

# CSCI 5622 Machine Learning

## ML Support Vector Machines

DATE	READ	DUE
Today, Sept 28	Burgess	Peer Fdbk Grades
Wed, Sept 30	Cristianini	Notes Papers 1&2
Mon, Oct 5	7	Exper. 1 plan (1 pg)

[www.RodneyNielsen.com/teaching/CSCI5622-F09/](http://www.RodneyNielsen.com/teaching/CSCI5622-F09/)

**Instructor: Rodney Nielsen**

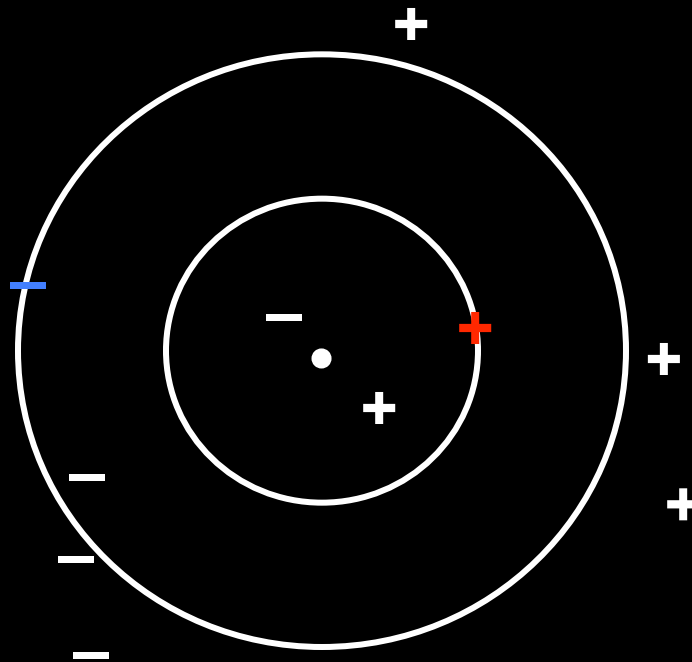
**Assistant Professor Adjunct, CU Dept. of Computer Science**

