Guerrilla Section #2

Topics

- Backprop
- Kernels
- SVM
- Duality
- QDA/LDA/Isocontours
- Logistic Regression

Backprop

Overview

Phase 1: propagation

Each propagation involves the following steps:

1. Propagation forward through the network to generate the output value(s)
2. Calculation of the cost (error term)
3. Propagation of the output activations back through the network using the training pattern target in order to generate the deltas (the difference between the targeted and actual output values) of all output and hidden neurons.

Phase 2: weight update

For each weight, the following steps must be followed:

1. The weight's output delta and input activation are multiplied to find the gradient of the weight.
2. A ratio (percentage) of the weight's gradient is subtracted from the weight.

This ratio (percentage) influences the speed and quality of learning; it is called the *learning rate*. The greater the ratio, the faster the neuron trains, but the lower the ratio, the more accurate the training is. The sign of the gradient of a weight indicates whether the error varies directly with, or inversely to, the weight. Therefore, the weight must be updated in the opposite direction, "descending" the gradient.

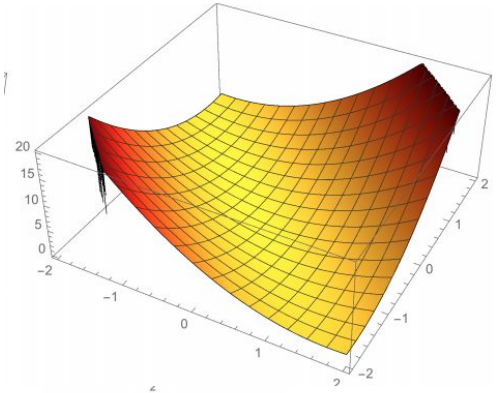
Tips

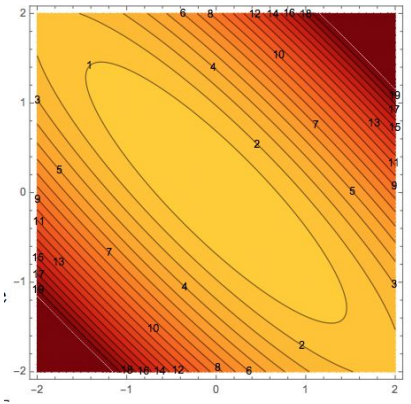
1) Chain Rule, Quotient Rule
2) Think about what weights and neurons are involved at each layer
3) Try a small example if you’re stuck

QDA/Isocontours

Isocontours of the Quadratic Form

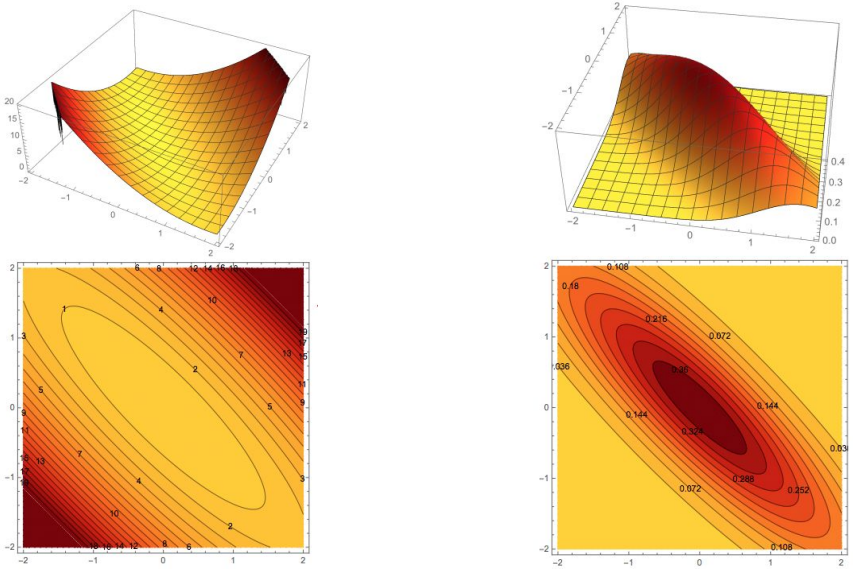
- An isocontour for a function f is $\{x : f(x) = t\}$ for some t. We call this an isocontour for t.
- Particularly interested in isocontours of quadratic form: $x^T Ax = 1$ where A is symmetric PSD
- Cases
  - All eigenvalues where eigenvalues are positive: ellipsoid with axes lengths $\frac{1}{\sqrt{\lambda_i}}$ and axes u_i
  - Some positive, some negative: hyperboloid





