CS-567 Machine Learning

Nazar Khan

PUCIT

Lecture 14
Linear Classification

Classification

- In the previous topic, regression, the goal was to predict *continuous* target variable(s) t given input variables vector $\mathbf{x}$.
- In *classification*, the goal is to predict *discrete* target variable(s) t given input variables vector $\mathbf{x}$.
- Input space is divided into *decision regions*.
- Boundaries between regions are called *decision boundaries/surfaces*.
- Training corresponds to finding optimal decision boundaries given training data $\{(\mathbf{x}_1, t_1), \ldots, (\mathbf{x}_N, t_N)\}$.

Classification

- Assign $\mathbf{x}$ to 1-of-K discrete classes C_k.
- Most commonly, the classes are distinct. That is, $\mathbf{x}$ is assigned to one and only one class.
- Convenient coding schemes for targets t are
  - $0/1$ coding for binary classification.
  - 1-of-K coding for multi-class classification. Example, for $\mathbf{x}$ belonging to class 3, the $K \times 1$ target vector will be coded as $\mathbf{t} = (0, 0, 1, 0, \dots, 0)^T$.

Linear Classification

- Like regression, the simplest classification model is *linear classification*.
  - This means that the decision surfaces are linear functions of $\mathbf{x}$, for example $y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T\mathbf{x} + w_0 = 0$.
  - That is, a linear decision surface is a $D-1$ dimensional hyperplane in D-dimensional space.
- Data in which classes can be *separated exactly* by *linear decision surfaces* is called *linearly separable*.

Linear Classification

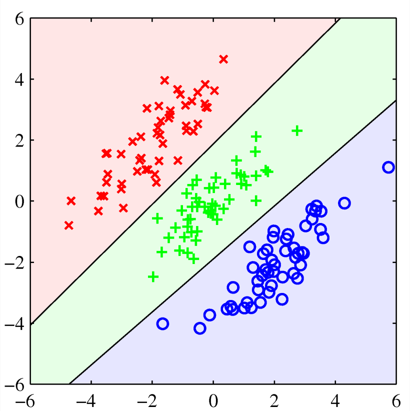

Figure: Linearly separable data and corresponding linear decision boundaries.

3 Approaches for Solving Classification (Decision) Problems

1. **Generative**: Infer posterior $p(\mathcal{C}_k|\mathbf{x})$
    - either by inferring $p(\mathbf{x}|\mathcal{C}_k)$ and $p(\mathbf{x})$ and using Bayes' theorem,
    - or by inferring $p(\mathbf{x}, \mathcal{C}_k)$ and marginalizing.
    - Called generative because $p(\mathbf{x}|\mathcal{C}_k)$ and/or $p(\mathbf{x}, \mathcal{C}_k)$ allow us to generate new $\mathbf{x}$'s.
2. **Discriminative**: Model the posterior $p(\mathcal{C}_k|\mathbf{x})$ directly.
    - If decision depends on posterior, then no need to model the joint distribution.
3. **Discriminant Function**: Just learn a discriminant function that maps $\mathbf{x}$ directly to a class label.
    - f($\mathbf{x}$)=0 for class $\mathcal{C}_1$.
    - f($\mathbf{x}$)=1 for class $\mathcal{C}_2$.
    - No probabilities

Linear Classification
Generalized Linear Model

- The simplest linear regression model computes continuous outputs $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$.
- By passing these continuous outputs through a non-linear function $f(\cdot)$, we can obtain discrete class labels.

$$y(\mathbf{x}) = f(\mathbf{w}^T\mathbf{x} + w_0)$$

- This is known as a *generalised linear model* and $f(\cdot)$ is known as the *activation function*.
  - Decision surfaces correspond to all inputs $\mathbf{x}$ where $y(\mathbf{x}) = \text{const}$. This is equivalent to the condition $\mathbf{w}^T\mathbf{x} + w_0 = \text{const}$.
  - Therefore, decision surfaces are linear functions of the input $\mathbf{x}$, even if $f(\cdot)$ is non-linear.
- As before, we can replace $\mathbf{x}$ by a non-linear transformation $\phi(\mathbf{x})$ and learn non-linear boundaries in $\mathbf{x}$-space by learning linear boundaries in ϕ-space.