**Research Assistant Professor, DU, Dept. of Electrical & Computer Engr.**

**Research Scientist, Boulder Language Technologies**

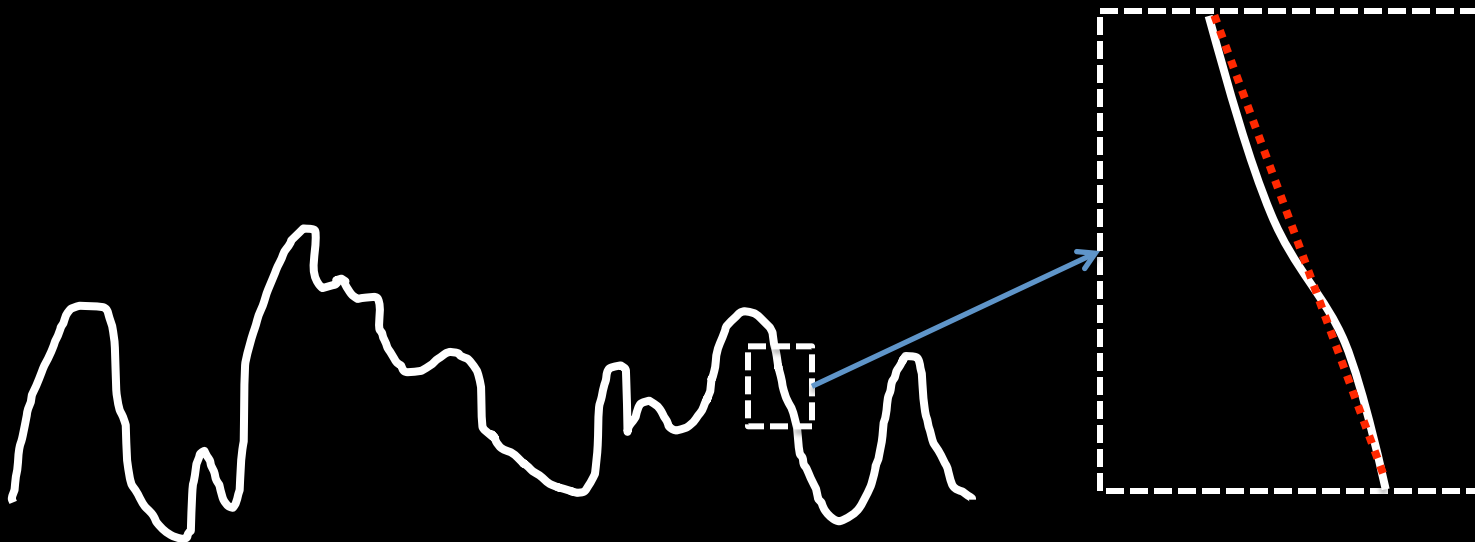
# ML $k$ -Nearest Neighbor

- $k$ -NN algorithm

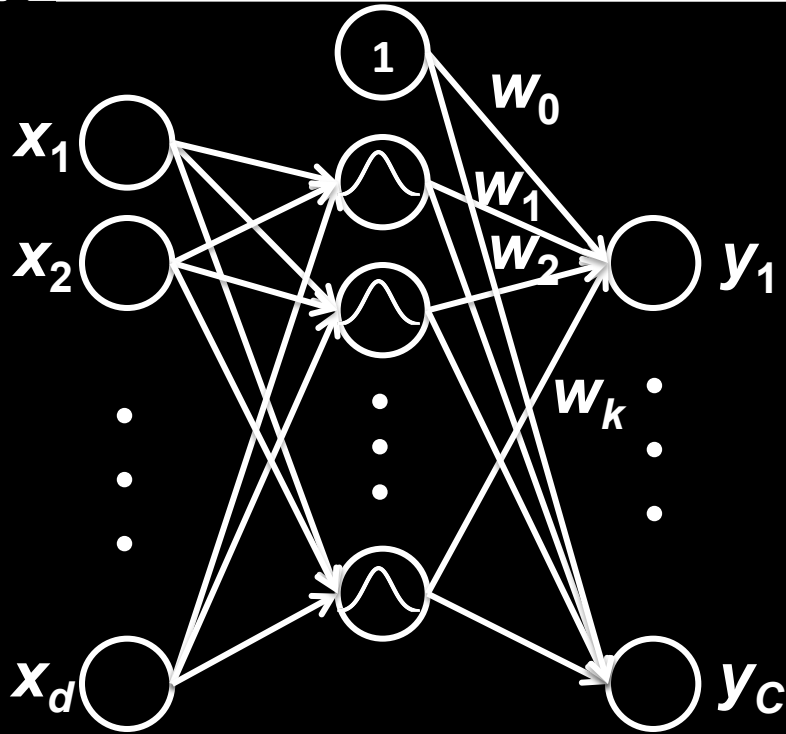


# ML Locally Weighted Regression

- **Generally, you should not need a complicated function approximation**
  - In a small enough local region a linear or quadratic function is a reasonable approximation



# ML Radial Basis Function Network



$$\begin{aligned} \mathbf{X}^{(k)} &= \mathbf{X}^{(i)}, \\ \mathbf{X}^{(k)} &= \bar{\mathbf{X}}^{(i)}, \\ &\dots \end{aligned}$$

$$\hat{f}(\mathbf{x}^{(q)}) \equiv w_0 + \sum_{k=1}^{n_k} w^{(k)} \exp\left(-\frac{\left(\mathbf{x}^{(q)} - \mathbf{x}^{(k)}\right)^2}{2\sigma^{(k)2}}\right)$$

# ML Cased-based Reasoning

- **Address correction**
  - Street number
  - Street # suffix
  - Street name
  - Directional
  - Street type
  - Unit type
  - Unit number
  - Unit #suffix
  - City
  - State
  - Zip code



