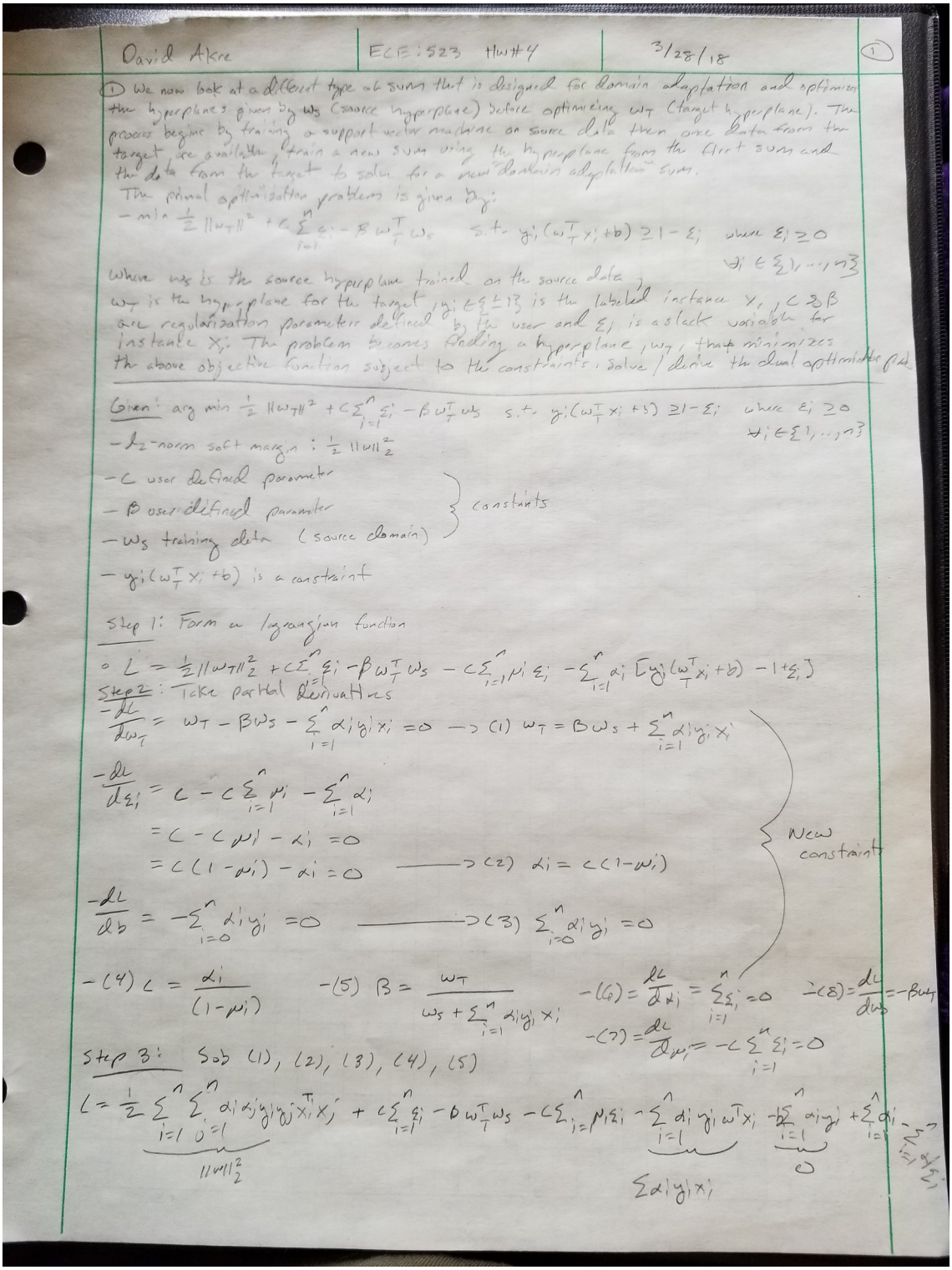
ECE: 523 HW #4
David Akre

Theory

Support Vector Machines and Boosting



David Akre | ECE: 523 HW#4 | 3/28/18 | (1)

(1) We now look at a different type of svms that is designed for domain adaptation and optimizes the hyperplanes given by w_S (source hyperplane) before optimizing w_T (target hyperplane). The process begins by training a support vector machine on source data then once data from the target are available train a new svm using the hyperplane from the first svms and the data from the target to solve for a new "domain adaptation" svm.