Visualization of a Gaussian

- Isocontours of gaussian just like isocontours of quadratic form with first case
- Covariance matrix PSD symmetric, so we get ellipsoidal isocontours
- $x^T \Sigma^{-1} x = 1$:



QDA

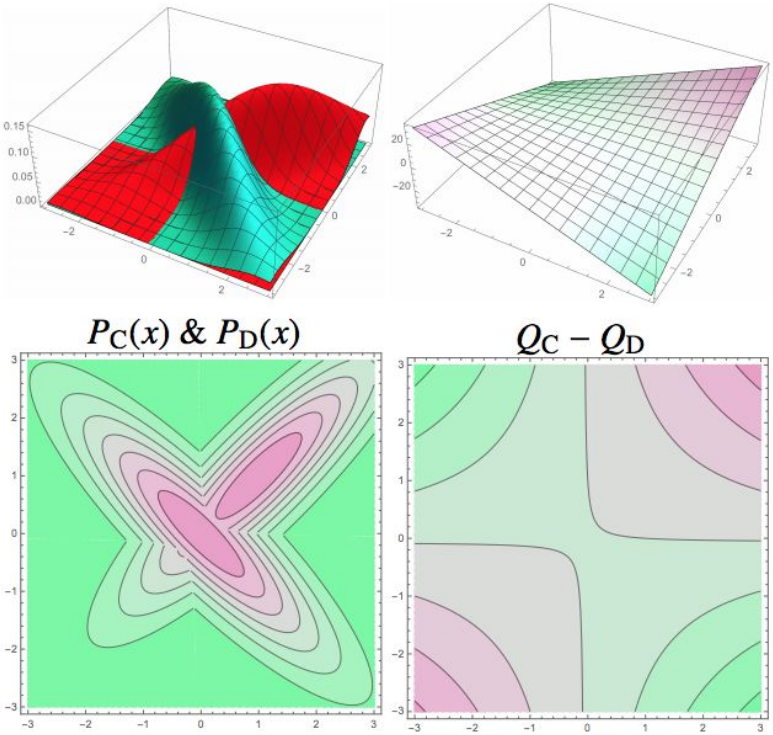
- Choosing class C that maximizes $P(X = x \mid Y = C)\pi_c$ equivalent to:

$$Q_C(x) = \ln((\sqrt{2\pi})^d P_C(x)\,\pi_C) = -\frac{1}{2}(x-\mu_C)^\top \Sigma_C^{-1}(x-\mu_C) - \frac{1}{2}\ln|\Sigma_C| + \ln\pi_C$$

↑
Gaussian for C

- Note this has a quadratic decision boundary
- For two classes, $Q_C(x) - Q_D(x)$ is the decision boundary