# ML **Lazy versus Eager**

---

- **Computation time**
  - Lazy methods are faster during training, generally
  - Eager methods are faster at classification time, generally
- **Inductive Bias**
  - Lazy methods can consider the query instance  $x^{(q)}$
  - Eager methods make a single global approximation in advance versus local approximations at classification / test time

# ML Advantages & Disadvantages

---

- **Advantage**
  - Zero training time
  - Target function might be too complex for most algorithms, but can still be reasonably represented by local approximations
- **Disadvantage**
  - Classification can be slow
  - Typically compares all attribute values, even those that are not relevant

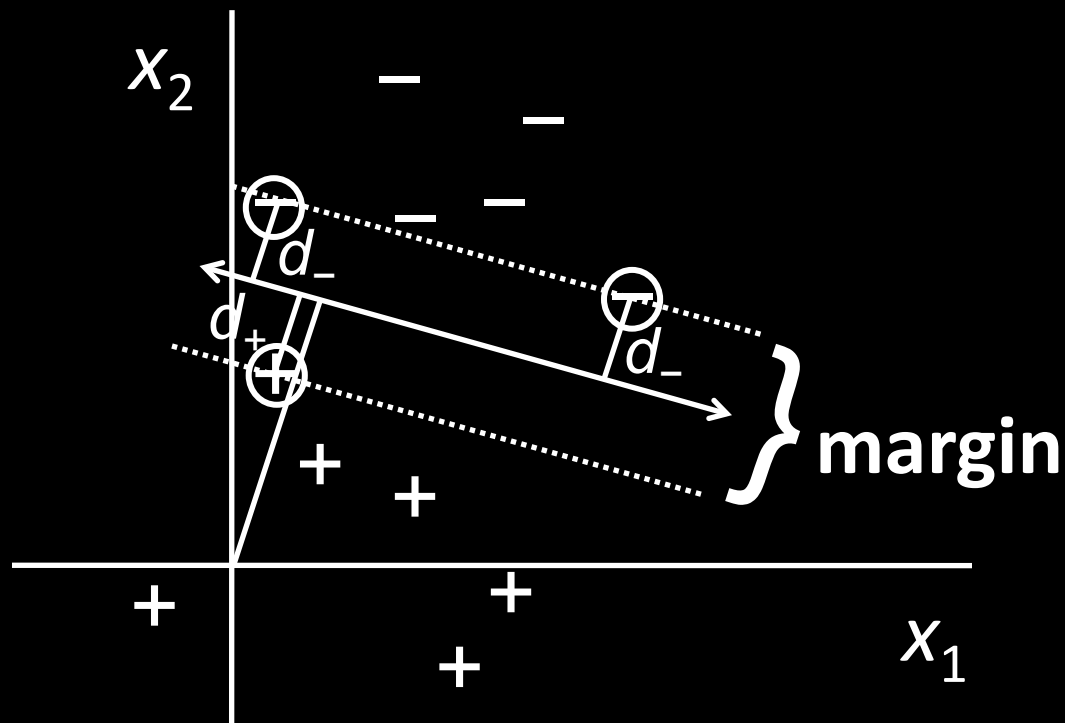
# ML Support Vector Machines (SVMs)

---

- **Pattern recognition applications**
  - Isolated handwritten digit recognition,
  - Object recognition,
  - Speaker identification,
  - Charmed quark detection,
  - Face detection in images, and
  - Text categorization.
- **Generalization usually as good and often significantly better than other methods**

# ML Linear Support Vector Machines

- Search for hyperplane that maximizes the margin ( $d_+ + d_-$ )



- **No prior knowledge**
  - E.g., would perform equally well on an image recognition if pixels were randomly permuted
- **Can be extremely slow at classification (but not typically)**

# ML Motivation for SVMs

---

- **Bias-Variance tradeoff**
- **Capacity control**
- **Overfitting**
- **Basic idea: best generalization if right tradeoff between**
  - **Error on training data and**
  - **Capacity to learn any training data without error**

