Classification

Jiayu Zhou

[1] Department of Computer Science and Engineering
Michigan State University
East Lansing, MI USA

Table of contents

Problem Statement

Given an input vector $\mathbf{x}$ and a set of training patterns $\{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$, the goal is to assign it to one of K discrete classes $\mathcal{C}_k$ where $k = 1, \ldots, K$

Different Approaches to Classification

- Construct a **discriminant function** which assigns each vector $\mathbf{x}$ to a specific class
- Model the conditional probability distribution $p(\mathcal{C}_k|\mathbf{x})$ in an inference stage, and use this distribution to make optimal decisions
- Two methods to model $p(\mathcal{C}_k|\mathbf{x})$
  - Discriminant model
    Example: Representing $p(\mathcal{C}_k|\mathbf{x})$ as parametric models and then optimizing the parameters using a training set
  - Generative method
    Model the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ and prior probabilities $p(\mathcal{C}_k)$, and compute $p(\mathcal{C}_k|\mathbf{x})$ using Bayes theorem

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

Basic Definitions

- Discriminant
  A **discriminant** is a function that takes an input vector $\mathbf{x}$ and assign it to one of K classes, denoted as $\mathcal{C}_k$
- Linear discriminant
  In linear discriminant, the decision boundaries (or decision surfaces) are hyperplanes in the input space
- Linear separable
  Data sets whose classes can be separated exactly by linear decision surfaces are said to be **linear separable**
- Generalized linear models

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

$f(.)$ is called **activation function**, and it may be **nonlinear**.

Discriminant Functions - Two Classes

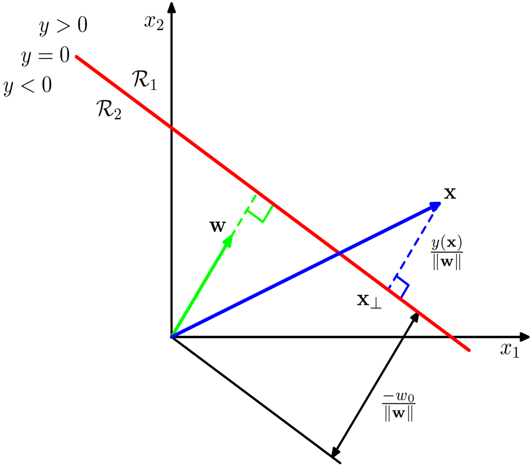
- Formulation

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

  - $\mathbf{w}$: weight vector
  - w_0: bias
  - $-w_0$: threshold
- Decision Rule
  An input vector $\mathbf{x}$ is assigned to class $\mathcal{C}_1$ if $y(\mathbf{x}) \geq 0$ and to class $\mathcal{C}_2$ otherwise
- The geometric property
  - $\mathbf{w}$: the direction of decision surface
  - w_0: the location of decision surface
  - The signed distance r of point $\mathbf{x}$ from the decision surface

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

An Example of Geometry



Discriminant Functions - Multiple Classes

- Possible Methods
  1. One-versus-the-rest
     Use $K-1$ two-class classifiers
  2. One-versus-one
     Use $K(K-1)/2$ two-class classifiers
- Define a single K-class discriminant comprising K linear functions of the form

$$y_k(\mathbf{x}) = \mathbf{w}_k^T \mathbf{x} + w_{k0}$$

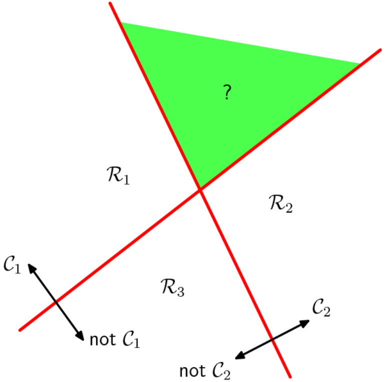
- Decision Rule
  Assigning a point $\mathbf{x}$ to class $\mathcal{C}_k$ if $y_k(\mathbf{x}) > y_j(\mathbf{x})$ for all $j \neq k$
- The decision boundary between class $\mathcal{C}_k$ and class $\mathcal{C}_j$

$$y_k(\mathbf{x}) = y_j(\mathbf{x})$$

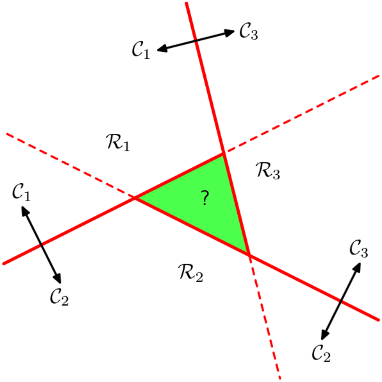
This boundary is hyperplane defined by:

$$(\mathbf{w}_k - \mathbf{w}_j)^T \mathbf{x} + (w_{k0} - w_{j0}) = 0$$

The Difficulty of One-versus-the-rest and One-versus-one



The dilemma of One-versus-the-rest



The dilemma of One-versus-one

Properties of Multi-class Classifier

The decision regions of the discriminant given by $y_k(\mathbf{x}) = \mathbf{w}_k^T\mathbf{x} + w_{k0}$ are convex.

