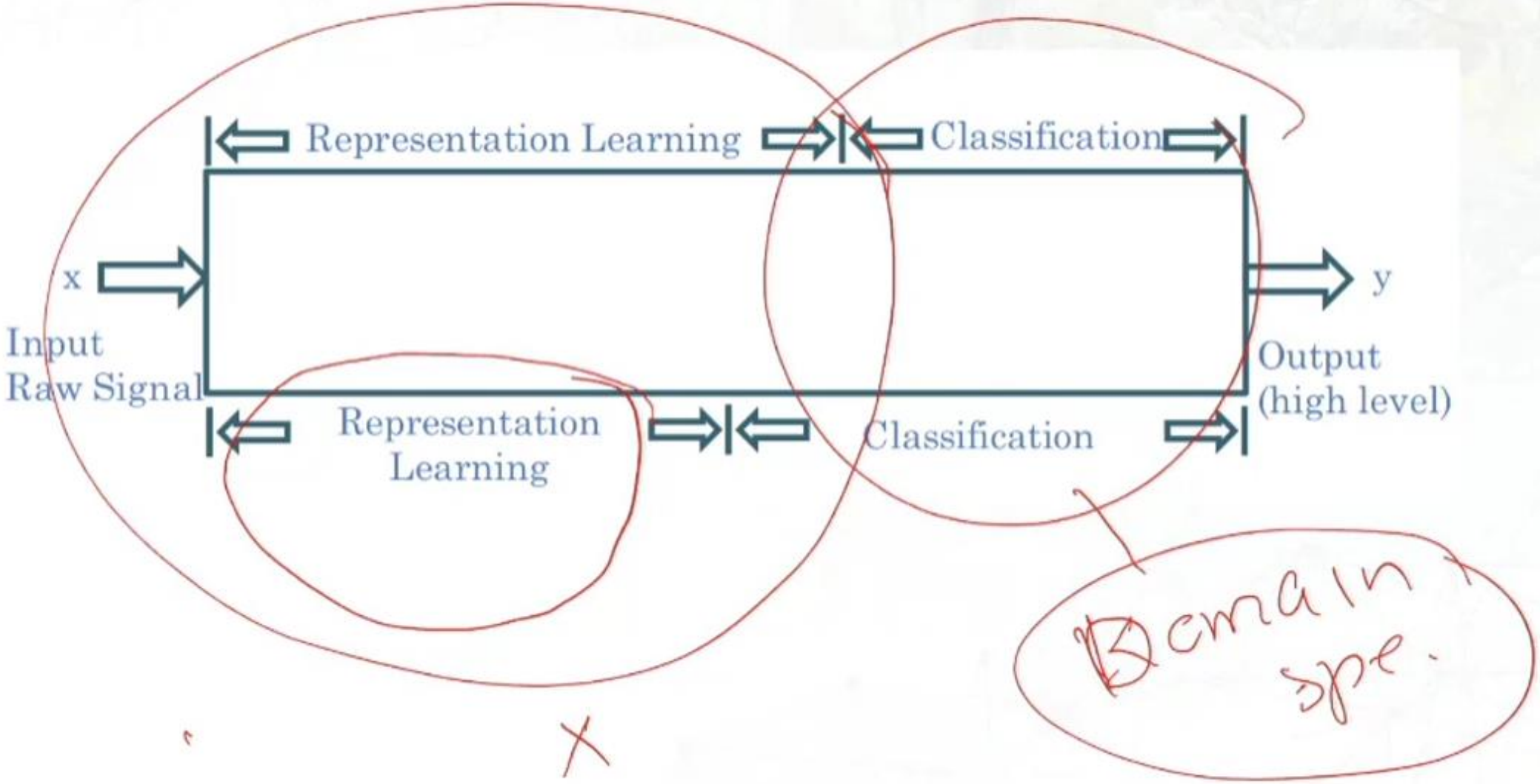




Deep Neural Networks



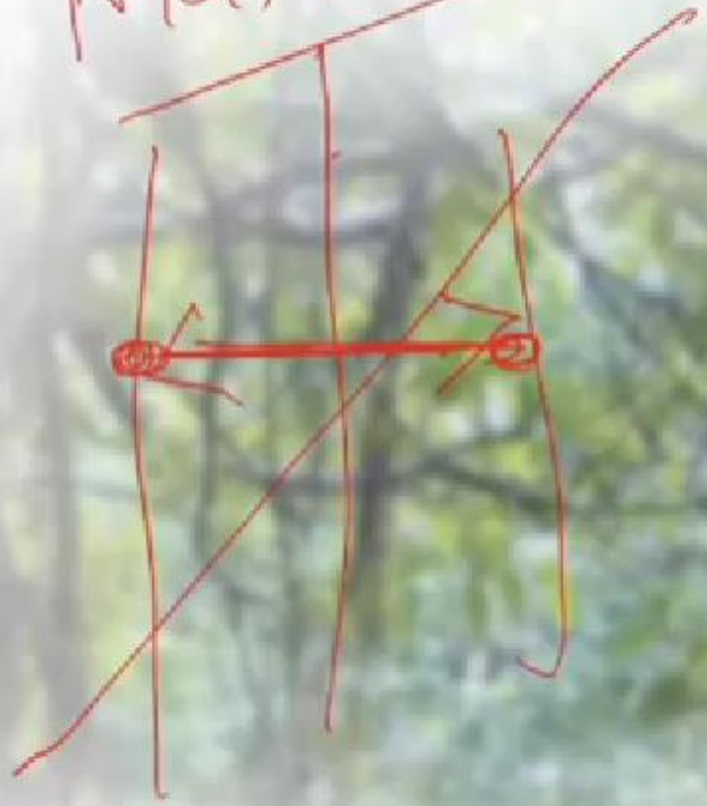
View Point


Representation Learning
Classification
x
Input
Raw Signal
y
Output
(high level)
Representation
Learning
Classification
Domain
spe.

Summary

- **Popular methods of today are**
  - Supervised
  - Discriminative
- **SVMs were/are popular.** Nice optimization problem to solve
- **Deep neural networks are becoming the standard for many problems**
  - Feature extraction
  - End to end training
  - Trained models for evaluation
  - Porting/Transforming network to network.

max-margin



$\min_{w} L + \|w\|$ # norm

Thanks!!

L-fn

Questions/Comments?

Regularize

S SS P