Tuesday, 11/3/2020

EE 660

MACHINE LEARNING FROM SIGNALS: FOUNDATIONS AND METHODS

Prof. B. Keith Jenkins

Lecture 21

Lecture 21 announcements

- Homework 8 is due Friday
- Graded and commented Homework 6 (Project Proposal) is available on D2L

Lecture 21 outline

- Boosting (part 2)
- Semi-supervised learning (SSL) (part 1)
  - Introduction and assumptions
  - Types: transductive and inductive
  - Self-training models

Boosting (part 2) – Adaboost

From last lecture: Can re-arrange L_m to get eqns. for Adaboost algorithm

(*) $\phi(\underline{x}_i, \underline{\gamma}_m)$ is chosen to minimize:

$$\phi_m = \underset{\phi}{\operatorname{argmin}} \Big\{ \underbrace{\sum_{i=1}^{N} w_{i,m} \, \mathbb{I}[\tilde{y}_i \neq \phi(\underline{x}_i)]}_{\text{sum of weights of misclassified data points}} \Big\}$$

(decision stump optimization)

(**) $\beta_m = \frac{1}{2} \log \frac{1 - err_m}{err_m}$, with

$= $ weight (importance) of classifier ϕ_m

$$\begin{cases} err_m \to 0 \Rightarrow \beta_m \to \infty \\ err_m \to 1 \Rightarrow \beta_m \to -\infty \\ err_m = 0.5 \Rightarrow \beta_m = 0 \end{cases}$$

(***) $$err_m = \frac{\sum_{i=1}^{N} w_{i,m} \, \mathbb{I}[\tilde{y}_i \neq \phi(\underline{x}_i, \underline{\gamma}_m)]}{\sum_{i=1}^{N} w_{i,m}}$$

$=$ sample-weighted error rate (at m^{th} iteration), which has been minimized in (*).

Algorith: Adaboost.M1

1. Initialize $w_i = \frac{1}{N} \ \forall i$

2. For $m=1$ to M:

   (i) Train classifier $\phi_m(\underline{x})$ [1-node CART] on weighted dataset $\mathcal{D}_m$ (weights $w_{i,m}$) per (*).

   (ii) Compute $err_m = \dfrac{\sum_{i=1}^{N} w_{i,m} \, \mathbb{I}[\tilde{y}_i \neq \phi_m(\underline{x}_i)]}{\sum_{i=1}^{N} w_{i,m}}$ from (***)

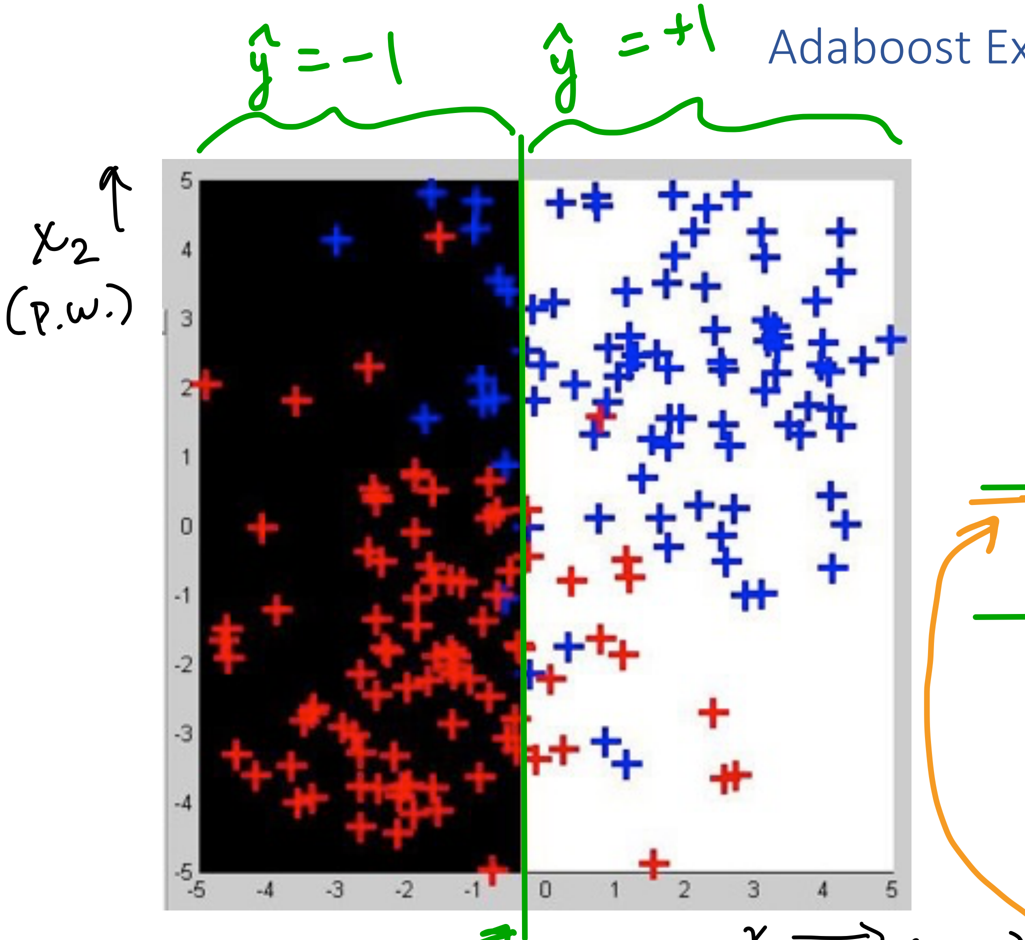
   (iii) Compute $\alpha_m = \log\left[\dfrac{1-err_m}{err_m}\right] = 2\beta_m$ in (**)