Proof

Any point $\hat{\mathbf{x}}$ that lies on the line connecting $\mathbf{x}_A$ and $\mathbf{x}_B$ can be expressed in the form

$$\hat{\mathbf{x}} = \lambda\mathbf{x}_A + (1-\lambda)\mathbf{x}_B, \text{where} 0 \leq \lambda \leq 1$$

If $\mathbf{x}_A$ and $\mathbf{x}_B$ lies in $\mathcal{R}_k$, then

$$y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A) \text{ for all } j \neq k$$

$$y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B) \text{ for all } j \neq k$$

From the linearity of the discriminant functions, it follows that

$$y_k(\hat{\mathbf{x}}) > y_j(\hat{\mathbf{x}}) \text{ for all } j \neq k$$

It means that $\hat{\mathbf{x}} \in \mathcal{R}_k$.

Least Squares for Classification

- Each 1-vs-rest classifier $\mathcal{C}_k$ is described by a linear model so that

$$y_k(\mathbf{x}) = \mathbf{w}_k^T \mathbf{x} + w_{k0}$$

  where $k = 1, \ldots, K-1$

- Or equivalently

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

  where $\tilde{\mathbf{W}} = [\tilde{\mathbf{w}}_1, \ldots, \tilde{\mathbf{w}}_{K-1}], \tilde{\mathbf{w}}_k = (w_{k0}, \mathbf{w}_k^T)^T, \tilde{\mathbf{x}} = (1, \mathbf{x}^T)^T$

- By defining the target matrix T, the sum-of-squares error function is

$$\begin{aligned} E_D(\tilde{\mathbf{W}}) &= \frac{1}{2}\|\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T}\|_F^2 \\ &= \frac{1}{2}\mathrm{Tr}\{(\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T})^T(\tilde{\mathbf{X}}\tilde{\mathbf{W}} - \mathbf{T})\} \end{aligned}$$

Least Squares for Classification (cont.)

- The solution is

$$\tilde{\mathbf{W}} = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\mathbf{T} = \tilde{\mathbf{X}}^\dagger\mathbf{T}$$

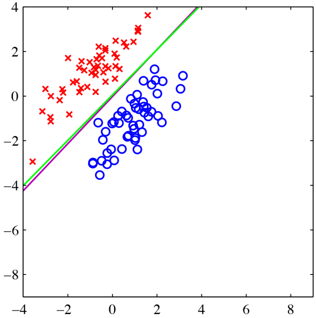
- Then the discriminant is

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

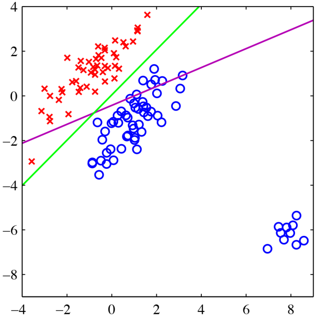
Drawbacks of Least Square

- Least-squares solutions **lack robustness** to outliers
  The sum-of-squares error function penalizes predictions that are "too correct" in that they lie a long way on the correct side of the decision boundary
- Sometimes least squares have **poor performance**
  Least squares corresponds to maximum likelihood under the assumption of a Gaussian conditional distribution, whereas binary target vectors clearly have a distribution that is far from Gaussian

Least-squares Solutions Lack Robustness to Outliers

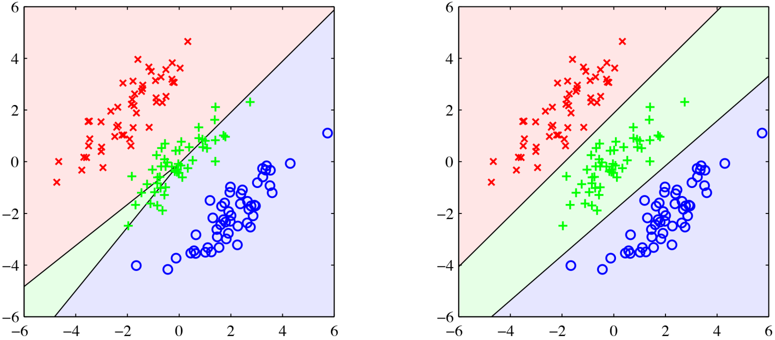


Original Boundary. Magenta curve is least squares and green curve is logistic regression



New Boundary

Poor Performance of Least-squares

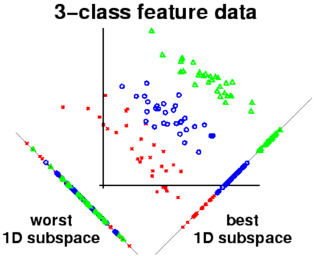


Least Squares

Logistic Regression

Fisher's Linear Discriminant

- Basic Idea
  Project the data in the original D-dimensional space into lower space. In the projection, we expect to **maximize the between-class distance** and **minimize within-class distance**



- For simplicity, we First consider the projection to 1-dimensional space for two-class problem.

