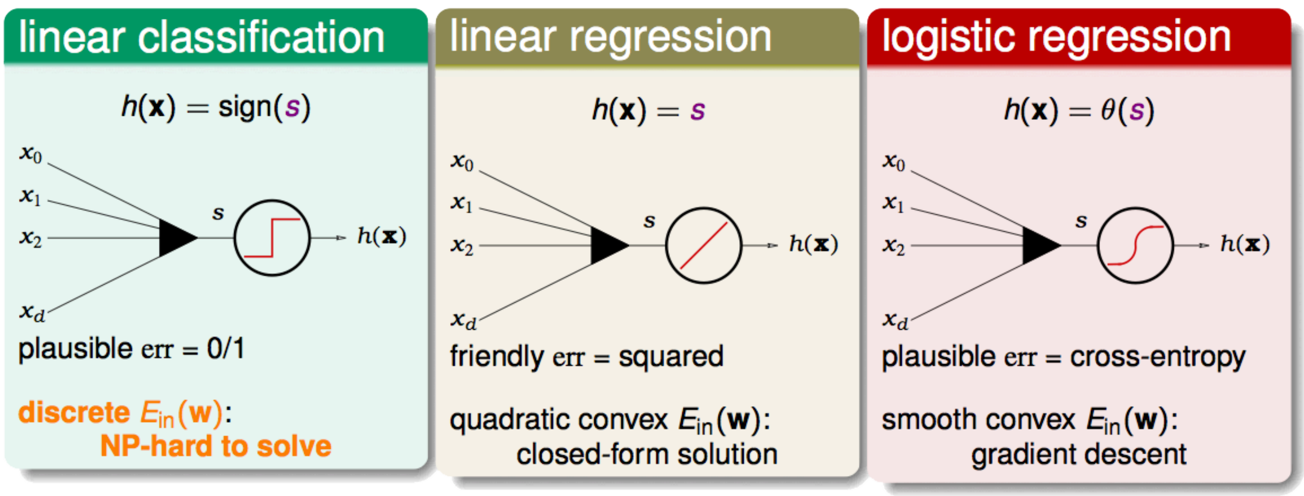
Week 11: Linear Models for Classification

Table of Contents

Linear Models for Binary Classification

1. Three linear models



- Linear Classification:
    - **Sign** of s as output
    - 0/1 error
    - Cost function $E_{in}(w)$ is a discrete, NP-hard question
- Linear Regression:
    - Outputs s directly as result
    - Squared error
    - $E_{in}(w)$ is a twice-differentiable, quadratic convex function, with closed form solution
- Logistic Regression
    - Sigmoid $\theta(s)$ as output
    - Cross-entropy as error
    - $E_{in}(w)$ is a smooth convex function, solvable using gradient descent

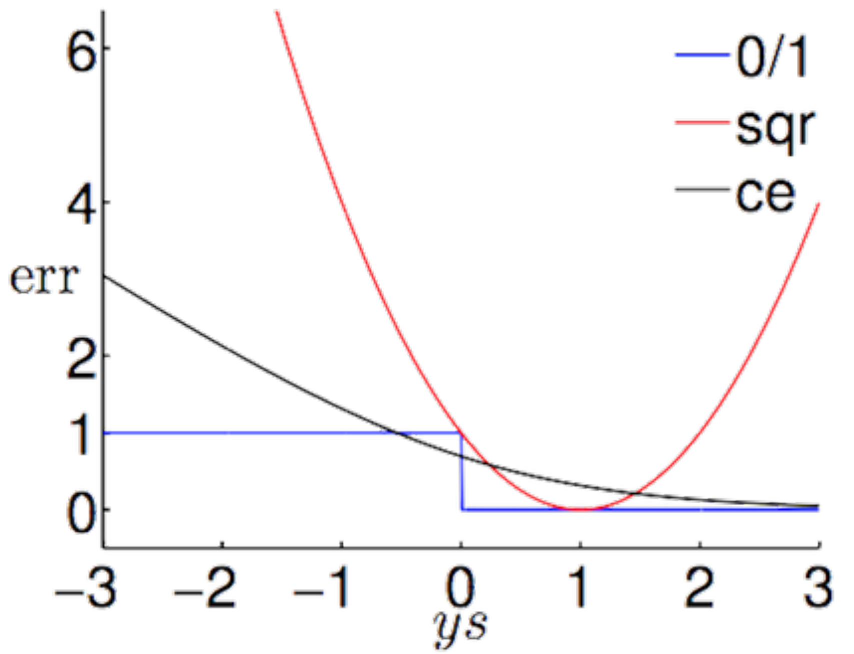
2. Given the difficulty of optimizing linear classification (recall PLA works only on linear-separable data), it's our interest to solve linear classification problem using linear regression or logistic regression (with some modification)