   (iv) Update $w_{i,m+1} = w_{i,m} \exp\left[\alpha_m \mathbb{I}(\tilde{y}_i \neq \phi_m(\underline{x}_i)\right] \ \forall i$

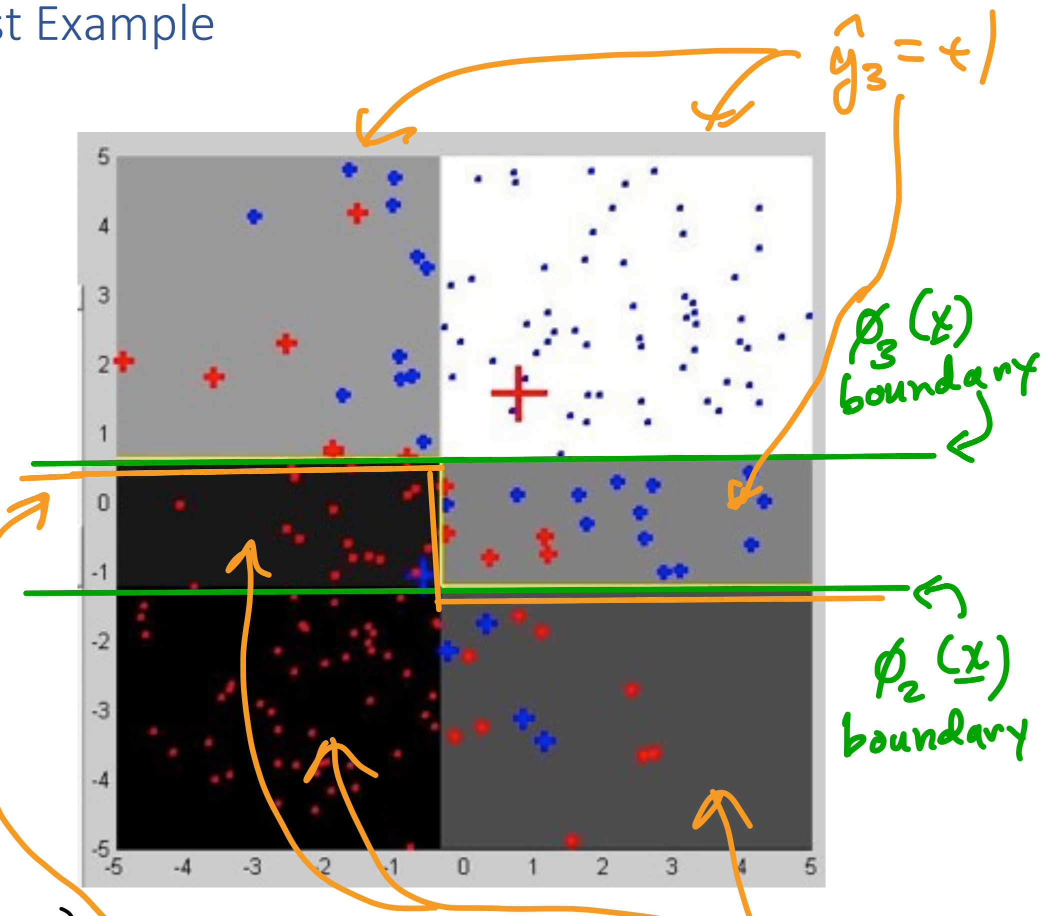
3. Return $\hat{f}(\underline{x}) = \sum_{m=1}^{M} \alpha_m \phi_m(\underline{x})$