The primal optimization problem is given by:

$$-\min \frac{1}{2}\|w_T\|^2 + C\sum_{i=1}^{n}\xi_i - \beta w_T^T w_S \quad s.t.\ y_i(w_T^T x_i + b) \geq 1-\xi_i \quad \text{where } \xi_i \geq 0 \quad \forall i \in \{1,\dots,n\}$$

where w_S is the source hyperplane trained on the source data,
w_T is the hyperplane for the target, $y_i \in \{-1,1\}$ is the labeled instance x_i, C & β are regularization parameters defined by the user and ξ_i is a slack variable for instance x_i. The problem becomes finding a hyperplane, w_T, that minimizes the above objective function subject to the constraints, solve / derive the dual optimization problem

Given: $\arg\min \frac{1}{2}\|w_T\|^2 + C\sum_{i=1}^{n}\xi_i - \beta w_T^T w_S \quad s.t.\ y_i(w_T^T x_i + b) \geq 1-\xi_i \quad \text{where } \xi_i \geq 0 \quad \forall i \in \{1,\dots,n\}$

- l_2-norm soft margin: $\frac{1}{2}\|w\|_2^2$
- C user defined parameter
- β user defined parameter
- w_S training data (source domain)

} constants

- $y_i(w_T^T x_i + b)$ is a constraint

Step 1: Form a lagrangian function

$$\mathcal{L} = \frac{1}{2}\|w_T\|_2^2 + C\sum_{i=1}^{n}\xi_i - \beta w_T^T w_S - C\sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\alpha_i\left[y_i(w_T^T x_i + b) - 1 + \xi_i\right]$$

Step 2: Take partial derivatives

$$\frac{-d\mathcal{L}}{dw_T} = w_T - \beta w_S - \sum_{i=1}^{n}\alpha_i y_i x_i = 0 \rightarrow (1)\ w_T = \beta w_S + \sum_{i=1}^{n}\alpha_i y_i x_i$$

$$\frac{-d\mathcal{L}}{d\xi_i} = C - C\sum_{i=1}^{n}\mu_i - \sum_{i=1}^{n}\alpha_i$$

$$= C - C\mu_i - \alpha_i = 0$$

$$= C(1-\mu_i) - \alpha_i = 0 \rightarrow (2)\ \alpha_i = C(1-\mu_i)$$

$$\frac{-d\mathcal{L}}{db} = -\sum_{i=0}^{n}\alpha_i y_i = 0 \rightarrow (3)\ \sum_{i=0}^{n}\alpha_i y_i = 0$$

New constraint

$$-(4)\ C = \frac{\alpha_i}{(1-\mu_i)} \qquad -(5)\ \beta = \frac{w_T}{w_S + \sum_{i=1}^{n}\alpha_i y_i x_i}$$

$$-(6) = \frac{d\mathcal{L}}{d\alpha_i} = \sum_{i=1}^{n}\xi_i = 0 \qquad -(8) = \frac{d\mathcal{L}}{dw_S} = -\beta w_T$$

$$-(7) = \frac{d\mathcal{L}}{d\mu_i} = -C\sum_{i=1}^{n}\xi_i = 0$$

Step 3: Sub (1), (2), (3), (4), (5)

$$\mathcal{L} = \underbrace{\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j}_{\|w\|_2^2} + C\sum_{i=1}^{n}\xi_i - \beta w_T^T w_S - C\sum_{i=1}^{n}\mu_i\xi_i - \underbrace{\sum_{i=1}^{n}\alpha_i y_i w^T x_i}_{\sum\alpha_i y_i x_i} - \underbrace{b\sum_{i=1}^{n}\alpha_i y_i}_{0} + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i\xi_i$$

$$L = \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + C\sum_{i=1}^{n}\xi_i - B w_T^T w_S - C\sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\alpha_i y_i x_i + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i\xi_i$$

$$= -\frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j + C\sum_{i=1}^{n}\xi_i - B w_T^T w_S - C\sum_{i=1}^{n}\mu_i\xi_i + \sum_{i=1}^{n}\alpha_i - \sum_{i=1}^{n}\alpha_i\xi_i$$

Δ : $\sum_{i=1}^{n}\sum_{j=1}^{n}\alpha_i y_i x_i$

∴ Sub in (1)

$$= \Delta + C\sum_{i=1}^{n}\xi_i - B w_S (B w_S + \sum \alpha_i y_i x_i) + \sum_{i=1}^{n}\alpha_i - C\sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\alpha_i\xi_i$$

$$= \Delta + \sum_{i=1}^{n}\alpha_i - B w_S + \sum_{i=1}^{n}\alpha_i y_i x_i + C\sum_{i=1}^{n}\xi_i + C\sum_{i=1}^{n}\mu_i\xi_i - \sum_{i=1}^{n}\alpha_i\xi_i \rightarrow \text{goes to zero from constraint (c)}$$

$$= \Delta + \sum_{i=1}^{n}\alpha_i (1 - B y_i x_i^T w_S)$$

$$\therefore \sum_{i=1}^{n}\alpha_i (1 - B y_i x_i^T w_i) - \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{n}\alpha_i\alpha_j y_i y_j x_i^T x_j$$

∴ All constraints are listed on step 2

(2) In a couple paragraphs answer: Why does boosting (generally) work?