# ML Statistical Learning Theory

- **Capacity control**
  - Too much capacity:  
under generalize/  
memorize
  - Too little capacity:  
over generalize/  
indiscriminant



Too many  
leaves →  
Not  
sunflower



Yellow  
circle →  
Sunflower

# ML Generalization Bounds

---

- Under what circumstances and how quickly does the mean of  $f$  converge to true mean

# ML Error, Loss, and Risk

- (Expected) *Risk* is the expected test set error

$$R(\alpha) = \int \frac{1}{2} |y - \hat{f}(\mathbf{x}, \alpha)| dP(\mathbf{x}, y)$$

- Empirical Risk is the mean training set error

$$R_{emp}(\alpha) = \frac{1}{2N} \sum_{i=1}^N |y^{(i)} - \hat{f}(\mathbf{x}^{(i)}, \alpha)|$$

- Loss is the error measure for  $\langle \mathbf{x}_i, y_i \rangle$

$$Loss = \frac{1}{2} |y^{(i)} - \hat{f}(\mathbf{x}^{(i)}, \alpha)|$$

# ML Risk (or Generalization) Bounds

- Risk bound with probability  $(1 - \eta)$ :

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(1 + \log 2N/h) - \log \eta/4}{N}}$$

- $h \geq 0$ , Vapnik Chervonenkis (VC) dimension
- $h$  is a measure of capacity of learner
- Second term on right is the VC confidence
- Independent of  $P(x,y)$ , but must be i.i.d.
- Can't compute  $R(\alpha)$
- If you know  $h$ , easy to compute risk bound

# ML Structural Risk Minimization

- **Essential idea:**
  - Given several different learners ( $H$  spaces) and a fixed sufficiently small  $\eta$ , choosing learner ( $H$  space) that minimizes right side of equation  
→ choosing learner which gives lowest bound on the actual risk
- **Principled method for choosing a learner**

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(1 + \log 2N/h) - \log \eta/4}{N}}$$

# ML Structural Risk Minimization

- **Given a fixed family of learning machines, to the extent the bound is tight for at least one learner, this is the best learner to use**
- **When the bound is not tight for any of the learners, experimental evidence typically rules**

