Machine Learning- Exercise 2

Least Square Linear Classifiers, SVM

Ali Athar & Idil Esen Zuelfikar
<lastname>@vision.rwth-aachen.de

RWTH Aachen University - Computer Vision Group
http://www.vision.rwth-aachen.de/

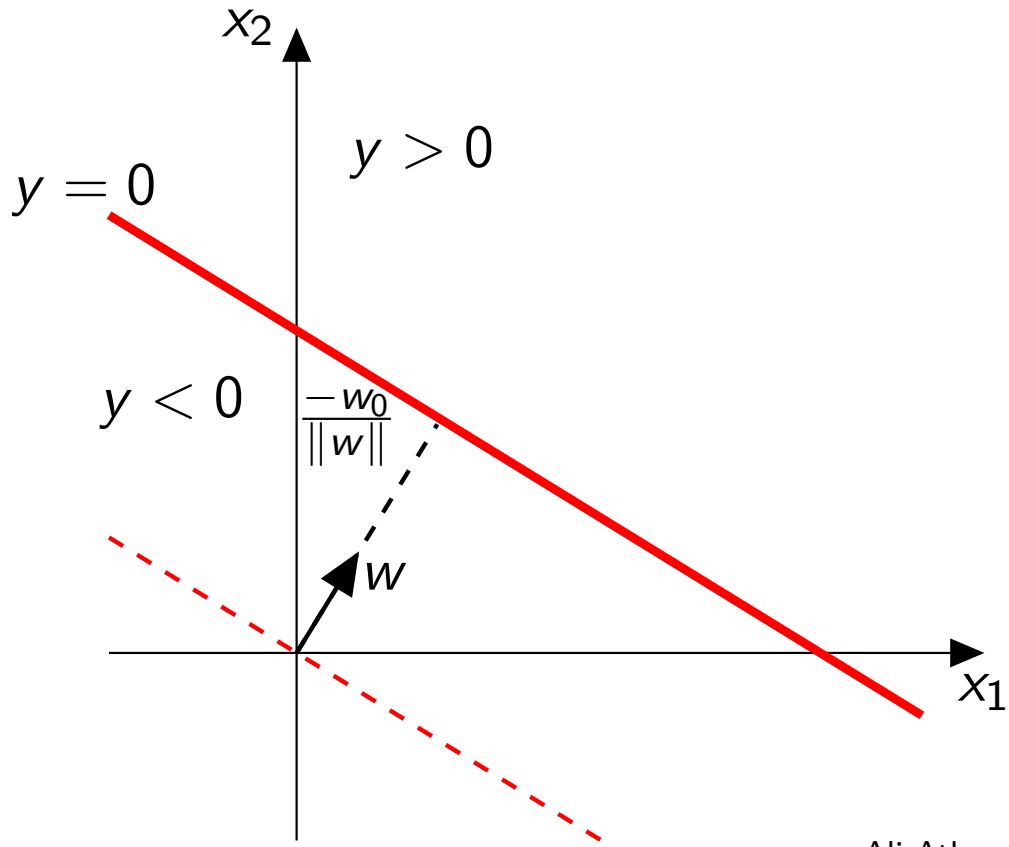
08.12.2020

Content

Content

Linear Discriminant Functions

- Two-class classification
- $y(x) = 0$ defines a hyperplane
- Normal vector w
- Offset $\frac{-w_o}{\|w\|}$



Linear Discriminant Functions (cont.)

- Notation

$$y(x) = w^T x + w_0 = \sum_{d=1}^{D} w_i x_i + w_0 = \sum_{d=0}^{D} w_i \tilde{x}_i = \tilde{w}^T \tilde{x}$$

$$x = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix} \qquad w = \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{pmatrix}$$

$$\tilde{x} = \begin{pmatrix} x_0 = 1 \\ x_1 \\ x_2 \\ \vdots \\ x_D \end{pmatrix} \qquad \tilde{w} = \begin{pmatrix} w_0 \\ w_1 \\ w_2 \\ \vdots \\ w_d \end{pmatrix}$$

Linear Discriminant Functions (cont.)