Linear Discriminant Functions

Two class case

- The simplest linear discriminant function is given by $y(\mathbf{x}) = \mathbf{w}^T\mathbf{x} + w_0$ where $\mathbf{w}$ is called the *weight vector* and w_0 is called the *bias*.
- Classification is performed via the non-linear step

$$\text{class}(\mathbf{x}) = \begin{cases} \mathcal{C}_1 & \text{if } y(\mathbf{x}) \geq 0 \\ \mathcal{C}_2 & \text{if } y(\mathbf{x}) < 0 \end{cases}$$

- We can view $-w_0$ as a *threshold*.
- Weight vector $\mathbf{w}$ is always orthogonal to the decision surface.
  - Proof: For *any* two points $\mathbf{x}_A$ and $\mathbf{x}_B$ on the surface, $y(\mathbf{x}_A) = y(\mathbf{x}_B) = 0 \Rightarrow \mathbf{w}^T(\mathbf{x}_A - \mathbf{x}_B) = 0$. Since vector $\mathbf{x}_A - \mathbf{x}_B$ is along the surface, $\mathbf{w}$ must be orthogonal.

Linear Discriminant Functions

Two class case

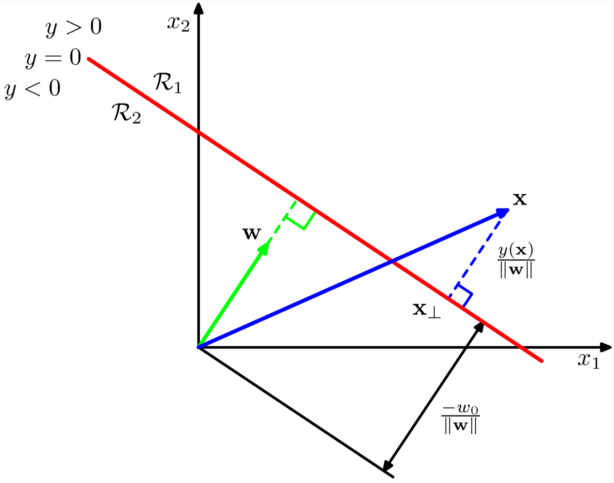

Figure: Geometry of linear discriminant function in $\mathbb{R}^2$.

Linear Discriminant Functions

Two class case

- Normal distance of any point $\mathbf{x}$ from decision boundary can be computed as $d = \frac{y(\mathbf{x})}{||\mathbf{w}||}$.
  - Proof:

$$\mathbf{x} = \mathbf{x}_\perp + d\frac{\mathbf{w}}{||\mathbf{w}||}$$

$$\Rightarrow \underbrace{\mathbf{w}^T\mathbf{x} + w_0}_{y(\mathbf{x})} = \underbrace{\mathbf{w}^T\mathbf{x}_\perp + w_0}_{y(\mathbf{x}_\perp)=0} + d\underbrace{\mathbf{w}^T\frac{\mathbf{w}}{||\mathbf{w}||}}_{||\mathbf{w}||}$$

$$\Rightarrow d = \frac{y(\mathbf{x})}{||\mathbf{w}||}$$

- Normal distance to boundary from origin $(\mathbf{x} = \mathbf{0})$ is $\frac{w_0}{||\mathbf{w}||}$.

Linear Discriminant Functions

- For notational convenience, bias can be included as a component of the weight vector via

$$\begin{aligned}\tilde{\mathbf{w}} &= (w_0, \mathbf{w}) \\ \tilde{\mathbf{x}} &= (1, \mathbf{x}) \\ y(\mathbf{x}) &= \tilde{\mathbf{w}}^T \tilde{\mathbf{x}}\end{aligned}$$

Linear Discriminant Functions
Multiclass case

- For K class classification with $K > 2$, we have 3 options
  1. Learn $K - 1$ *one-vs-rest* binary classifiers.
  2. Learn $K(K-1)/2$ *one-vs-one* binary classifiers for every possible pair of classes. Each point can be classified based on majority vote among the discriminant functions.
  3. Learn K discriminant functions $y_1, \ldots, y_K$ and then $\text{class}(\mathbf{x}) = \arg\max_k y_k(\mathbf{x})$.
- Options 1 and 2 lead to ambiguous classification regions.

Linear Discriminant Functions
Multiclass Ambiguity

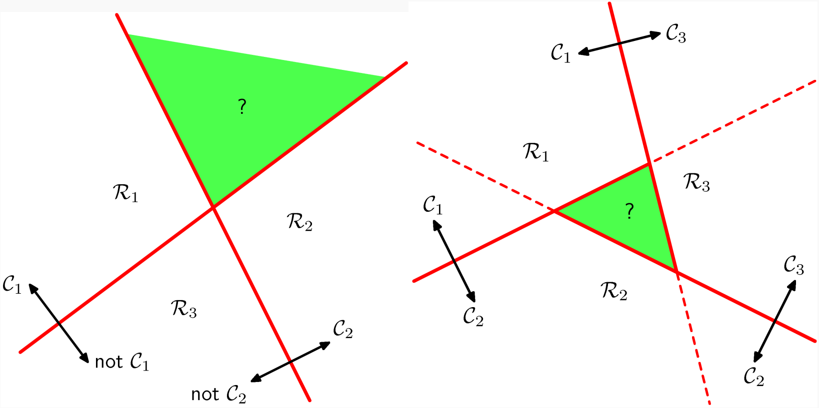

Figure: Ambiguity of multiclass classification using two-class linear discriminant functions.