Formal Formulation

- N_1: number of points in class $\mathcal{C}_1$
  N_2: number of points in class $\mathcal{C}_2$
- The mean vectors of each class in original space

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \in \mathcal{C}_1} \mathbf{x}_n \tag{1}$$

$$\mathbf{m}_2 = \frac{1}{N_2} \sum_{n \in \mathcal{C}_2} \mathbf{x}_n \tag{2}$$

- The linear projection is

$$y = \mathbf{w}^T \mathbf{x}$$

Fisher's Linear Discriminant

- The distance between classes is measured by the distance of means in the projected 1-dimensional space

$$m_2 - m_1 = \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)$$

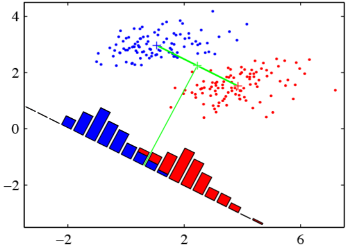
- The measure of within-class distance is measured by the variance within each class in the projected 1-dimensional space

$$s_k^2 = \sum_{n \in \mathcal{C}_k} (y_n - m_k)^2$$

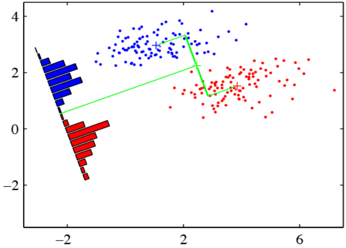
- Fisher criterion

$$\max J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

Example of Linear Projection



Projection onto the line jointing class means



LDA Projection

Fishers Linear Discriminant

- Reformulation

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

  - Between-class covariance matrix $\mathbf{S}_B$

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T$$

  - Within-class covariance matrix $\mathbf{S}_W$

$$\mathbf{S}_W = \sum_{n \in \mathcal{C}_1} (\mathbf{x}_n - \mathbf{x}_1)(\mathbf{x}_n - \mathbf{x}_1)^T + \sum_{n \in \mathcal{C}_2} (\mathbf{x}_n - \mathbf{x}_2)(\mathbf{x}_n - \mathbf{x}_2)^T$$

- Set gradient to zero

$$\frac{\partial J}{\partial \mathbf{w}} = 0 \Leftrightarrow (\mathbf{w}^T \mathbf{S}_B \mathbf{w})\mathbf{S}_W \mathbf{w} = (\mathbf{w}^T \mathbf{S}_W \mathbf{w})\mathbf{S}_B \mathbf{w} \Rightarrow \mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)$$

Least Squares Versus Fishers Linear Discriminant

- For two-class problems, Fishers Linear Discriminant can be considered as a special case of least squares.
- For multi-class problems, Fishers Linear Discriminant can also be considered a special case of least squares by constructing a special indicator matrix (Ye, ICML 2007).

Two-class Problem I

- Key Idea: define a special target variable for different classes in least squares
- Consider a special least squares with the following target variable

$$t_n = \begin{cases} \frac{N}{N_1} & \text{if } n \in \mathcal{C}_1 \\ -\frac{N}{N_2} & \text{if } n \in \mathcal{C}_2 \end{cases}$$

- Properties of the target variable:

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

$$\sum_{n=1}^{N} t_n x_n = N(\mathbf{m}_1 - \mathbf{m}_2)$$

Two-class Problem II

- The corresponding sum-of-squares error function for the target variable is

$$\min E = \frac{1}{2}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + w_0 - t_n)^2$$

- Setting the derivatives of E with respect to w_0 and $\mathbf{w}$ to 0, we have

$$\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + w_0 - t_n) = 0 \Leftrightarrow w_0 = -\mathbf{w}^T\mathbf{m}$$

$$\text{where } \mathbf{m} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n = \frac{1}{N}(N_1\mathbf{m}_1 + N_2\mathbf{m}_2)$$

$$\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{x}_n + w_0 - t_n)\mathbf{x}_n = 0 \Leftrightarrow (\mathbf{S}_W + \frac{N_1N_2}{N}\mathbf{S}_B)\mathbf{w} = N(\mathbf{m}_1 - \mathbf{m}_2)$$

Two-class Problem III

$$
\begin{aligned}
(\mathbf{S}_W + \frac{N_1 N_2}{N}\mathbf{S}_B)\mathbf{w} &= N(\mathbf{m}_1 - \mathbf{m}_2) \\
\Leftrightarrow \mathbf{S}_W\mathbf{w} &= N(\mathbf{m}_1 - \mathbf{m}_2) - \frac{N_1 N_2}{N}\mathbf{S}_B\mathbf{w}
\end{aligned}
$$