- Generalized K-Class classification

$$y_k(x) = w_k^T x + w_{k0} \qquad k \in \{1, \ldots, K\}$$

- Group the functions using vector notation

$$y(x) = \widetilde{W}^T \tilde{x} = (\tilde{w}_1, \tilde{w}_2, \ldots, \tilde{w}_K)^T \tilde{x} = \begin{pmatrix} w_{10} & w_{11} & \ldots & w_{1D} \\ w_{20} & w_{21} & \ldots & w_{2D} \\ \vdots & \vdots & \ddots & \vdots \\ w_{K0} & w_{K1} & \ldots & w_{KD} \end{pmatrix} \tilde{x}$$

- Target value

$$t = \begin{pmatrix} t_1, & t_2, & \ldots, & t_K \end{pmatrix}^T$$

Generalized Linear Discriminant

- For the entire dataset we can write

$$
\begin{aligned}
Y(\widetilde{X}) &= \widetilde{X}\widetilde{W} \\
\widetilde{W} &= \begin{pmatrix} \tilde{w}_1, & \tilde{w}_2, & \ldots, & \tilde{w}_K \end{pmatrix} && \in \mathbb{R}^{D\times K} \\
\widetilde{X} &= \begin{pmatrix} \tilde{x}_1, & \tilde{x}_2, & \ldots, & \tilde{x}_N \end{pmatrix}^T && \in \mathbb{R}^{N\times D} \\
T &= \begin{pmatrix} t_1, & t_2, & \ldots, & t_K \end{pmatrix}^T && \in \mathbb{R}^{N\times K}
\end{aligned}
$$

- Result of the comparison

$$Y(\widetilde{X}) - T = \widetilde{X}\widetilde{W} - T \in \mathbb{R}^{N\times K}$$

- Goal: Choose $\widetilde{W}$ such that this is minimal

Least Squares Classification

- Directly try to minimize the sum of squares error
- Formulation

$$E_D(\widetilde{W}) = \frac{1}{2}\sum_{k=1}^{K}\sum_{n=1}^{N}\left(y_k(x_n; w) - t_{kn}\right)^2$$

- In matrix notation

$$E_D(\widetilde{W}) = \frac{1}{2}\mathrm{Tr}\left((\widetilde{X}\widetilde{W} - T)^T(\widetilde{X}\widetilde{W} - T)\right)$$

Least Squares Classification (cont.)

- Taking the derivative yields

$$\begin{aligned}
\frac{\partial}{\partial \widetilde{W}} E_D(\widetilde{W}) &= \frac{1}{2} \frac{\partial}{\partial \widetilde{W}} \mathrm{Tr}\Big((\widetilde{X}\widetilde{W} - T)^T (\widetilde{X}\widetilde{W} - T) \Big) \\
&= \frac{1}{2} \frac{\partial}{\partial (\widetilde{X}\widetilde{W} - T)^T (\widetilde{X}\widetilde{W} - T)} \\
&\quad \mathrm{Tr}\Big((\widetilde{X}\widetilde{W} - T)^T (\widetilde{X}\widetilde{W} - T) \Big) \\
&\quad \cdot \frac{\partial}{\partial \widetilde{W}} (\widetilde{X}\widetilde{W} - T)^T (\widetilde{X}\widetilde{W} - T) \\
&= \widetilde{X}^T (\widetilde{X}\widetilde{W} - T) \stackrel{!}{=} 0
\end{aligned}$$

Least Squares Classification (cont.)

- Minimizing the least squares error

$$\frac{\partial}{\partial \widetilde{W}} E_D(\widetilde{W}) = \widetilde{X}^T(\widetilde{X}\widetilde{W} - T) \stackrel{!}{=} 0$$

$$\Leftrightarrow \quad \widetilde{X}\widetilde{W} = T$$

$$\Leftrightarrow \quad \widetilde{W} = (\widetilde{X}^T\widetilde{X})^{-1}\widetilde{X}^T T$$

$$\Leftrightarrow \quad \widetilde{W} = \widetilde{X}^+ T$$

Least Squares Classification (cont.)

- All though the training data is well separated in to two separate classes, the decision boundary is not optimal for the reason that the data points which are too far from the decision boundary tend to bend the decision boundary towards one side of the class. This happens due the high sensitivity of least squares towards outliers.

