10-601 Introduction to Machine Learning

Machine Learning Department
School of Computer Science
Carnegie Mellon University

Neural Networks

Matt Gormley
Lecture 11
Feb. 21, 2018

Reminders

- **Homework 4: Logistic Regression**
  - **Out: Wed, Feb 16**
  - **Due: Sun, Feb 25 at 11:59pm**
- **New reading on Probabilistic Learning** (reused later in the course for MLE/MAP)
- **Schedule changes:**
  - lecture on Friday (2/23)
  - no lecture on Monday (2/26)

Q&A

Neural Networks Outline

- **Logistic Regression (Recap)**
  - Data, Model, Learning, Prediction
- **Neural Networks**
  - A Recipe for Machine Learning
  - Visual Notation for Neural Networks
  - Example: Logistic Regression Output Surface
  - 2-Layer Neural Network
  - 3-Layer Neural Network
- **Neural Net Architectures**
  - Objective Functions
  - Activation Functions
- **Backpropagation**
  - Basic Chain Rule (of calculus)
  - Chain Rule for Arbitrary Computation Graph
  - Backpropagation Algorithm
  - Module-based Automatic Differentiation (Autodiff)

NEURAL NETWORKS

Background

A Recipe for Machine Learning

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^N$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

Face

Face

Not a face

Examples: Linear regression, Logistic regression, Neural Network

Examples: Mean-squared error, Cross Entropy

Background

A Recipe for Machine Learning

1. Given training data:

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^N$$

2. Choose each of these:
   - Decision function

$$\hat{\boldsymbol{y}} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

3. Define goal:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^N \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

4. Train with SGD:
(take small steps opposite the gradient)

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

Background

A Recipe for

Gradients

Backpropagation can compute this gradient!

And it's a **special case of a more general algorithm** called reverse-mode automatic differentiation that can compute the gradient of any differentiable function efficiently!

1. Given training dat

$$\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^N$$

2. Choose each of th

– Decision function

$$\hat{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

– Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

opposite the gradient)

$$\boldsymbol{\theta}^{(t} \quad {}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