QDA Decision Boundary



SVM/Duality

SVM Overview

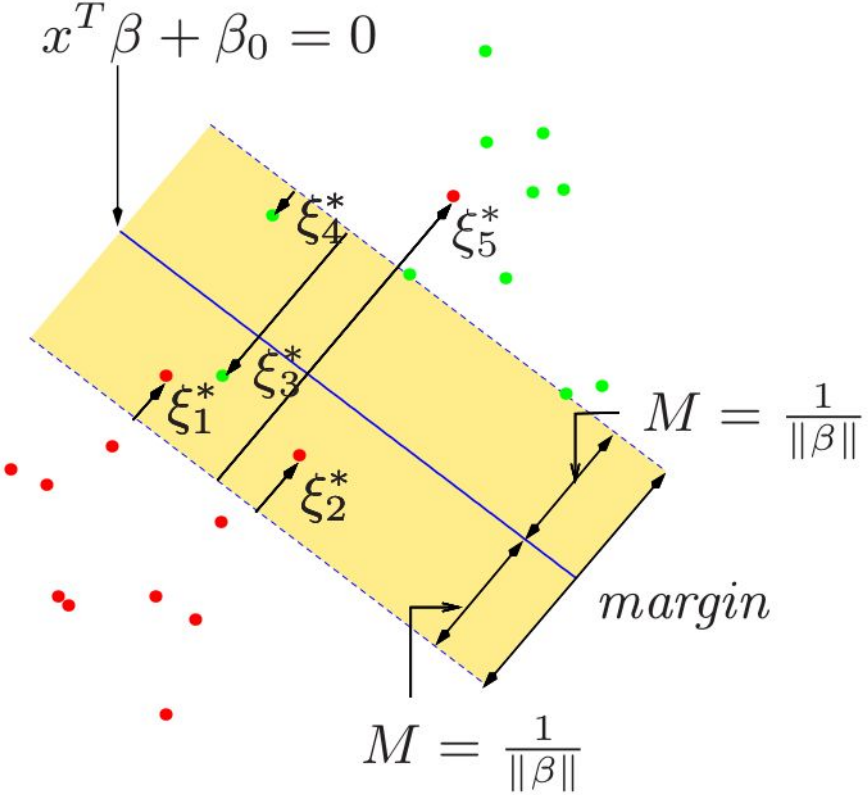
- SVM a quadratic optimization problem of form

$$\max_{w,t,\xi} \varepsilon(w,t,\xi) = \max_{w,t,\xi} \frac{1}{2}||w||_2^2 + \mathcal{C}\sum_{i=1}^{n} \xi_i$$

$$\text{s.t. } y_i(w \cdot x_i - t) \geq 1 - \xi_i,\ \xi_i \geq 0$$

- Points are allowed to violate margin but at cost
- ξ_i known as "slack" term; amount to violate margin
- $\mathcal{C} \geq 0$ controls amount to penalize for violating margin

Geometric Interpretation (Primal)



Duality

- Primal problem:

$$\min_x f(x)$$
$$\text{s.t. } g(x) \leq 0$$

- Dual problem:

$$\max_\alpha \min_x \mathcal{L}(x, \alpha)$$
$$\text{s.t. } \alpha \geq 0$$

- $\mathcal{L}(x, \alpha)$ is known as the Lagrangian of the primal
- $\mathcal{L}(x, \alpha) = f(x) + \alpha^T g(x)$

KKT Conditions

The following are necessary conditions at optimum:

1. Stationary: $\frac{d\mathcal{L}}{dx} = \frac{df(x)}{dx} + \alpha^T \frac{dg(x)}{dx} = 0$
2. Primal feasibility: $g(x) \leq 0$
3. Dual feasibility: $\alpha \geq 0$
4. Complementary slackness: $\alpha_i g(x)_i = 0$

Kernels

Kernels

A kernel is a function $k : D \times D \to R$, where $D \subseteq \mathcal{R}^d$ is the domain, which is

1. Symmetric: $k(x, x') = k(x', x)$ for all $x, x' \in D$
2. Positive Semi-definite: for every finite collection $\{x_1, \ldots, x_n\} \subseteq D$, the Gram matrix K given by $K_{ij} = k(x_i, x_j)$ is positive semi-definite

Kernel Trick

- Try to rewrite computations can be written in terms of inner products
- If so, we can replace the inner products with calls to a kernel function
- Now possible to implicitly perform inner product in some higher (possibly infinite) dimensional space

Logistic Regression

Motivation

- For binary classification, we want to know the probability with which our prediction is correct
- Compute

$$P(y = 1|x)$$

Probability label is 1 given prediction

$$P(y = 0|x)$$

Probability label is 0 given prediction

Cost function

- Sigmoid function to predict probabilities
  - Limits the output to (0,1)
  - This gives your necessary probability

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}},$$

$$g(z) = \frac{1}{1 + e^{-z}}$$

- Cost function
  - Probability y in each class

Find w that minimizes

$$J = -\sum_{i=1}^{n} \left(y_i \ln s(X_i \cdot w) + (1 - y_i) \ln (1 - s(X_i \cdot w)) \right)$$