Note that

$$
\mathbf{S}_B\mathbf{w} = (\mathbf{m}_2 - \mathbf{m}_1)\left((\mathbf{m}_2 - \mathbf{m}_1)^T\mathbf{w}\right) = s(\mathbf{m}_2 - \mathbf{m}_1), \text{ where } s \in \mathbb{R}
$$

Therefore

$$
\begin{aligned}
\mathbf{S}_W\mathbf{w} &= s'(\mathbf{m}_2 - \mathbf{m}_1), \text{ where } s' \in \mathbb{R} \\
&\Rightarrow \mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1)
\end{aligned}
$$

This result is the same as Fishers Linear Discriminant

Multi-class LDA I

- Suppose the class number is K, we consider project the data in the original D-dimensional space ($D > K$) data space into D'-dimensional space, where $D' > 1$

$$\mathbf{y} = \mathbf{W}^T\mathbf{x}, \text{ where } \mathbf{W} = [\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_{D'}]$$

- The within class covariance $\mathbf{S}_W$

$$\mathbf{S}_W = \sum_{k=1}^{K} \mathbf{S}_k$$

where

$$\mathbf{S}_k = \sum_{n \in \mathcal{C}_k} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^T$$

$$\mathbf{m}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{x}_n$$

Multi-class LDA II

- The total covariance matrix $\mathbf{S}_T$

$$\mathbf{S}_T = \sum_{n=1}^{N} (\mathbf{x}_n - \mathbf{m}_k)(\mathbf{x}_n - \mathbf{m}_k)^T$$

$$\mathbf{m} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{x}_n$$

- The between class covariance $\mathbf{S}_B$

$$\mathbf{S}_B = \sum_{k=1}^{K} N_k (\mathbf{m}_k - \mathbf{m})(\mathbf{m}_k - \mathbf{m})^T$$

- From these definitions we can show that

$$\mathbf{S}_T = \mathbf{S}_B + \mathbf{S}_W$$

Multi-class LDA III

- In the projected space, we can define similar structures

$$\mathbf{s}_W = \sum_{k=1}^{K} \sum_{n \in \mathcal{C}_k} (\mathbf{y}_n - \boldsymbol{\mu}_k)(\mathbf{y}_n - \boldsymbol{\mu}_k)^T$$

$$\mathbf{s}_B = \sum_{k=1}^{K} N_k (\boldsymbol{\mu}_k - \boldsymbol{\mu})(\boldsymbol{\mu}_k - \boldsymbol{\mu})^T$$

where

$$\boldsymbol{\mu} = \frac{1}{N} \sum_{n=1}^{N} \mathbf{y}_n$$

$$\boldsymbol{\mu}_k = \frac{1}{N_k} \sum_{n \in \mathcal{C}_k} \mathbf{y}_n$$

Multi-class LDA IV

- Many objective functions can be chosen in the lower space.
- One common choice is

$$\max J(\mathbf{W}) = \mathsf{Tr}\{\mathbf{s}_W^{-1}\mathbf{s}_B\}$$
$$\Leftrightarrow \max J(\mathbf{W}) = \mathsf{Tr}\{(\mathbf{W}\mathbf{S}_W\mathbf{W}^T)^{-1}(\mathbf{W}\mathbf{S}_B\mathbf{W}^T)\}$$

- In fact, $\mathbf{W}$ is given by the D' eigenvectors of $\mathbf{S}_W^{-1}\mathbf{S}_B$ corresponding to the D' largest eigenvalues.
- $\mathbf{S}_B$ is composed of the sum of K matrices, each of which is an outer product of two vectors and therefore of rank 1. In addition, only $(K-1)$ of these matrices are independent. Thus, $\mathbf{S}_B$ has rank at most equal to $(K-1)$ and so there are at most $(K-1)$ nonzero eigenvalues. In practice, we commonly set $D' = K - 1$.

Perceptron

- The simple yet powerful algorithm we studied in the last lecture.

Generative Model Vs. Discriminative Mode

- Probabilistic Generative Model
  Model the class-conditional densities $p(\mathbf{x}|\mathcal{C}_k)$ and prior probabilities $p(\mathcal{C}_k)$, and compute $p(\mathcal{C}_k|\mathbf{x})$ using Bayes' theorem

$$p(\mathcal{C}_k|\mathbf{x}) = \frac{p(\mathbf{x}|\mathcal{C}_k)p(\mathcal{C}_k)}{p(\mathbf{x})}$$

- Probabilistic Discriminative Model
  Maximize a likelihood function defined through the conditional distribution $p(\mathcal{C}_k|\mathbf{x})$
- Advantages of Discriminative Models
  - Fewer adaptive parameters need to be determined.
  - Performance will be improved, especially when the class-conditional density assumption gives a poor approximation to the true distributions.

Caution

We have deviated from the textbook!

Predicting a Probability

Will someone have a heart attack over the next year?

age	62 years
gender	male
blood suger	120 mg/DL40,000
HDL	50
LDL	120
Mass	190 lbs
Height	5' 10"
...	...