Error Functions Revisited

1. Main difference among the three linear models is their error functions, however, within the scope of binary classification, similarities can be drawn
    - Given linear scoring function $s = w^T x$ for binary classification $y \in \{-1, +1\}$, the following equations hold

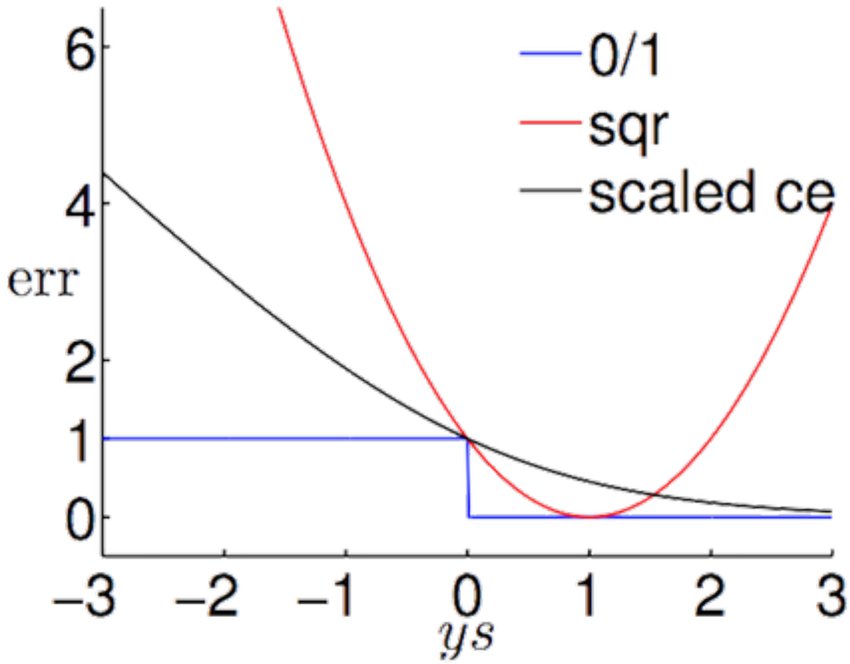
linear classification	linear regression	logistic regression
$h(\mathbf{x}) = \text{sign}(s)$	$h(\mathbf{x}) = s$	$h(\mathbf{x}) = \theta(s)$
$\text{err}(h, \mathbf{x}, y) = [\![h(\mathbf{x}) \neq y]\!]$	$\text{err}(h, \mathbf{x}, y) = (h(\mathbf{x}) - y)^2$	$\text{err}(h, \mathbf{x}, y) = -\ln h(y\mathbf{x})$
$\text{err}_{0/1}(s, y)$	$\text{err}_{\text{SQR}}(s, y)$	$\text{err}_{\text{CE}}(s, y)$
$= [\![\text{sign}(s) \neq y]\!]$	$= (s - y)^2$	$= \ln(1 + \exp(-ys))$
$= [\![\text{sign}(ys) \neq 1]\!]$	$= (ys - 1)^2$	