Linear Discriminant Functions

Multiclass case

- We can write the K-class discriminant function as

$$\mathbf{y}(\mathbf{x}) = \tilde{\mathbf{W}}^T \tilde{\mathbf{x}}$$

- For learning, we can write the error function as

$$\begin{aligned} E(\tilde{\mathbf{W}}) &= \frac{1}{2}\sum_{n=1}^{N} ||\mathbf{y}(\mathbf{x}_n) - \mathbf{t}_n||^2 \\ &= \frac{1}{2}\sum_{n=1}^{N} (\tilde{\mathbf{W}}^T \tilde{\mathbf{x}}_n - \mathbf{t}_n)^T (\tilde{\mathbf{W}}^T \tilde{\mathbf{x}}_n - \mathbf{t}_n) \end{aligned}$$

- The optimal discriminant function parameters can be computed as $\tilde{\mathbf{W}}^* = \tilde{\mathbf{X}}^\dagger \mathbf{T}$ where $\tilde{\mathbf{X}}^\dagger$ is the pseudo-inverse of the design matrix $\tilde{\mathbf{X}}$ and $\mathbf{T}$ is the matrix of target vectors.
- As before, we can also work in ϕ-space where we will use the corresponding $\tilde{\mathbf{\Phi}}$ as the design matrix.

Linear Discriminant Functions
Least Squares Solution

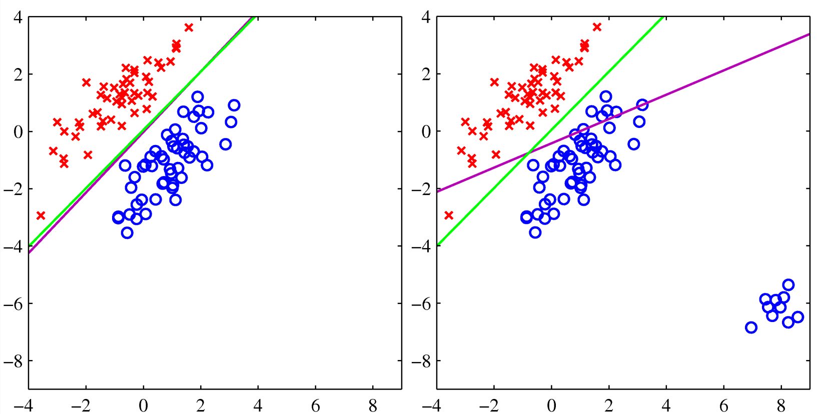

Figure: Least squares solution is sensitive to outliers.

Fisher's Linear Discriminant

Two class case

- Project all data onto a single vector $\mathbf{w}$.
- Classify by thresholding projected coefficents.
- Optimal vector is one which
  - maximises between-class distance, and
  - minimises within-class distance.

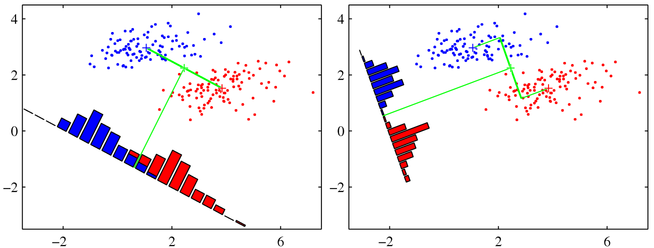

Figure: Fisher's linear discriminant. Classify by thresholding projections onto a vector $\mathbf{w}$ that maximises inter-class distance and minimises intra-class distances.

Fisher's Linear Discriminant

Two class case

- Let $\mathbf{m}_k = \frac{\sum_{n \in \mathcal{C}_k} \mathbf{x}_n}{N_k}$ be the mean vector of points belonging to class $\mathcal{C}_k$.
- Projection of this mean is then $m_k = \mathbf{w}^T \mathbf{m}_k$.
- Variance around projected mean can be written as $s_k^2 = \sum_{n \in \mathcal{C}_k} (\mathbf{w}^T \mathbf{x}_n - \mathbf{w}^T \mathbf{m}_k)^2$.
- Suitability of any projection direction $\mathbf{w}$ can then be written as

$$
\begin{aligned}
J(\mathbf{w}) &= \frac{\text{Inter-class variance}}{\text{Intra-class variance}} \\
&= \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \\
&= \frac{(\mathbf{w}^T \mathbf{m}_2 - \mathbf{w}^T \mathbf{m}_1)^2}{\sum_{n \in \mathcal{C}_1} (\mathbf{w}^T \mathbf{x}_n - \mathbf{w}^T \mathbf{m}_1)^2 + \sum_{n \in \mathcal{C}_2} (\mathbf{w}^T \mathbf{x}_n - \mathbf{w}^T \mathbf{m}_2)^2}
\end{aligned}
$$