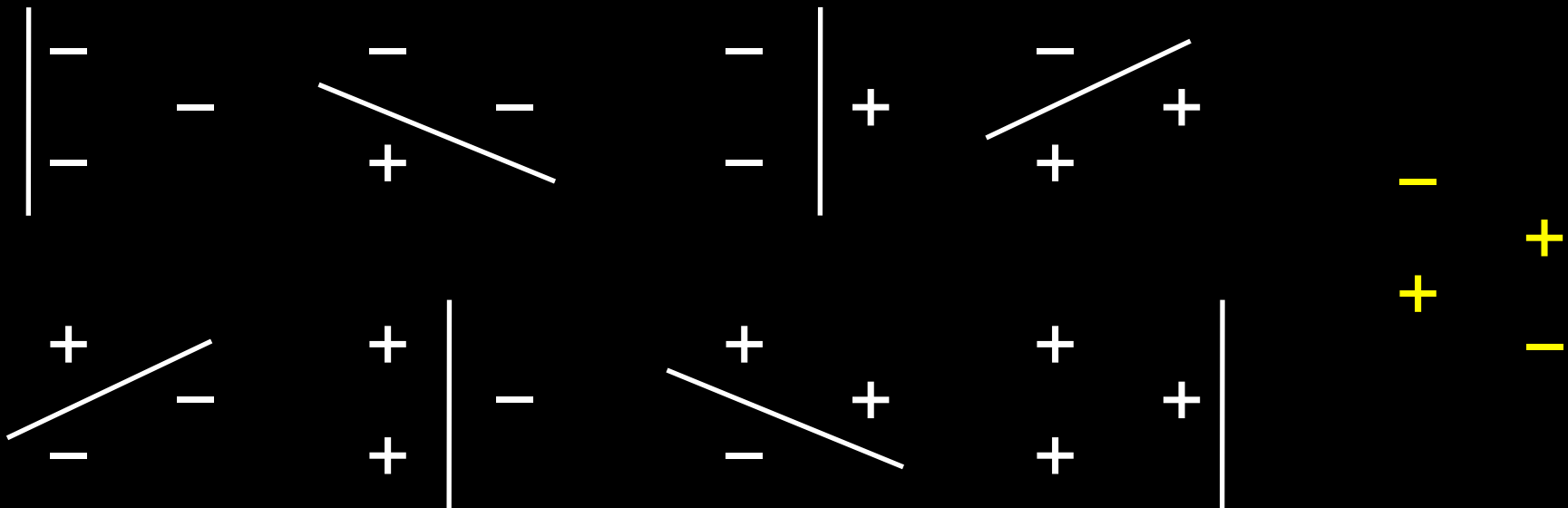
# VC Dimension

---

- Property of a set of functions  $\{f(\alpha)\}$
- Assume Boolean-valued functions
- $N$  points can be labeled  $2^N$  ways
- If for some set of  $N$  points and every possible labeling, some  $f(\alpha)$  in  $\{f(\alpha)\}$  can correctly label the data, then the set of points is *shattered* by  $\{f(\alpha)\}$
- VC dimension,  $h$ , for  $\{f(\alpha)\}$  is the max # of training points that can be shattered by  $\{f(\alpha)\}$  i.e., at least 1 set of  $h$  points can be shattered

# VC Dimension

- If for some set of  $N$  points & every possible labeling of that set, some  $f(\alpha)$  in  $\{f(\alpha)\}$  can correctly label the data, then the set of points is *shattered* by  $\{f(\alpha)\}$
- Assume  $\{f(\alpha)\}$  is the set of oriented lines and  $x$  in  $\mathbb{R}^N$ , then  $h=3$



- **Theorem 1:** Consider some set of  $m$  points in  $\mathbb{R}^d$ . Choose any one of the points as the origin. Then the  $m$  points can be shattered by oriented hyperplanes if and only if the position vectors of the remaining points are linearly independent
- The VC dimension,  $h$ , of a set of oriented hyperplanes in  $\mathbb{R}^d$  is  $d+1$ .

# ML VC Dimension vs. # Parameters

---

- Generally fewer parameters  $\rightarrow$  Lower VC dimension
- Not always true

ML

# VC Dimension

---

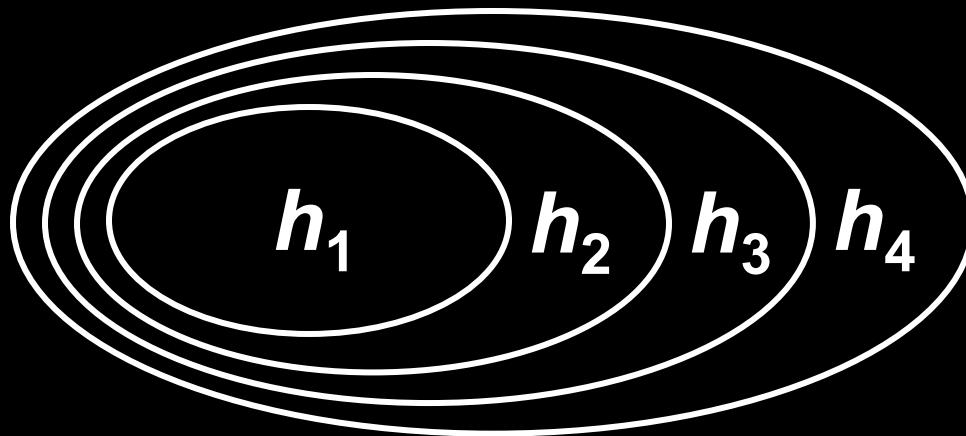
- **Monotonically increasing function in  $h$**

$$R(\alpha) \leq R_{emp}(\alpha) + \sqrt{\frac{h(1 + \log 2N/h) - \log \eta/4}{N}}$$