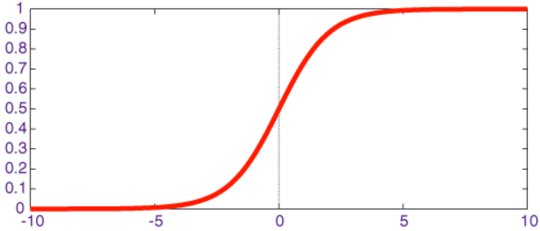
Classification: Yes/No $\Rightarrow$ Logistic Regression: Likelihood of heart attack.
In logistic regression, $y \in [0, 1]$.

$$h(\mathbf{x}) = \sigma\left(\sum_{i=0}^{d} w_i x_i\right) = \sigma(\mathbf{w}^T\mathbf{x})$$

Sigmoid Function σ

Logistic regression, $h(\mathbf{x}) = \sigma\left(\sum_{i=0}^{d} w_i x_i\right) = \sigma(\mathbf{w}^T\mathbf{x})$.

$$\sigma(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$

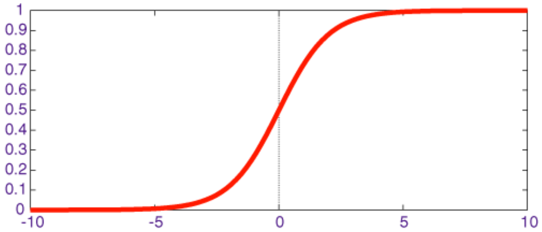


How is $\sigma(-s)$ related to $\sigma(s)$?

Sigmoid Function σ

Logistic regression, $h(\mathbf{x}) = \sigma\left(\sum_{i=0}^{d} w_i x_i\right) = \sigma(\mathbf{w}^T\mathbf{x})$.

$$\sigma(s) = \frac{e^s}{1+e^s} = \frac{1}{1+e^{-s}}$$



$$\sigma(-s) = \frac{e^{-s}}{1+e^{-s}} = \frac{1}{1+e^s} = 1 - \sigma(s)$$

The Data is Still Binary, ± 1

$$\mathcal{D} = (\mathbf{x}_1, y_1 = \pm 1), \ldots, (\mathbf{x}_N, y_N = \pm 1)$$

$$\mathbf{x}_n \leftarrow \text{a person's health information}$$
$$y_n = \pm 1 \leftarrow \textbf{did} \text{ they have a heart attack or not}$$

- We cannot measure a *probability*.
- We can only see the occurence of an event an try to *infer* a probability.
- In the textbook we used $+1/0$ encoding instead of $+1/-1$, which leads to a slightly different (but equivalent) formulation.

What Makes an h Good?

'fitting' the data means finding a good h

h is good if: $\begin{cases} h(\mathbf{x}_n) = \sigma(\mathbf{w}^T\mathbf{x}) \approx 1 & \text{whenever } y_n = +1; \\ h(\mathbf{x}_n) = \sigma(\mathbf{w}^T\mathbf{x}) \approx 0 & \text{whenever } y_n = -1. \end{cases}$

A simple error measure that captures this:

$$E(h) = \frac{1}{N} \sum_{n=1}^{N} \left(h(\mathbf{x}_n) - \frac{1}{2}(1 + y_n) \right)^2.$$

Hard to minimize!

The Logistic Loss Function

$$E(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + \exp\left(-y_n \cdot \mathbf{w}^T\mathbf{x}\right))$$

It looks complicated and ugly $(\ln, e^{(\cdot)}, \dots)$,
But,

- it is based on an intuitive probabilistic interpretation of h.
- it is very convenient and mathematically friendly ('easy' to minimize).

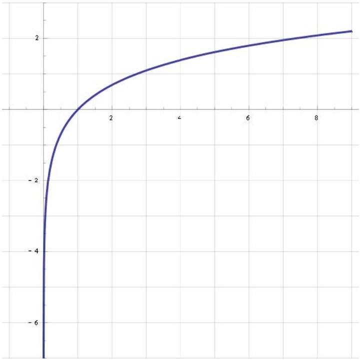
Verify:

- $y_n = +1$ encourages $\mathbf{w}^T\mathbf{x}_n \gg 0$, so $h(\mathbf{x}_n) = \sigma(\mathbf{w}^T\mathbf{x}_n) \approx 1$
- $y_n = -1$ encourages $\mathbf{w}^T\mathbf{x}_n \ll 0$, so $h(\mathbf{x}_n) = \sigma(\mathbf{w}^T\mathbf{x}_n) \approx 0$