Fisher's Linear Discriminant

Two class case

$$J(\mathbf{w}) = \frac{(\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1))(\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1))^T}{\sum_{k=1}^{2}\sum_{n\in\mathcal{C}_k}(\mathbf{w}^T(\mathbf{x}_n - \mathbf{m}_k))^2}$$

$$= \frac{\mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w}}{\mathbf{w}^T\left(\sum_{k=1}^{2}\sum_{n\in\mathcal{C}_k}(\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^T\right)\mathbf{w}}$$

$$= \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} \qquad (\mathbf{S}_B \text{ and } \mathbf{S}_W \text{ are symmetric due to outer-products})$$

$$\nabla_{\mathbf{w}}E(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{S}_W\mathbf{w}) - \mathbf{w}^T\mathbf{S}_W\mathbf{w}\nabla_{\mathbf{w}}(\mathbf{w}^T\mathbf{S}_B\mathbf{w})}{(\mathbf{w}^T\mathbf{S}_W\mathbf{w})^2} \quad (\because \text{quotient rule})$$

$$= \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}(2\mathbf{S}_W\mathbf{w}) - \mathbf{w}^T\mathbf{S}_W\mathbf{w}(2\mathbf{S}_B\mathbf{w})}{(\mathbf{w}^T\mathbf{S}_B\mathbf{w})^2} \quad \left(\because \nabla_{\mathbf{v}}(\mathbf{v}^T\mathbf{M}\mathbf{v}) = (\mathbf{M} + \mathbf{M}^T)\mathbf{v}\right)$$

Fisher's Linear Discriminant
Two class case

- Objective J can be maximized by equating gradient to the $\mathbf{0}$ vector

$$\mathbf{w}^T\mathbf{S}_B\mathbf{w}(\mathbf{S}_W\mathbf{w}) = \mathbf{w}^T\mathbf{S}_W\mathbf{w}(\mathbf{S}_B\mathbf{w})$$

- Since we only care about the direction of projection, we can drop the scalar factors to get

$$\begin{aligned}
\mathbf{S}_W\mathbf{w} &= \mathbf{S}_B\mathbf{w} \\
\mathbf{S}_W\mathbf{w} &= (\mathbf{m}_2 - \mathbf{m}_1)\underbrace{(\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w}}_{\text{scalar}} \\
\mathbf{S}_W\mathbf{w} &\propto (\mathbf{m}_2 - \mathbf{m}_1) \\
\mathbf{w} &\propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)
\end{aligned}$$

Perceptron Algorithm

Two-class Classification

- Target t_n is taken to be either $+1$ or -1.
- A perceptron classifies its input via the non-linear step function

$$y(\phi) = \begin{cases} 1 & \text{if } \mathbf{w}^T\phi_n \geq 0 \\ -1 & \text{if } \mathbf{w}^T\phi_n < 0 \end{cases}$$

- Extremely simplified model of biological neuron.
- *Perceptron criterion*: $\mathbf{w}^T\phi_n t_n > 0$ for correctly classified point.
- Error can be defined on the set $\mathcal{M}(\mathbf{w})$ of misclassified points.

$$E(\mathbf{w}) = \sum_{n\in\mathcal{M}(w)} -\mathbf{w}^T\phi_n t_n$$

- Optimal $\mathbf{w}$ can be learned via gradient descent.
- For linearly separable data, perceptron learning is guaranteed to find the decision boundary in finite iterations.

Gradient Descent

- Gradient is the direction (in input space) of maximum rate of increase of a function.
- To minimize, move in negative gradient direction.

$$\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \eta \nabla_{\mathbf{w}E(\mathbf{w})}$$

- Also known as *gradient descent*.
- Local versus global minima.
- Learning rate η should be decayed to *avoid osscillation* and to *converge* to a local minimum.
- Different types of gradient descent:
  - Batch ($\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \eta \nabla_{\mathbf{w}} E$)
  - Sequential ($\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \eta \nabla_{\mathbf{w}} E_n$)
  - Stochastic (same as sequential but n is chosen randomly).
  - Mini-batches ($\mathbf{w}^{\text{new}} = \mathbf{w}^{\text{old}} - \eta \nabla_{\mathbf{w}} E_{\mathcal{B}}$)
- Most common variations are stochastic gradient descent (SGD) and SGD using mini-batches.