   and $\hat{y}(\underline{x}) = \text{sign}\{\hat{f}(\underline{x})\}$

Adaboost Example

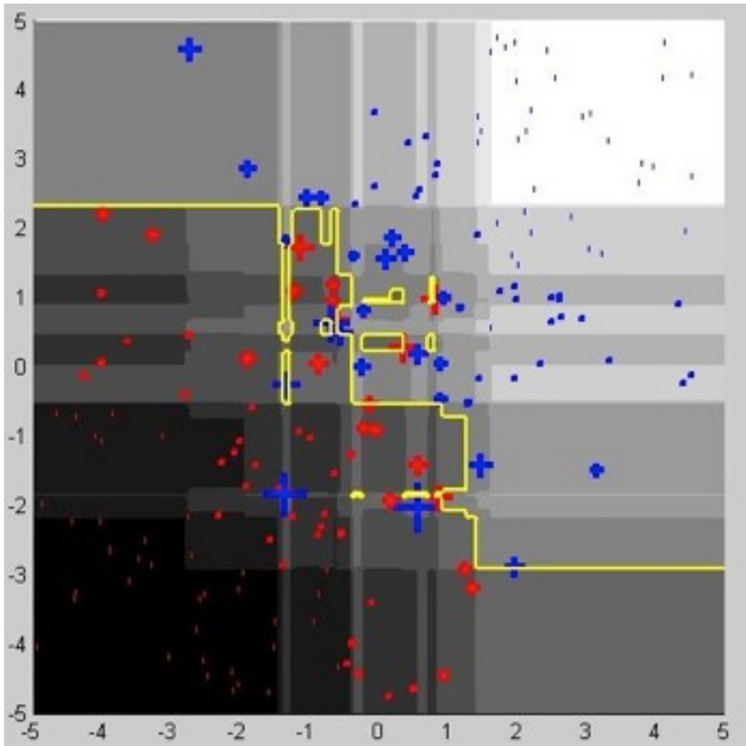


(a) m = 1 completed

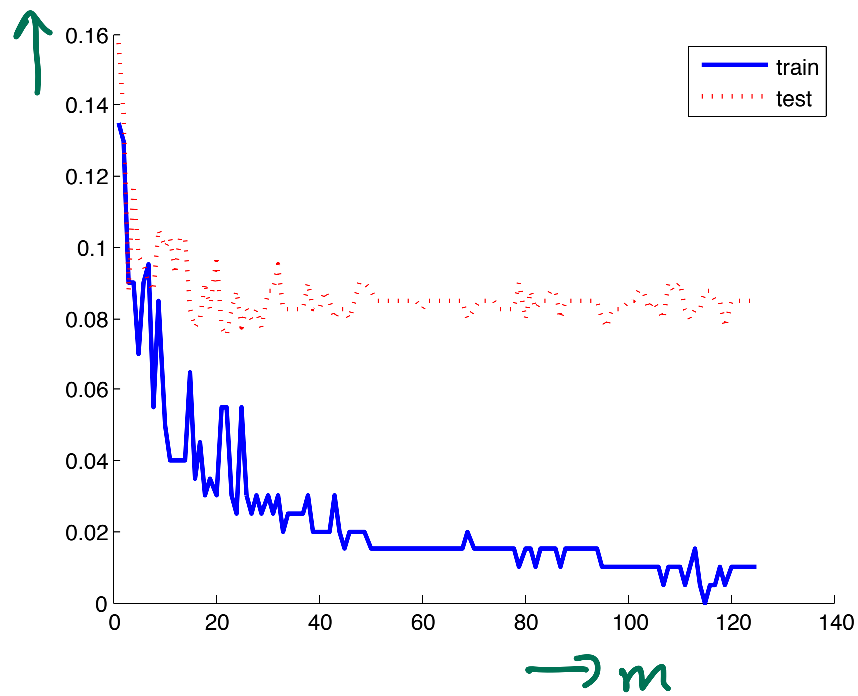


(b) m = 3 completed

Murphy Fig. 16.10



(c) m = 120 completed



Murphy Fig. 16.8: Error rate vs. m