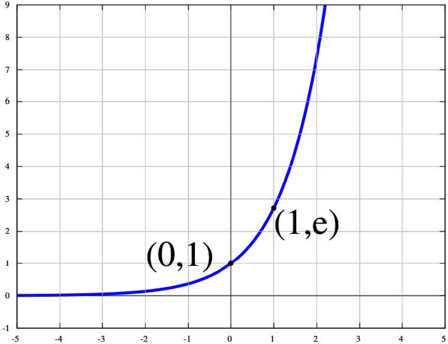
The Logistic Loss Function

$$E(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + \exp(-y_n \mathbf{w}^T \mathbf{x}_n))$$

ln



exp



The Probabilistic Interpretation

Suppose that $h(\mathbf{x}) = \sigma(\mathbf{w}^T\mathbf{x})$ closely captures $P[+1|\mathbf{x}]$

$$P(y|\mathbf{x}) = \begin{cases} \sigma(\mathbf{w}^T\mathbf{x}) & \text{for } y = +1; \\ 1 - \sigma(\mathbf{w}^T\mathbf{x}) & \text{for } y = -1. \end{cases}$$

And thus

$$P(y|\mathbf{x}) = \begin{cases} \sigma(\mathbf{w}^T\mathbf{x}) & \text{for } y = +1; \\ \sigma(-\mathbf{w}^T\mathbf{x}) & \text{for } y = -1. \end{cases}$$

... or, more compactly

$$P(y|\mathbf{x}) = \sigma(y \cdot \mathbf{w}^T\mathbf{x})$$

The Likelihood

$$P(y|\mathbf{x}) = \sigma(y \cdot \mathbf{w}^T \mathbf{x})$$

Assume: $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$ are independently generated.
Likelihood:

The probability of getting the $y_1, \ldots, y_N$ in $\mathcal{D}$ from the corresponding $\mathbf{x}_1, \ldots, \mathbf{x}_N$:

$$P(y_1, \ldots, y_N | \mathbf{x}_1, \ldots, \mathbf{x}_n) = \prod_{n=1}^{N} P(y_n | \mathbf{x}_n).$$

The likelihood measures the probability that the data were generated if f were h.

Maximizing the Likelihood

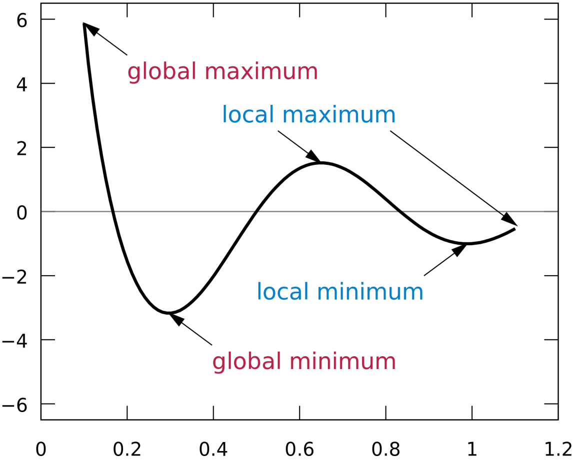
$$
\begin{aligned}
&\max \prod_{n=1}^{N} P(y_n|\mathbf{x}_n) \\
\Leftrightarrow &\max \ln \left(\prod_{n=1}^{N} P(y_n|\mathbf{x}_n) \right) \equiv \max \sum_{n=1}^{N} \ln P(y_n|\mathbf{x}_n) \\
\Leftrightarrow &\min -\frac{1}{N} \sum_{n=1}^{N} \ln P(y_n|\mathbf{x}_n) \equiv \min \frac{1}{N} \sum_{n=1}^{N} \ln \frac{1}{P(y_n|\mathbf{x}_n)} \\
\equiv &\min \frac{1}{N} \sum_{n=1}^{N} \ln \frac{1}{\sigma(y_n \mathbf{w}^T \mathbf{x}_n)} \\
\equiv &\min \frac{1}{N} \sum_{n=1}^{N} \ln(1 + \exp(-y_n \mathbf{w}^T \mathbf{x}_n))
\end{aligned}
$$

How to Minimize $E(\mathbf{w})$

- Classification - PLA/Pocket (iterative)
- Regression - Solving $\nabla_{\mathbf{w}} E(\mathbf{w}) = \mathbf{0}$.
- **Logistic Regression - No closed form solution.**

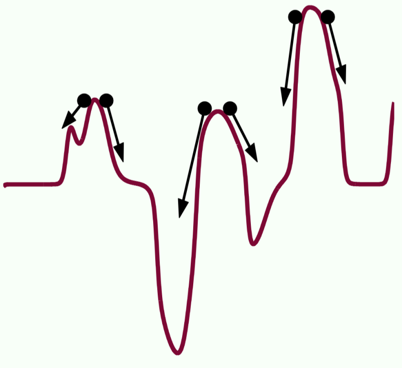
Numerically/iteratively set $\nabla_{\mathbf{w}} E(\mathbf{w}) \rightarrow \mathbf{0}$