    - **Classification correctness score** (ys) exists in all three error functions, and is the sole independent variable with respect to err
2. Plotting 0/1 error, squred error, and cross-entropy with respect to (ys) results in



    - 0/1 error: $err = 1 \ iff ys \leq 0$
    - Squared error: large if $ys \ll 1, \ $ over-charge if$ys \gg 1$
        - Squared error is not very well-suited as error function in linear classification, as it is hyperbolic and is large when ys is large or small. Whereas 0/1 error decreases once the sign of ys becomes positive.
    - Cross-entropy error: $Monotonic\ of\ ys, \ $ small $err_{ce} \leftrightarrow$ small $err_{0/1}$
3. Noticed in the graph above, cross-entropy error is **smaller than 1** for some $ys \sim 0$. A **logrithmic-scaled** cross-entropy is therefore introduced to counter-act this difference and make sure that the error functions being evaluated all produce $err = 0 \ when \ ys = 1$

$$\textit{scaled}\ \text{ce}: err_{sce}(s, y) = log_2(1 + exp(-ys))$$



- Scaled cross-entropy provides an **upper bound** for 0/1 error in linear classificaiton

Theoretical Implication of Upper Bound

1. For any ys where $s = w^T x$, the graphs above show that

$$err_{0/1}(s, y) \leq err_{SCE}(s, y) = \frac{1}{ln2} err_{CE}(s, y)$$
$$\Rightarrow E_{in}^{0/1}(w) \leq E_{in}^{SCE}(w) = \frac{1}{ln2} E_{in}^{CE}(w)$$
$$E_{out}^{0/1}(w) \leq E_{out}^{SCE}(w) = \frac{1}{ln2} E_{out}^{CE}(w)$$

2. Applying VC bounds onto the upper bound gives

VC on 0/1:

$$\begin{aligned} E_{\text{out}}^{0/1}(\mathbf{w}) &\leq E_{\text{in}}^{0/1}(\mathbf{w}) + \Omega^{0/1} \\ &\leq \frac{1}{\ln 2} E_{\text{in}}^{\text{CE}}(\mathbf{w}) + \Omega^{0/1} \end{aligned}$$

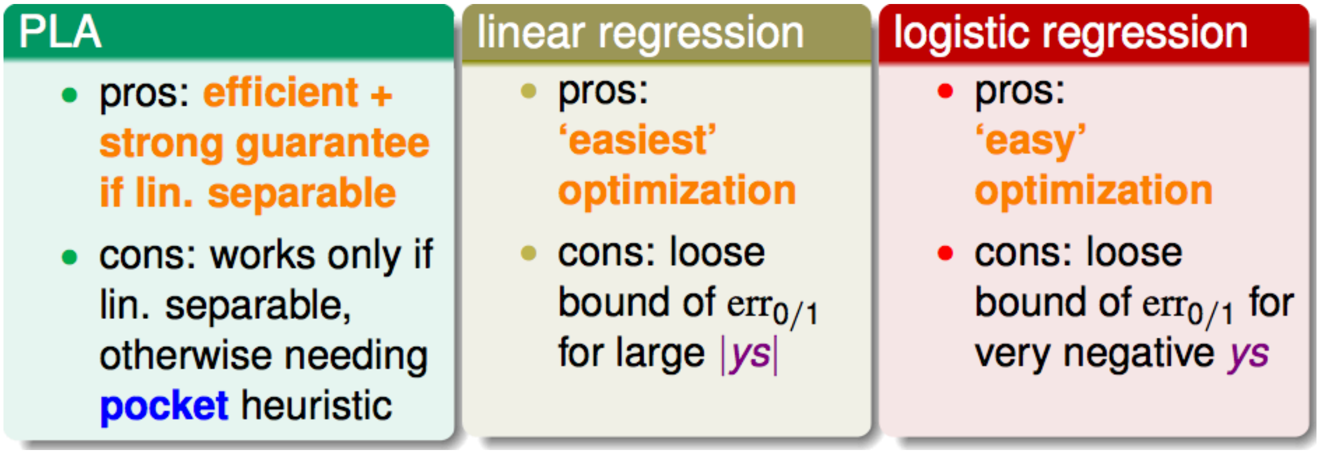
VC-Reg on CE :