Semi-Supervised Learning (SSL) [Zhu and Goldberg text]

Training set consists of:

l labeled instances $\mathcal{D}_L = \left\{\left(\underline{x}_i, y_i\right)\right\}_{i=1}^{l}$ and u unlabeled instances $\mathcal{D}_U = \left\{\underline{x}_j\right\}_{j=l+1}^{l+u}$

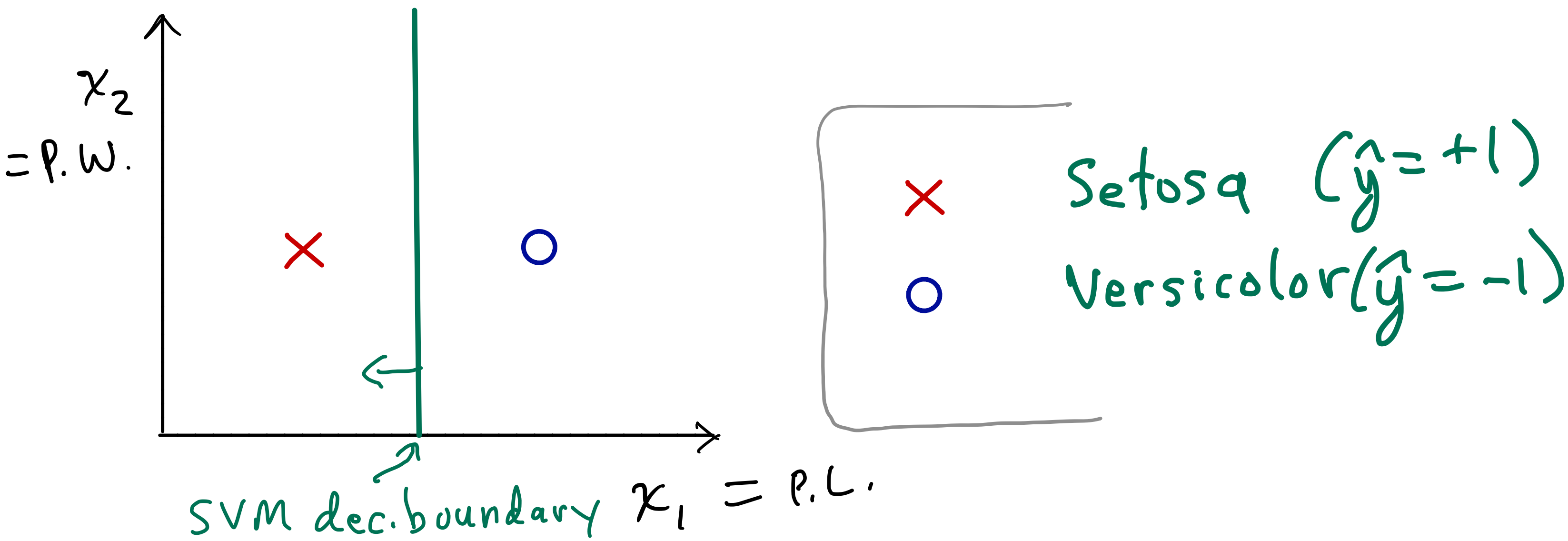
Why SSL?