∴ Boosting generally works because this algorithm combines a series of weak learning algorithms into a strong one. The algorithm starts with a set of inputs, T learning rounds and a hypothesis class. It then initializes D_t to $1/N$ to act as an equal weight to be applied in a higher manner towards misclassifications. After this the algorithm will then iterate over 1 to N to determine the following:

(1) classifier based on minimum error for h, S, and D_t (eg hypothesis, data set, weight) optimization

(2) ε_t sum from the weights D_t (ie weighted loss)

(3) α_t weight based on classifier

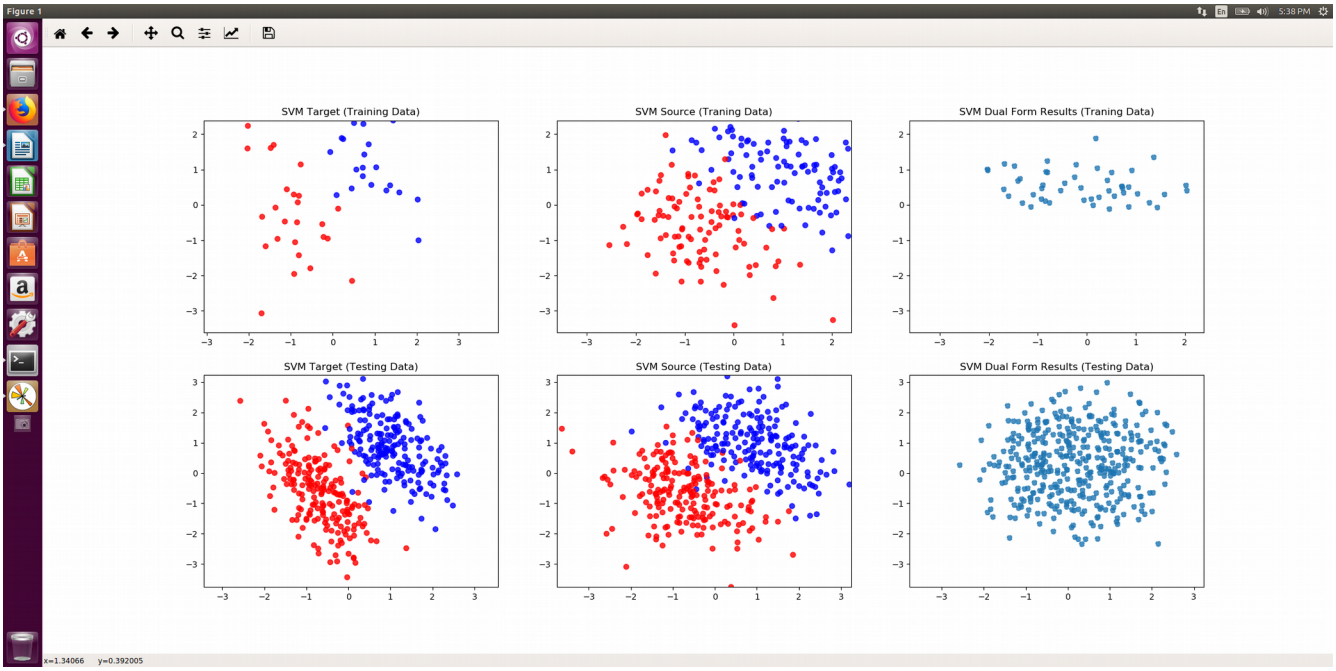
(4) D_{t+1} adjusted optimization weight based on the previous optimization weight over Z_t constant multiplied by the exponential of α_t, y_i, and $h_t(x_i)$

Thus on every iteration the optimizing factor/weight will be adjusted to be higher for misclassified hypothesis-data inputs, and conversely be less weight for proper classifications. Overall the boosting algorithm provides for a means to make predictions above 50% correct (ie 51% correct). Additionally based on Z_t derivation one can see incorrect classifications can be no more than $\frac{1}{2}$ or 50%.