Content

Support Vector Machine - Recap

- The SVM tries to find a classifier which maximizes the margin between 2 classes.
- Up to now considered linear classifiers

$$y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + b$$

  - Formulation as a complex optimization problem
  - Find the hyperplane satisfying

$$\underset{\mathbf{w},b}{\operatorname{argmin}} \frac{1}{2}||\mathbf{W}||^2$$

  - under the constraints

$$t_n(\mathbf{w}^T\mathbf{x}_n + b) \geq 1, \forall n$$

  - based on training data points x_n and target value $t_n \in \{-1, 1\}$

Support Vector Machine - Recap

- Lagrangian primal form

$$\mathcal{L}_p = \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} \alpha_n \{t_n(\mathbf{w}^T\mathbf{x}_n + b) - 1\}$$

$$= \frac{1}{2}||\mathbf{w}||^2 - \sum_{n=1}^{N} \alpha_n \{t_n y(\mathbf{x}_n) - 1\}$$

- The solution of $\mathcal{L}_p$ needs to fulfill the KKT conditions
  - Necessary and sufficient conditions

$$\alpha_n \geq 0 \qquad \lambda \geq 0$$

$$t_n y(\mathbf{x}_n) - 1 \geq 0 \qquad f(x) \geq 0$$

$$\alpha_n \{t_n y(\mathbf{x}_n) - 1\} = 0 \qquad \lambda f(x) = 0$$

Support Vector Machine - Recap

- Solution for hyperplane
  - computed as linear combination of the training examples

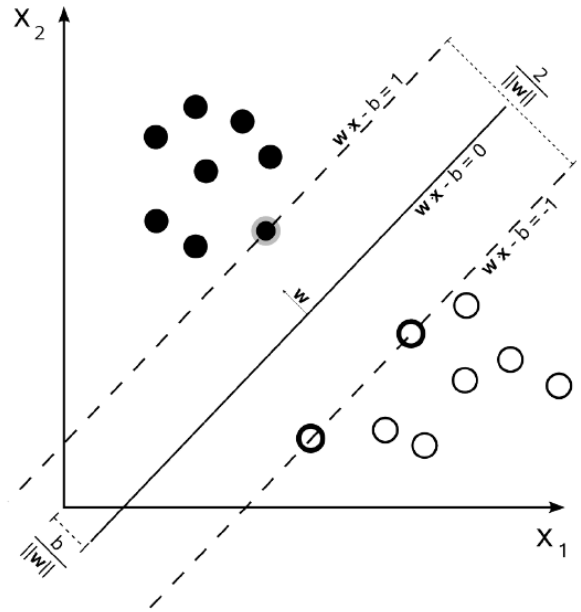
$$\mathbf{w} = \sum_{n-1}^{N} \alpha_n t_n \mathbf{x}_n$$

  - Suppose solution: $\alpha_n \neq 0$ only for some points, the support vectors
    - Only the SVs actually influence the decision Boundary!
  - Compute b by averaging over all support vectors:

$$b = \frac{1}{N_S} \sum_{n \in \mathcal{S}} \left(t_n - \sum_{m \in \mathcal{S}} \alpha_m t_m \mathbf{x}_m^T \mathbf{x}_n \right)$$

Support Vector Machine - Recap

- The training points for which $\alpha_n \geq 0$ are called support vectors



- Graphical interpretation:
  - The support vectors are the points on the margin.
  - They define the margin and thus the hyperplane.
  - All other points can be discarded.

Support Vector Machine - Recap

- Primal form has time complexity of $\mathcal{O}(D^3)$ in D dimensions
- Dual form can be obtained by substituting weight vectors

$$\mathbf{w} = \sum_{n=1}^{N} \alpha_n t_n \mathbf{x}_n$$

$$\mathcal{L}_d(\alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m t_n t_m (\mathbf{x}_m^T \mathbf{x}_n)$$

- Under the conditions

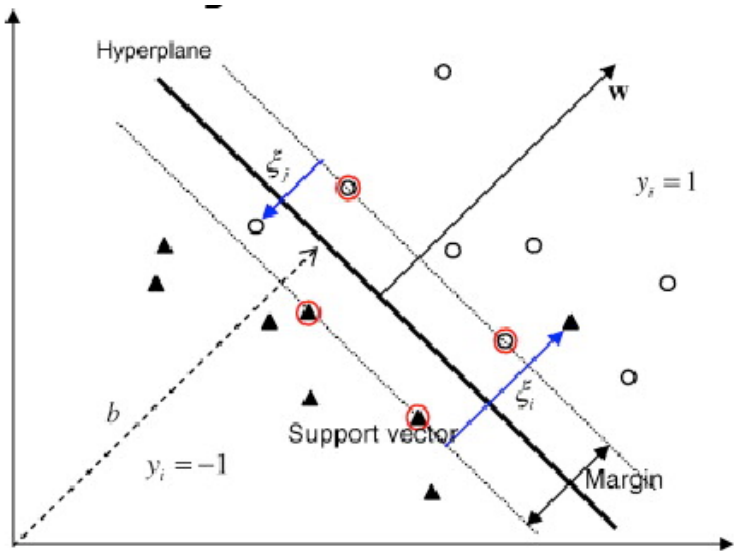
$$\alpha_n \geq 0 \quad \forall n$$

$$\sum_{n=1}^{N} \alpha_n t_n = 0$$

- time complexity between $\mathcal{O}(N)$ and $\mathcal{O}(N^2)$

Support Vector Machine - Recap

- In case of overlapping class distribution, we allow some points to be miss-classified but with linear penalty that increases with the distance from the boundary.