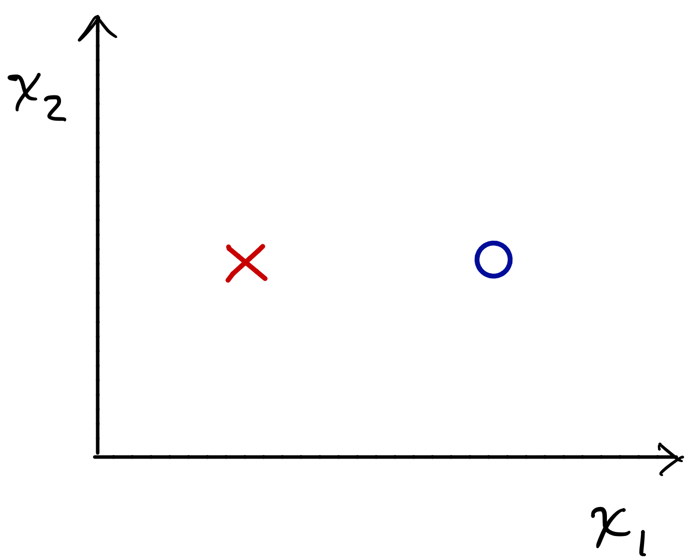
(i) Can be expensive or impractical labels on very many data pts.

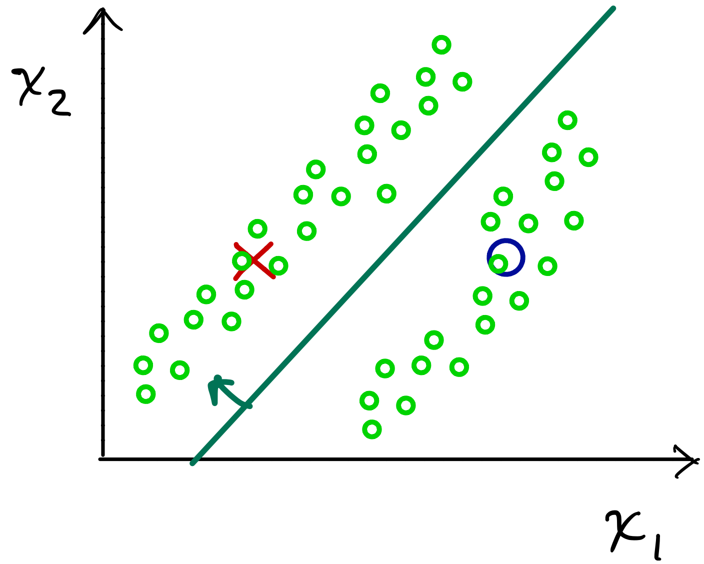
(ii) Often have access to plentiful unlabeled data points.

=> The goal is to train a system using both sets $\mathcal{D}_L$ and $\mathcal{D}_U$, and achieve better out-of-sample performance than training on $\mathcal{D}_L$ only.

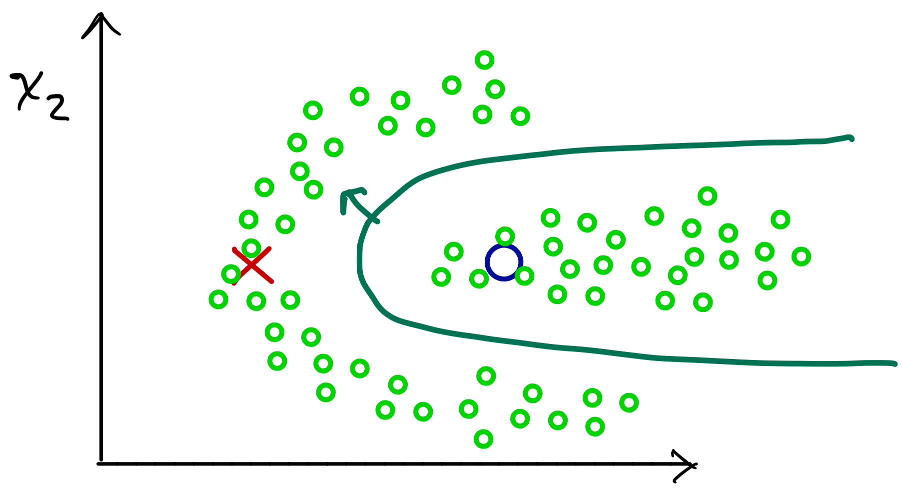
Q: Is this possible?