Goals for Today's Lecture

1. Explore a **new class of decision functions** (Neural Networks)
2. Consider **variants of this recipe** for training

A Recipe for

2. Choose each of these:
   - Decision function

$$\hat{y} = f_{\boldsymbol{\theta}}(\boldsymbol{x}_i)$$

   - Loss function

$$\ell(\hat{\boldsymbol{y}}, \boldsymbol{y}_i) \in \mathbb{R}$$

4. Train with SGD: (Take small steps opposite the gradient)

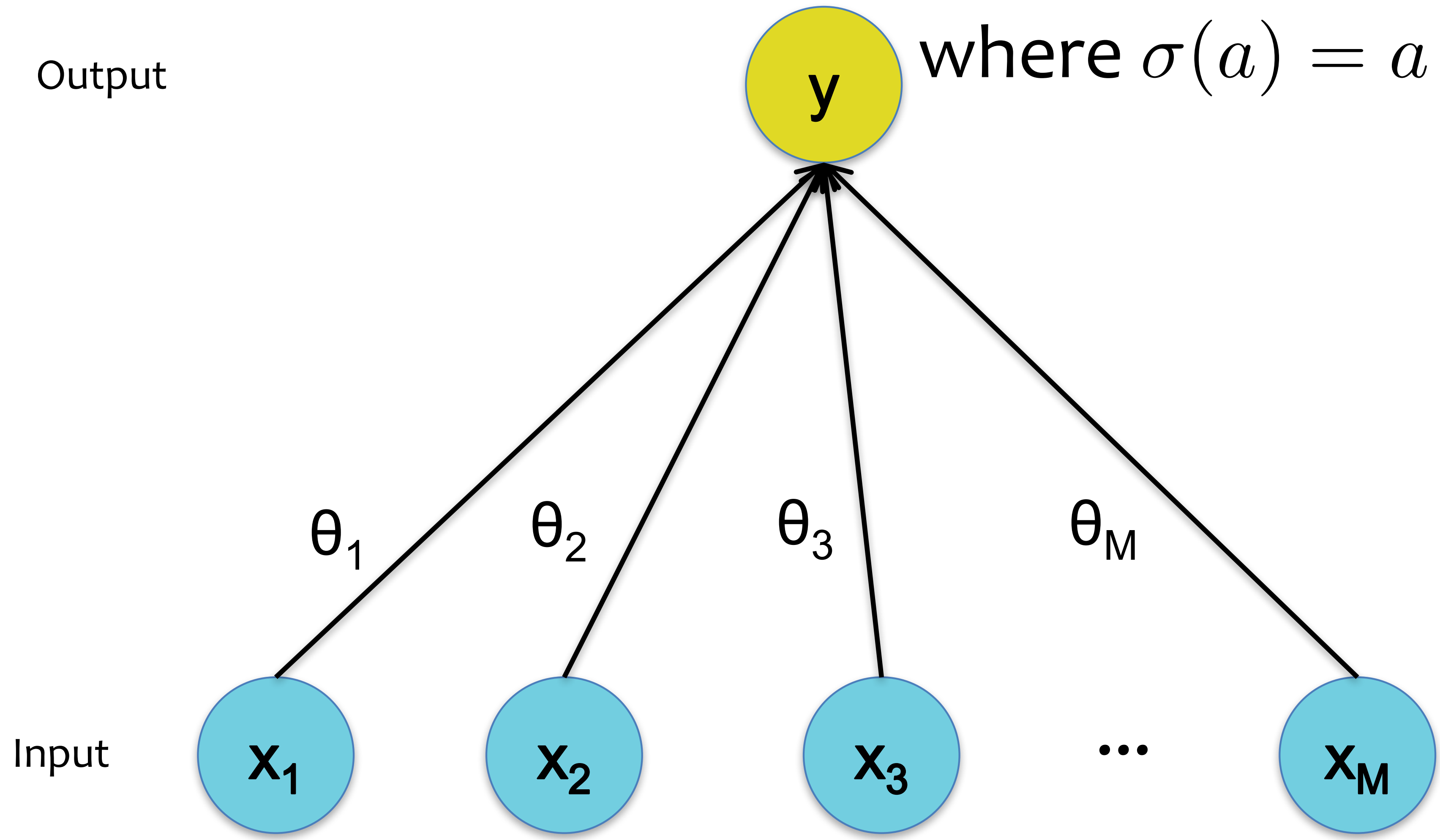
$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \eta_t \nabla \ell(f_{\boldsymbol{\theta}}(\boldsymbol{x}_i), \boldsymbol{y}_i)$$