$$\begin{aligned} E_{\text{out}}^{0/1}(\mathbf{w}) &\leq \frac{1}{\ln 2} E_{\text{out}}^{\text{CE}}(\mathbf{w}) \\ &\leq \frac{1}{\ln 2} E_{\text{in}}^{\text{CE}}(\mathbf{w}) + \frac{1}{\ln 2} \Omega^{\text{CE}} \end{aligned}$$

   - Small **in-sample** cross-entropy error $E_{in}^{CE}(w)$ **guarantees** small **out-of-sample** 0/1 error $E_{out}^{0/1}(w)$ $\Rightarrow$ Logistic regression can be used for linear classification
     - Linear gression also works, just looser bound

3. Regression for classification
   - Run logistic(/linear) regression on $\mathcal{D}$ with $y_n \in \{-1, +1\}$ to get w_{REG}
   - Return $g(x) = sign(w_{REG}^T x)$

4. Comparison of linear classification approaches



5. Linear regression is sometimes used to *set* w_0 for PLA/pocket/logistic regression optimization
   - Closed-form solution
   - Guaranteed, *loose* upper bound
6. Logistic regression often perferred over pocket
   - Similar computational cost as pocket, but easier optimization

Stochastic Gradient Descent

1. Motivation: Seeking an optimization scheme for logistic regression with only $O(1)$ computation time per iteration
   - Previous logistic regression optimization through gradient descent (or pocket algorithm) requires $O(N)$ time per iteration, to go through all samples and calculate error
   - In comparision, PLA requires only $O(1)$ time, but data must be linear-separable for PLA to converge
2. Graident descent for logistic regression

$$w_{t+1} \leftarrow w_t + \eta \underbrace{\frac{1}{N} \sum_{n=1}^{N} \theta(-y_n w_t^T x_n)(y_n x_n)}_{-\nabla E_{in}(w_t)}$$

   - **Approximate** update direction $v \approx -\nabla E_{in}(w_t)$ through a **single point** x_n, y_n
   - By doing so we avoid computing $\frac{1}{N} \sum_{n=1}^{N}$ for each update
     - View $\frac{1}{N} \sum_{n=1}^{N}$ as **expectation** ε over **uniform** chocie of n
3. Stochastic gradient vs. true gradient
   - **Stochastic graident**: $\nabla_w err(w, x_n, y_n)$ on **random** n
   - **True gradient**: $\nabla_W E_{in}(w) = \varepsilon_{random\ n} \nabla_W err(w, x_n . y_n)$
4. Stochastic gradient descent (SGD)
   - stochastic gradient = true gradient + **zero-mean** 'noise' directions
   - Idea: replace <span style="color:blue"true graident by stochastic gradient
   - Assumption: After **enough iterations**, **average** true graident $\approx$ **average** stochastic gradient
   - Pros
     - Simpler and cheaper compuation
     - Useful for **large data** (where full dataset iteration is impractical) or **online learning**(training data comes one at a time)
   - Cons
     - Less stable in nature, especially with large step size η