x_2
$= P.W.$
SVM dec. boundary
$x_1 = P.L.$
Setosa $(\hat{y} = +1)$
Versicolor $(\hat{y} = -1)$


x_2
x_1



o unlabeled training data $\mathcal{D}_U$



x_1

Assumptions

1. Labeled data points are representative (not outliers).

2. All data points are drawn i.i.d. from underlying densities:

$$\left.\begin{array}{l} p(\underline{x} \mid y) \text{ and } p(y) \\ \text{or } \; p(y \mid \underline{x}) \text{ and } p(\underline{x}) \end{array}\right\} \text{ for labeled data } \mathcal{D}_L$$

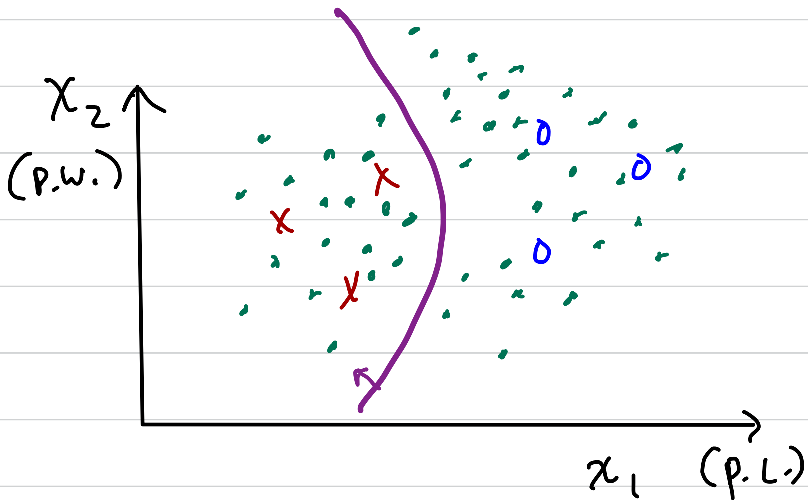
$$p(x) \qquad \text{for unlabeled data } \mathcal{D}_U$$

* are consistent

$$p(\underline{x}) = \sum_y p(\underline{x} \mid y)\, p(y)$$

2 major types of SSL

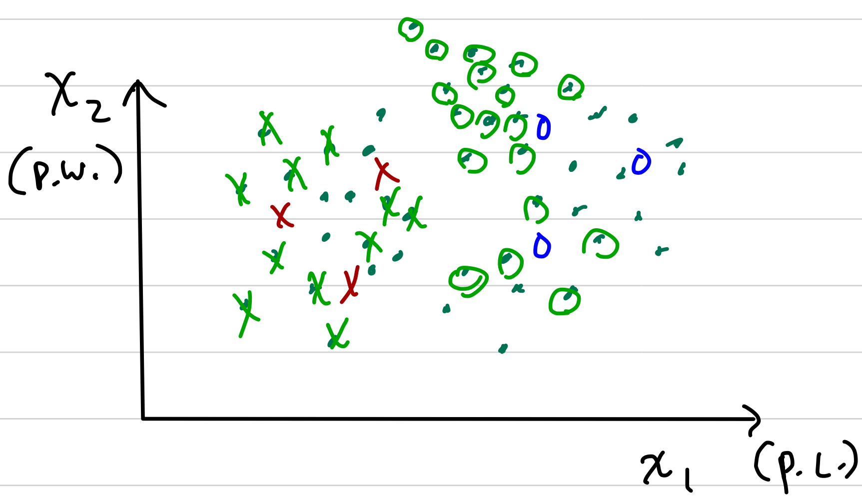
Inductive SSL learns $\hat{y} = \text{sign}\{\hat{f}(\underline{x})\}$ over all feature space $\mathcal{X}$.



x setosa
o versicolor

$\Rightarrow$ divides all feature space into decision regions

Transductive SSL learns $\hat{y}_i = \text{sign}\{\hat{f}(\underline{x}_i)\}$ $\forall \underline{x}_i \in \mathcal{D}_U$.



x setosa
o versicolor

$\Rightarrow$ only the points in $\mathcal{D}_U$ are classified