Optimization 101



Finding The Best Weights - Hill Descent

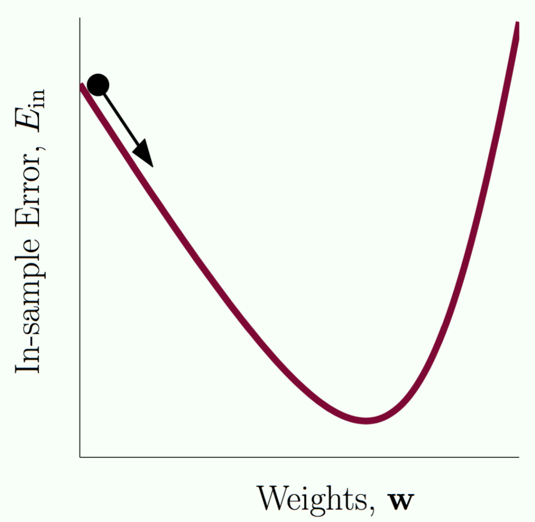
- Ball on a compliated hilly terrain
  - rolls down to a *local valley* (*a local minimum*)



- Questions
  - How to get to the bottom of the deepest valley?
  - How to do this when we don't have gravity?

Q1: How to get to the bottom of the deepest valley?

- Our E has only one valley



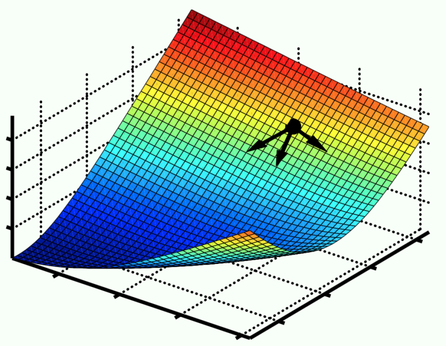
Q2: How to "roll down"

- Assume we are at weights $\mathbf{w}(t)$ and we take a step of size η in the direction $\hat{\mathbf{v}}$:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta\hat{\mathbf{v}}$$

  We get to pick $\hat{\mathbf{v}}$

- What's the best direction to take the step?



The Gradient is the Fastest Way to Roll Down

- We want $\hat{\mathbf{v}}$ that maximizes $\Delta E = E(\mathbf{w}(t)) - E(\mathbf{w}(t+1))$
- We can approximate the ΔE

$$\begin{aligned}\Delta E = E(\mathbf{w}(t)) - E(\mathbf{w}(t+1)) &= E(\mathbf{w}(t)) - E(\mathbf{w}(t) + \eta\hat{\mathbf{v}}) \\ &= E(\mathbf{w}(t)) - \left(E(\mathbf{w}(t)) + \eta\nabla E(\mathbf{w}(t))^T\hat{\mathbf{v}} + O(\eta^2)\right) \\ &= -\eta\nabla E(\mathbf{w}(t))^T\hat{\mathbf{v}} - O(\eta^2) \approx -\eta\nabla E(\mathbf{w}(t))^T\hat{\mathbf{v}}\end{aligned}$$

- When $\hat{\mathbf{v}}$ is a direction vector ($\|\hat{\mathbf{v}}\| = 1$), we have

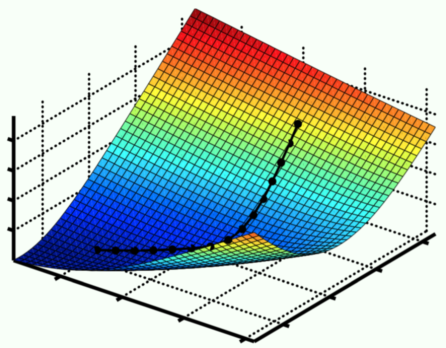
$$\Delta E \approx -\eta\nabla E(\mathbf{w}(t))^T\hat{\mathbf{v}} \leq \eta\|\nabla E(\mathbf{w}(t))\|_2$$

when $\hat{\mathbf{v}} = -\frac{\nabla E(\mathbf{w}(t))}{\|\nabla E(\mathbf{w}(t))\|_2}$

"Rolling Down" $\equiv$ Iterating the Negative Gradient

Update using negative gradient..

$$\mathbf{w}(0) \to \mathbf{w}(1) \to \mathbf{w}(2) \to \mathbf{w}(3) \to \ldots$$

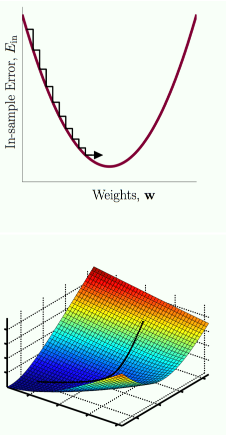


$\eta = 0.5$; 15 steps

How to set the step size?