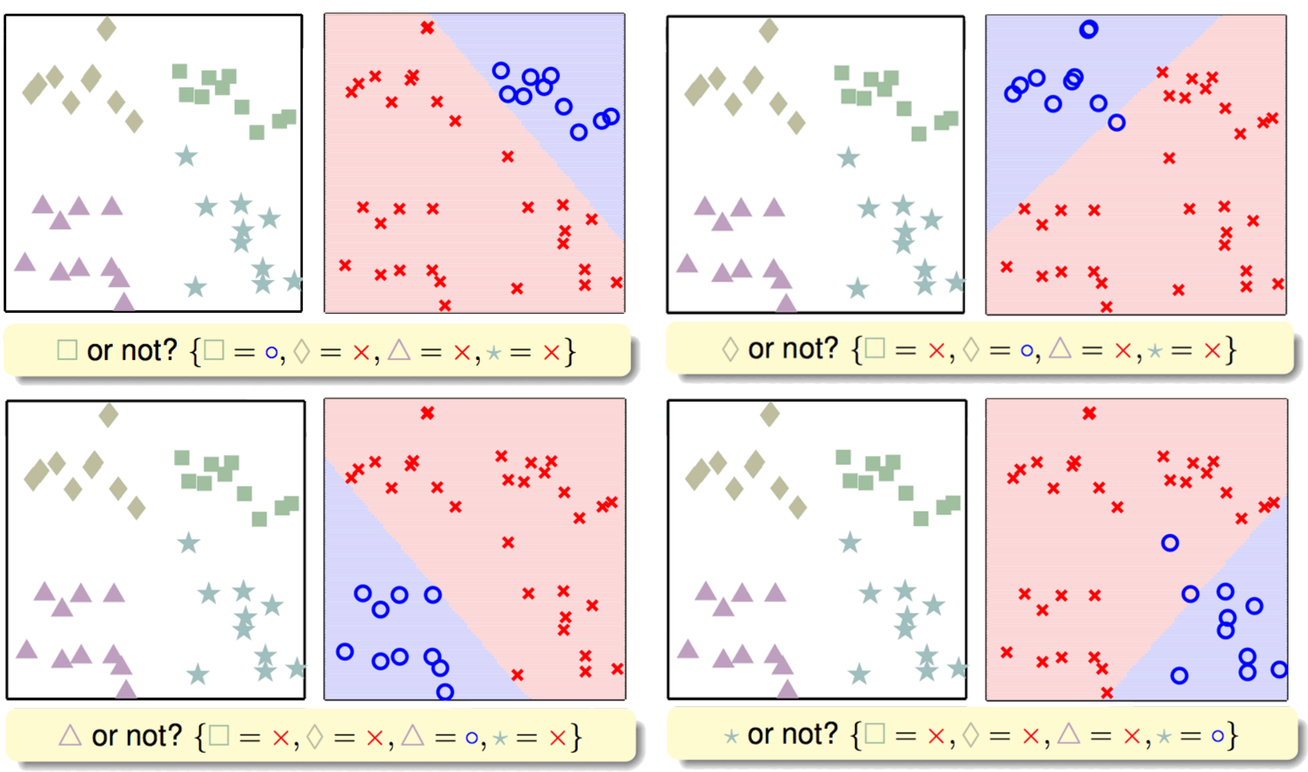
        - Update direction depends on point chosen, which might not always be accurate reflection of errors across the entire data set
    - SGD logistic regression update

$$w_{t+1} \leftarrow w_t + \eta\underbrace{\theta(-y_n w_t^T x_n)(y_n x_n)}_{-\nabla err(w_t, x_n, y_n)}$$

5. SGD and PLA
    - SGD logistic regression $w_{t+1} \leftarrow w_t + \eta \cdot \theta(-y_n w_t^T x_n)(y_n x_n)$
    - PLA $w_{t+1} \leftarrow w_t + 1 \cdot \|(y_n \neq sign(w_t^T x_n)\|(y_n x_n)$
    - SDG logistic regression $\approx$ *soft* PLA
        - Corrected by **acutal size** of the error, not 0/1 error
    - PLA $\approx$ SGD logistic regression with $\eta = 1$ when $w_t^T x_n$ is large
6. Rules of thumb for SGD
    - Set stop condition based on **number of iterations**, not by true gradient
        - Otherwise requires computation over full data set, defies the purpose of SGD
        - Stop after enough iterations
    - Use relatively small step size η to counteract instability of SGD
        - Good step size ~ 0.1 when x in proper range