Decision Functions

Linear Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

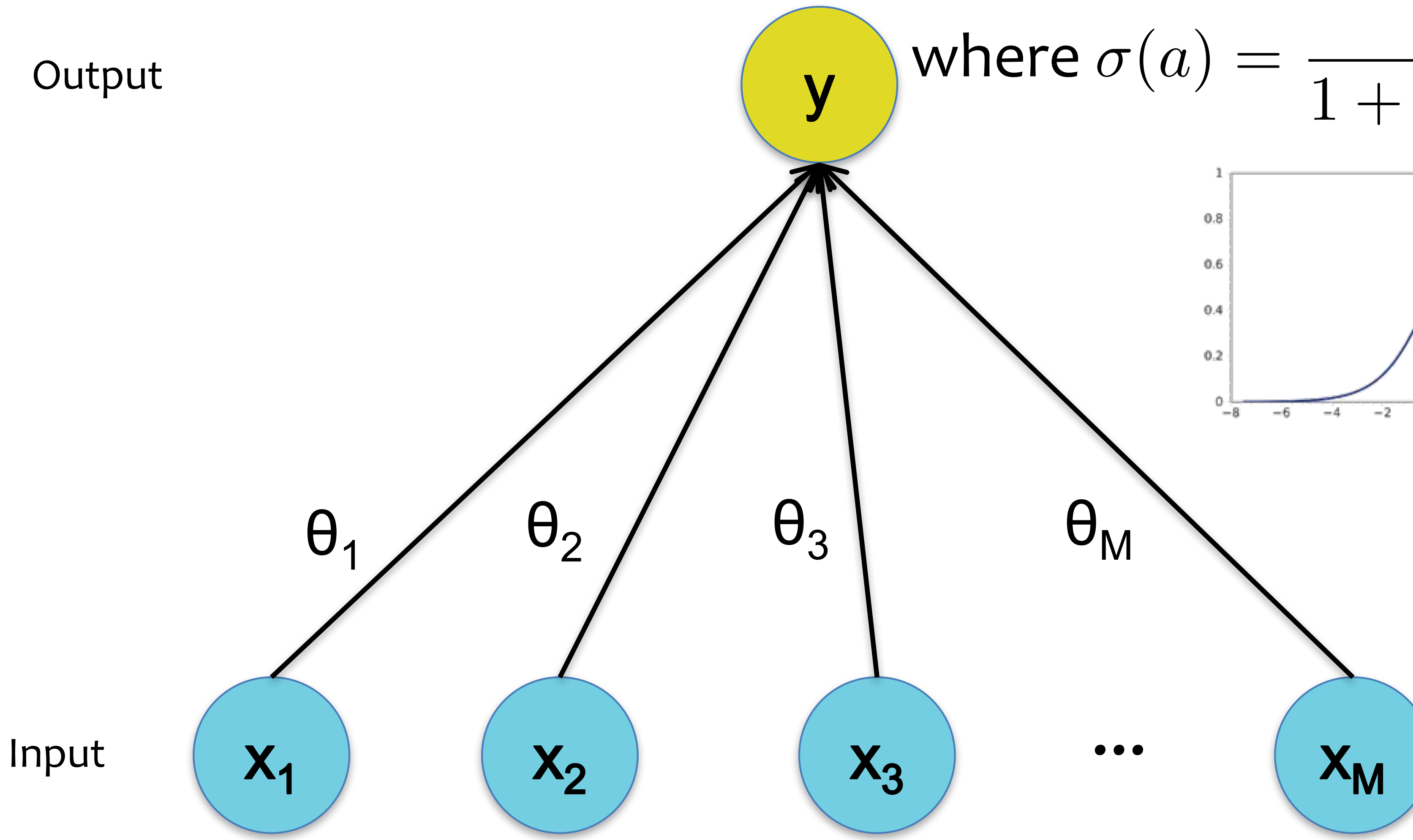
where $\sigma(a) = a$

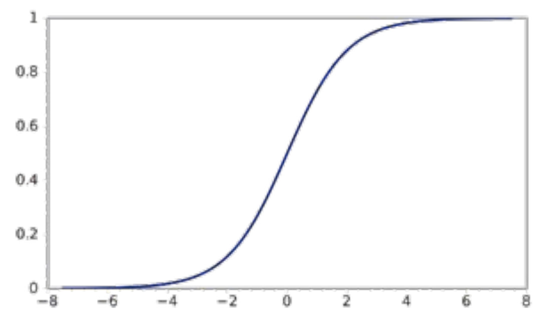


Logistic Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

where $\sigma(a) = \dfrac{1}{1 + \exp(-a)}$





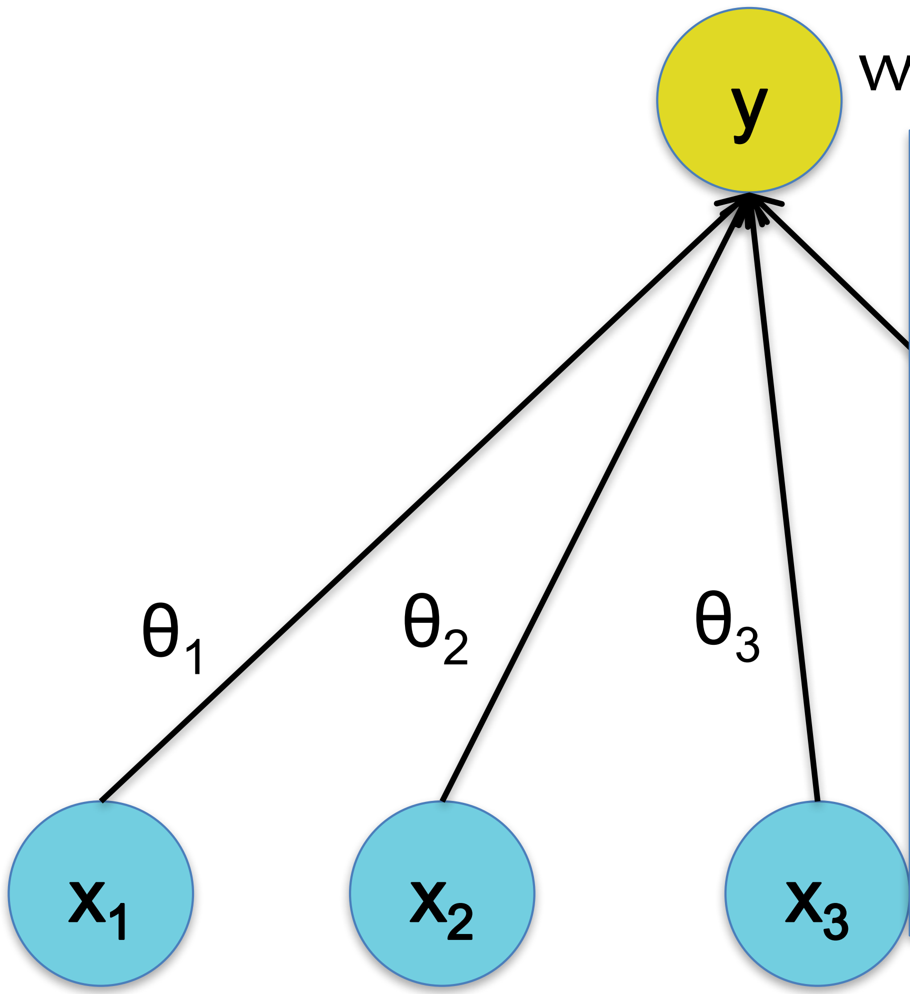
Decision Functions

Logistic Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

where $\sigma(a) = \frac{1}{1 + \exp(-a)}$

Output



Input