- **Nearest Neighbor**
  - Infinite VC Dimension
- **Perceptron**
  - VC Dimension,  $h = d + 1$
- **Decision Tree**
  - Infinite VC Dimension
- **Logistic Regression**
  - VC Dimension,  $h = d + 1$

# ML Structural Risk Minimization

- Find the subset of functions that minimizes the actual risk
- Train a learner for each subset
- Choose the subset of functions with the minimum empirical risk + VC confidence



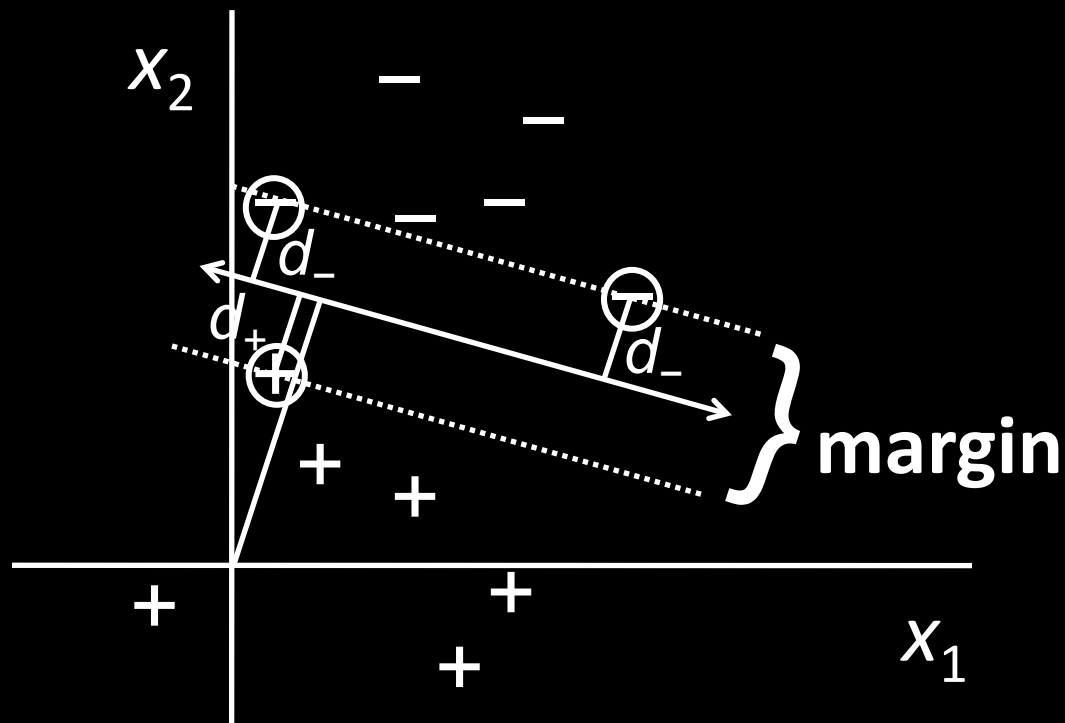
# ML Linear Support Vector Machines

---

- Separable case
- Separating hyperplane:  $wx + b = 0$ 
  - $w$  is normal to the hyperplane
  - $|b|/||w||$  is the perpendicular distance from the hyperplane to the origin
  - $||w||$  is the Euclidean norm of  $w$
- Let  $d_+$  ( $d_-$ ) be the shortest distance from the hyperplane to a positive (negative) example.
- Let  $(d_+ + d_-)$  be the *margin*

# ML Linear Support Vector Machines

- Search for hyperplane that maximizes the margin ( $d_+ + d_-$ )



# ML SVM Formulation

---

- $w x^{(i)} + b \geq +1$  for  $y^{(i)} = +1$
- $w x^{(i)} + b \leq -1$  for  $y^{(i)} = -1$
- Or  $y^{(i)}(w x^{(i)} + b) - 1 \geq 0$  for all  $i$
- Minimize  $\|w\|^2$ , subject to above constraints (one for each training instance)
- Introduce Lagrange multipliers
- Solve convex quadratic programming problem