TDA 206: Discrete Optimization

TA: Vinay Jethava, jethava@chalmers.se
Office Hours: Friday, 2.00-5.00 p.m.

Due date: February 22, 2011

Exercise 4

General Instructions This exercise uses the CVX[1] optimization package. You can run matlab with correct CVX paths by using the command *cvx* at the command line on student lab machines. The command *cvx_where* at matlab prompt provides the full path to the CVX installation.

Part I — Max-margin classifier

This exercise compares a linear programming based classifier with a max-margin classifier. Let $D = \{(x_i, y_i)\}_{i=1}^{N}$ denote training data where $x_i \in \Re^d$ are points in $d-$dimensional space and $y_i \in \{-1, +1\}$ are the corresponding labels. A linear classifier can be written as $\hat{y} = f(x) = \text{sign}(w^\top x - b)$, where $w \in \Re^d$ and $b \in \Re$ are the two unknown parameters. The unknown parameters w, b are learnt from the training data. Then, a new test point x_1 can be classified by computing $\hat{y}_1 = \text{sign}(w^\top x_1 - b)$.

One possible approach for learning the unknown parameters w, b from the training data is using the linear programming formulation given below:

$$\begin{array}{ll} \min_{w,b,\delta} & \delta \\ \text{s.t.} & y_i(w^\top x_i - b) \geq 1 - \delta \quad \forall\, 1 \leq i \leq N \\ & \delta \geq 0 \end{array} \tag{1}$$

A more robust approach is using the maximum margin classifier. The following optimization problem describes a soft margin classifier

$$\begin{array}{lllll} \min_{w,b,\xi} & \frac{1}{2}\,||w||_2^2 + C\sum_{i=1}^{N}\xi_i & & & \\ \text{s.t.} & y_i(w^\top x_i - b) \geq 1 - \xi_i & \forall 1 \leq i \leq N & [\alpha_i] \\ & \xi_i \geq 0 & \forall 1 \leq i \leq N & [\rho_i] \end{array} \tag{2}$$

where C is a user-controlled parameter. The dual formulation to (2) is given by:

$$\begin{array}{ll} \max_{\alpha} & \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j x_i^\top x_j \\ \text{s.t.} & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad \forall\, 1 \leq i \leq N \end{array} \tag{3}$$

[1] `http://cvxr.com/cvx/`

The optimal w^* has the form $w^* = \sum_{i=1}^{N} \alpha_i^* y_i x_i$.

Questions

1. Implement the linear and max-margin classifiers in (1) and (2) using CVX and run it on the dataset `http://www.cse.chalmers.se/edu/year/2011/course/TDA206_Discrete_Optimization/hw4/exercise1.mat` containing a matrix *training_data* and vector *training_label* using $C = 1$.
2. On the same plot, show the following:
   a) The points belonging to the two classes using different colors.
   b) The line $w^\top x - b = 0$ corresponding to the classifier in (1).
   c) The line $w^\top x - b = 0$ and lines $w^\top - b = \pm 1$ (dashed) in (2).
   d) Highlight the points having dual variable $\alpha_i^* \neq 0$ in (2).
3. Experiment with the soft-margin setting as folllows:
   a) Remove the terms ξ_i in (2). ("hard" margin)
   b) Explore different values of $C \in (0, \infty)$.

Part II — Weighted Max-Cut

This exercise explores semi-definite relaxation of the weighted max-cut problem. Given a graph $G = (V, E)$ with vertices $V = 1, \ldots, n$ and non-negative weights w_{Ij} for each edge $(i, j) \in E$, the weighted maximum cut $Y : V \to \{-1, 1\}$ is given by:

$$\begin{array}{ll} \max_Y & \frac{1}{2} \sum_{(i,j)\in E} w_{ij}(1 - y_i * y_j) \\ \text{s.t.} & y_i^2 = 1 \quad (\text{OR } y_i \in \{-1, +1\}) \quad \forall i \in V \end{array} \tag{4}$$

This is a Quadratic Integer Program (QIP) which is known to be NP-Complete. Multiple algorithms have been developed which provide an approximate solution to the above problem.

1. *Random approximation* A $\frac{1}{2}$-factor approximation algorithm which independently assigns each vertex v_i to S or $\bar{S}$ with equal probability i.e.

$$P(y_i = +1) = P(y_i = -1) = \frac{1}{2}, \forall 1 \leq i \leq N \tag{5}$$

2. *Greedy approximation* A $\frac{1}{2}$-factor approximation algorithm is based on iterating through the vertices and greedily assigning vertex i into set S or $\bar{S}$ based on which placement maximizes the weight of the cut w.r.t. already assigned vertices $\{1, \ldots, i-1\}$.

3. *SDP relaxation*The SDP relaxation works by changing the labels y_i to vectors $\vec{y}_i \in \Re^n$ subject to the constraint $||\vec{y}_i||_2 = 1$. This can be formulated as an semidefinite program[2] given as:

$$\begin{array}{ll} \min_X & \sum_{i,j} w_{ij} x_{ij} \quad (= Tr(WX)) \\ \text{s.t.} & diag(X) = 1 \\ & X \succcurlyeq 0 \end{array} \tag{6}$$

[2]See course website for more information

Then, X is positive semi-definite and can be factorized as $X = VV^\top$ using Cholesky factorization. The overall algorithm is given as follows:

Algorithm 1 Goemans & Williamson algorithm

Input: W {weight matrix}
  Find X^* as in (6) {S.D.P. Formulation}
  Find $V = chol(X^*)$ {Cholesky Factorization}
  Choose $\vec{r} \in \mathcal{B}^n$ {Random direction in unit ball}
  return $S = sign(V * r)$ {Assignment using randomized rounding}

Sometimes, the Cholesky Factorization fails due to numerical errors. Then, use X instead of V in later steps.

Questions The following questions use the dataset at `http://www.cse.chalmers.se/edu/year/2011/course/TDA206_Discrete_Optimization/hw4/graphs.tgz`. This dataset has a small manually annotated graph in the file *small_W.txt* and a number of moderate graphs in the path *${PWD}/mac/rudy/*. The format of the files is:

```
nV nE
x1 y1 w1
x2 y2 w2
...
```

where nV and nE on the first line denote the number of vertices and edges in the graph. Each subsequent line gives the weight corresponding to an edge.[3]

1. Implement the three algorithms discussed and compare their results on the given datasets. An template for the code is provided at `http://www.cse.chalmers.se/edu/year/2011/course/TDA206_Discrete_Optimization/hw4/template.m`.
2. We note that V is a collection of vectors that we are trying to cluster into two classes. Randomized rounding procedure is one way to do it (with theoretical guarantees). This question investigates what happens in case we use an alternate scheme. Use the rows of the V matrix obtained in 1 for hierarchical clustering[4] into two clusters. The assignment obtained gives the max-cut using this scheme. How does the assignment compare for (*a*) *small_W.txt* (small graph) (*b*) *p01_100.7* (moderate graph).

[3] For more details, see `http://biqmac.uni-klu.ac.at/biqmaclib.html`
[4] See matlab functions *linkage*, *cluster*, *dendrogram*