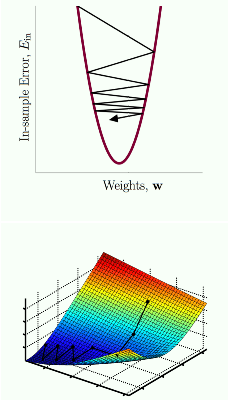
Step Size

η too small



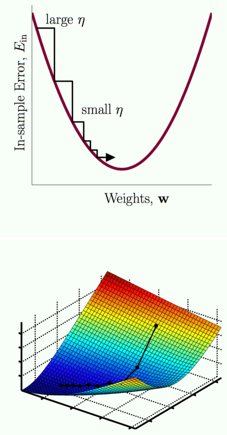
$\eta = 0.1$; 75 steps

η too large



$\eta = 2$; 10 steps

η just right



variable η_t; 10 steps

Fixed Learning Rate Gradient Descent

- We want $\eta_t \to 0$ when closer to the minimum:

$$\eta_t = \eta \cdot \|\nabla E(\mathbf{w}(t))\|_2$$

- Plug in our update rule:

$$\hat{\mathbf{v}} = -\eta_t \cdot \frac{\nabla E(\mathbf{w}(t))}{\|\nabla E(\mathbf{w}(t))\|_2} = -\eta \cdot \cancel{\|\nabla E(\mathbf{w}(t))\|_2} \cdot \frac{\nabla E(\mathbf{w}(t))}{\cancel{\|\nabla E(\mathbf{w}(t))\|_2}}$$

Learning Algorithm

1. Initialize at step $t = 0$ to $\mathbf{w}(0)$.
2. **for** $t = 0, 1, 3, \ldots$ **do**
    1. Compute the graident
    $$\mathbf{g}_t = \nabla E(\mathbf{w}(t))$$
    2. Update the weights (along the negative gradient):
    $$\mathbf{w}(t+1) = \mathbf{w}(t) + \eta(-\nabla E(\mathbf{w}(t)))$$
    3. Iterate 'until it is time to stop'
3. **end for**
4. Return the final weights.

Learning by Gradient Descent

- The gradient descent algorithm can be applied to minimize any *smooth* function, e.g.,
  - logistic regression $E(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + \exp(-y_n \mathbf{w}^T \mathbf{x}_n))$
  - ridge regression $E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N} (\mathbf{w}^T \mathbf{x}_n - y_n)^2 + \lambda \|\mathbf{w}\|_2^2$
- When the objective function is convex, we can obtain **one** global optimal solution.
- What if the data points cannot be loaded into the memory?

Stochastic Gradient Descent (SGD)

- A variation of GD that considers only the error on one data point.

$$E(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N} \ln(1+\exp(-y_n \cdot \mathbf{w}^T\mathbf{x}_n)) = \frac{1}{N}\sum_{n=1}^{N} \ell(\mathbf{w}, \mathbf{x}_n, y_n)$$

- Algorithm
  - Pick a random data point $(\mathbf{x}_*, y_*)$
  - Run an iteration of GD on $\ell(\mathbf{w}, \mathbf{x}_*, y_*)$
- Logistic Regression

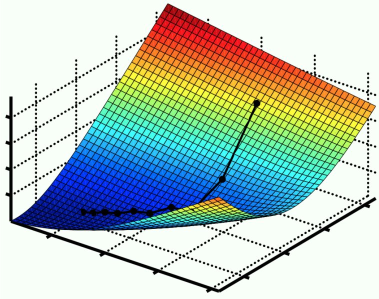
$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) + y_*\mathbf{x}_*\left(\frac{\eta}{1+\exp(+y_*\mathbf{w}^T\mathbf{x}_*)}\right)$$

Recall PLA:

$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) + y_*\mathbf{x}_*$$

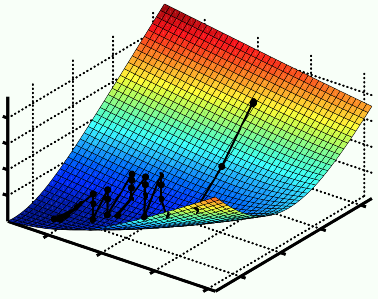
Stochastic Gradient Descent (SGD)

Gradient Descent



$\eta = 6, N = 10, t = 10$

Stochastic Gradient Descent



$\eta = 2, t = 30$

Remarks: Stochastic Gradient Descent

- The ‘average’ move is the same as GD;
- Computation: fraction $\frac{1}{N}$ cheaper per step
- Stochastic: helps escape local minima
- Simple
- Insights into logistic regression: similar to PLA.

Remarks

- Comparison of logistic regression and generative model in D-dimensional space
  - In logistic regression, only $D+1$ parameters (components of $\mathbf{w}$ and bias)
  - In generative model, suppose Gaussian class-conditional densities and maximum likelihood method are used, the number of parameters is $D(D+5)/2+1$
    - Means: $2D$ parameters
    - Shared covariance: $(D+1)D/2$ parameters
    - Prior $p(\mathcal{C}_1)$: 1 parameter
- **Maximum Likelihood** method is used to determine the parameters of the logistic regression model.
- Maximum likelihood can exhibit severe over-fitting.
- This can be overcome by inclusion of a prior and finding a MAP solution for $\mathbf{w}$, or equivalently by adding a regularization term to the error function.