Support Vector Machine - Recap

- Slack variables
  - $\xi \geq 0$ for each data point.
- Interpretation
  - $\xi_n = |t_n - y(\mathbf{x}_n)|$
  - $\xi_n = 0$ for points correctly classified or on the margin
  - $0 \leq \xi_n \leq 1$ points lie inside the margin on the correct side of the decision boundary
  - $y(x_n) = 0$ and $\xi = 1$ for the data point on the decision boundary
  - $\xi_n > 1$ for misclassified points
- The exact classification constraints are then replaced with

$$t_n y(\mathbf{x}) \geq 1 - \xi_n$$

Support Vector Machine - Recap

- Goal is to maximize margin while softly penalizing points that lie on the wrong side of the margin boundary.
- Therefore we minimize :

$$C \sum_{n=1}^{N} \xi_n + \frac{1}{2}||\mathbf{w}||^2$$

- Where the parameter C controls the trade-of between the slack variable penalty and the margin.
- In the limit of $C \to \infty$ we will recover the earlier, fully linearly separable case (non overlapping class distribution).

Support Vector Machine - Recap

- The corresponding Lagrangian multiplier can be given:

$$\mathcal{L}(\mathbf{w}, b, \alpha, \xi) = \frac{1}{2}||\mathbf{w}||^2 + C\sum_{n=1}^{N}\xi_n - \sum_{n=1}^{N}\alpha_n\left\{t_n y(\mathbf{x}_n) - 1 + \xi_n\right\} - \sum_{n=1}^{N}\mu_n\xi_n$$

- Where α_n and μ_n are Lagrangian multipliers
- The corresponding set of KKT conditions are given by:

$$\begin{aligned} \alpha_n &\geq 0 & \mu_n &\geq 0 \\ t_n y(\mathbf{x}_n) - 1 + \xi_n &\geq 0 & \xi_n &\geq 0 \\ \alpha_n\left\{t_n y(\mathbf{x}_n) - 1 + \xi_n\right\} &= 0 & \mu_n\xi_n &= 0 \end{aligned}$$

- We can optimize $\mathbf{w}, b, \{\xi_n \mid n \in \{1, \ldots, N\}\}$ as follows:

$$\frac{\partial L}{\partial \mathbf{w}} = 0 \quad \Rightarrow \quad \mathbf{w} = \sum_{n=1}^{N}\alpha_n t_n x_n$$

$$\frac{\partial L}{\partial b} = 0 \quad \Rightarrow \quad 0 = \sum_{n=1}^{N}\alpha_n t_n$$

$$\frac{\partial L}{\partial \xi_n} = 0 \quad \Rightarrow \quad a_n = C - \mu_n$$

Support Vector Machine - Recap

- New SVM Dual: Maximize

$$\mathcal{L}_d(\alpha) = \sum_{n=1}^{N} \alpha_n - \frac{1}{2} \sum_{n=1}^{N} \sum_{m=1}^{N} \alpha_n \alpha_m t_n t_m (\mathbf{x}_m^T \mathbf{x}_n)$$

- Under conditions

$$0 \leq \alpha_n \leq C$$

$$\sum_{n=1}^{N} \alpha_n t_n = 0$$

- This is a quadratic programming problem

Support Vector Machine - Recap

- Using one of the KKT condition and result from partial derivation of Lagrange function we can derive condition for data points to be either support vector or slacks as below,
- We know that

$$\frac{\partial L}{\partial \xi_n} = 0 \qquad \Rightarrow \qquad \alpha_n = C - \mu_n$$

and

$$\mu_n \geq 0$$
$$\xi_n \geq 0$$
$$\mu_n \xi_n = 0$$

- For support vectors and slacks $0 \leq \alpha_n \leq \mathcal{C}$
- Moreover for slacks $\xi_n \geq 0$ which implies $\mu_n = 0$
- From the partial derivation shown above $\rightarrow \alpha_n = \mathcal{C}$