Practice

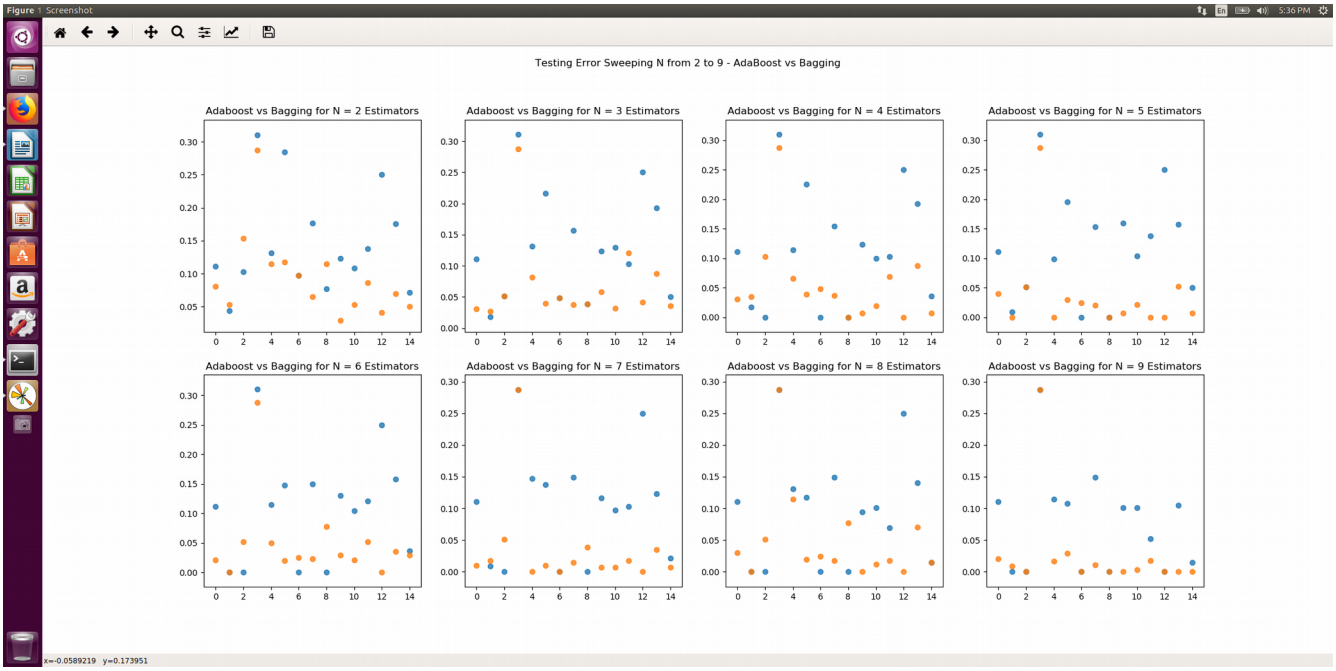
Support Vector Machines

Python Matplotlib Output:
As one can see from the matplotlib output the SVM converges the source and target data are minimized more towards the mean of where the SVM would create its decision boundary. The training data minimization is more effective than the testing minimization with cvxpy.



Ensembles

Python Matplotlib output:
As one can see from the matplotlib output below the testing error does converge close to 0 for Bagging, and it moves more slowly for AdaBoosting.



Console output Below (Mapping between X axis values and the csv file):

```
Terminal
[Fri Mar 30 05:36 PM | dakre@dakre-VirtualBox ~/machine_learning/hw4 (master)]
-> python ensembles.py
X-Axis Identifier
x = 0 coordinates with annealing_test.csv
x = 1 coordinates with breast-cancer-wisc-diag_test.csv
x = 2 coordinates with breast-cancer-wisc-prog_test.csv
x = 3 coordinates with congressional-voting_test.csv
x = 4 coordinates with haberman-survival_test.csv
x = 5 coordinates with cylinder-bands_test.csv
x = 6 coordinates with conn-bench-sonar-mines-rocks_test.csv
x = 7 coordinates with adult_test.csv
x = 8 coordinates with echocardiogram_test.csv
x = 9 coordinates with credit-approval_test.csv
x = 10 coordinates with bank_test.csv
x = 11 coordinates with heart-hungarian_test.csv
x = 12 coordinates with audiology-std_test.csv
x = 13 coordinates with breast-cancer_test.csv
x = 14 coordinates with breast-cancer-wisc_test.csv
[Fri Mar 30 05:37 PM | dakre@dakre-VirtualBox ~/machine_learning/hw4 (master)]
->
```