Multiclass Classification

1. One-versus-All (OVA)



    - Identifying **one class at a time**. Combine all one-class classification models to form the final multiclass model
    - Ties (areas where more than one label is present) are handled by calculating *conditional* probabilities w.r.t each class, and use the class with **highest** conditional probability as the final

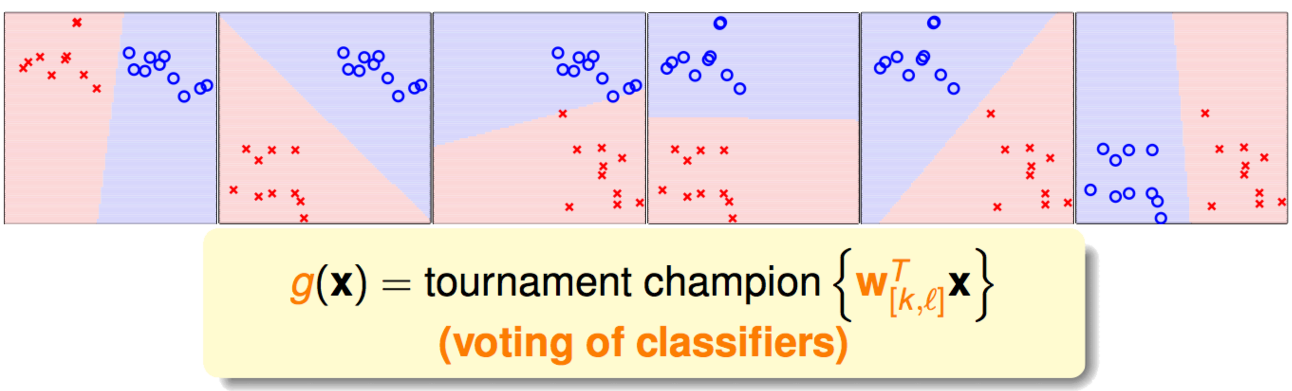
    label
    2, OVA decomposition
  - For $k \in y$
    - Obtain $w_{[k]}$ by running **logistic regression** on

$$\mathcal{D}_{[k]} = \{(x_n, y'_n = 2\|y_n = k\| - 1)\}_{n=1}^{k}$$

    - Return

$$g(x) = \arg \max_{k \in y} (w_{[k]}^T x)$$

2. Pros and Cons of OVA
   - Pros
     - Efficient
     - Can be coupled with **any logistic regression-like approaches**
   - Cons
     - Tranining set $D_{[k]}$ is often **unbalanced** when number of classese K is large
     - OVA in such case tend to perform poorly on classes with few appearances in training set
       - e.g. All but one sub hypothesis returns -1. End up picking one with highest confidence for -1, which is incorrect
3. Multinomial logistic regression
   - Extension of OVA
   - Requires probabilities produced by all sub hypotheses on a given class to **sum up to 1**
   - Better performance on unbalanced training set
4. One-versu-one (OVO)



   - Compare one **pair** of classes at a time. Combine all pairwise classification models, and choose the class that "wins" in most comparisons to be the final label for a given area.
   - For $(k, l) \in y \times y$, obtain $w_{[k,l]}$ by running **linear binary classification** on

$$\mathcal{D}_{[k,l]} = \{(x_n, y'_n = 2\|y_n = k\| - 1) : y_n = k \ or \ y_n = l\}$$

   - Return $g(x) =$ tournament champion $\{w_{[k,\ l]}^T x\}$
5. Pros and cons of OVO
   - Pros
     - Efficient ("smaller" training problems, comparing strictly between two classes, not one against all other classes)

    - Can be paired with **any binary classification approaches**
    - Stable (due to "tournament voting"), less susceptable to unbalanced training set
  - Cons
    - More memory usage, $O(k^2)w_{[k,l]}$
    - Slower prediction, more training