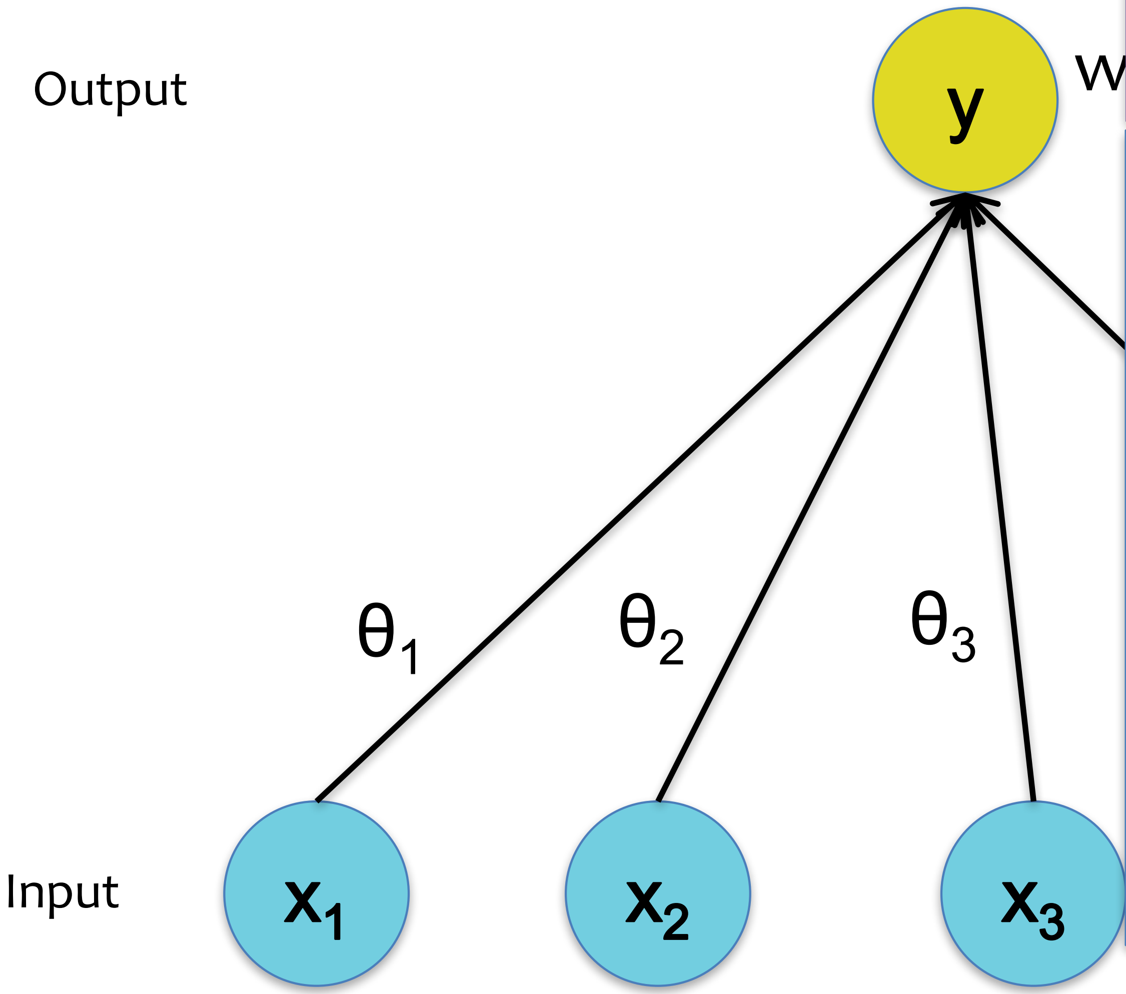
Face

Face

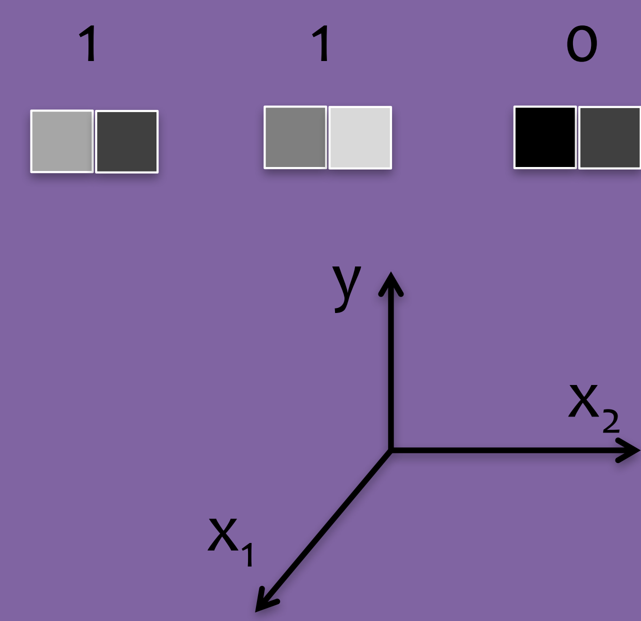
Not a face

Logistic Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$



In-Class Example

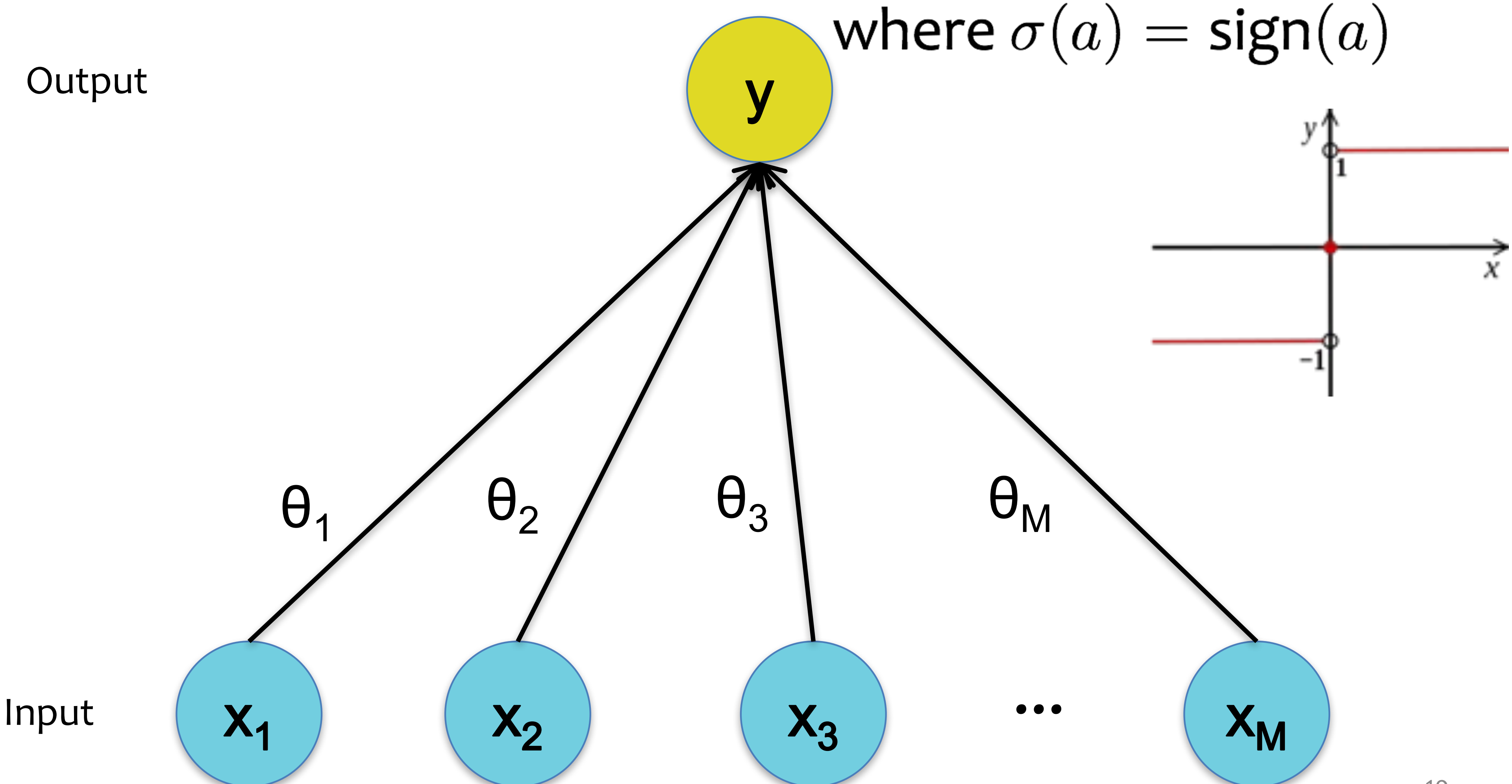


Decision Functions

Perceptron

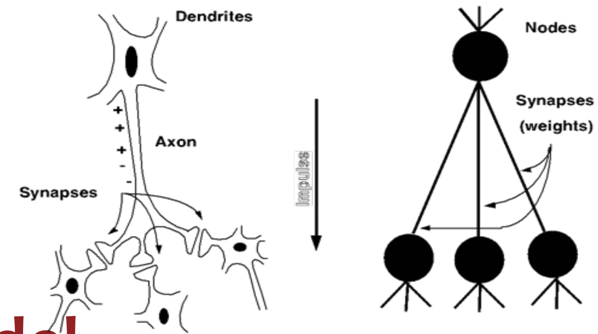
$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

where $\sigma(a) = \text{sign}(a)$



From Biological to Artificial

The motivation for Artificial Neural Networks comes from biology…



Biological “Model”

- **Neuron:** an excitable cell
- **Synapse:** connection between neurons
- A *neuron* sends an **electrochemical pulse** along its *synapses* when a sufficient voltage change occurs
- **Biological Neural Network:** collection of neurons along some pathway through the brain

Artificial Model

- **Neuron:** node in a directed acyclic graph (DAG)
- **Weight:** multiplier on each edge
- **Activation Function:** nonlinear thresholding function, which allows a neuron to “fire” when the input value is sufficiently high
- **Artificial Neural Network:** collection of neurons into a DAG, which define some differentiable function

Biological “Computation”

- Neuron switching time : ~ 0.001 sec
- Number of neurons: ~ 10^{10}
- Connections per neuron: ~ $10^{4\text{-}5}$
- Scene recognition time: ~ 0.1 sec

Artificial Computation

- Many neuron-like threshold switching units
- Many weighted interconnections among units
- Highly parallel, distributed processes

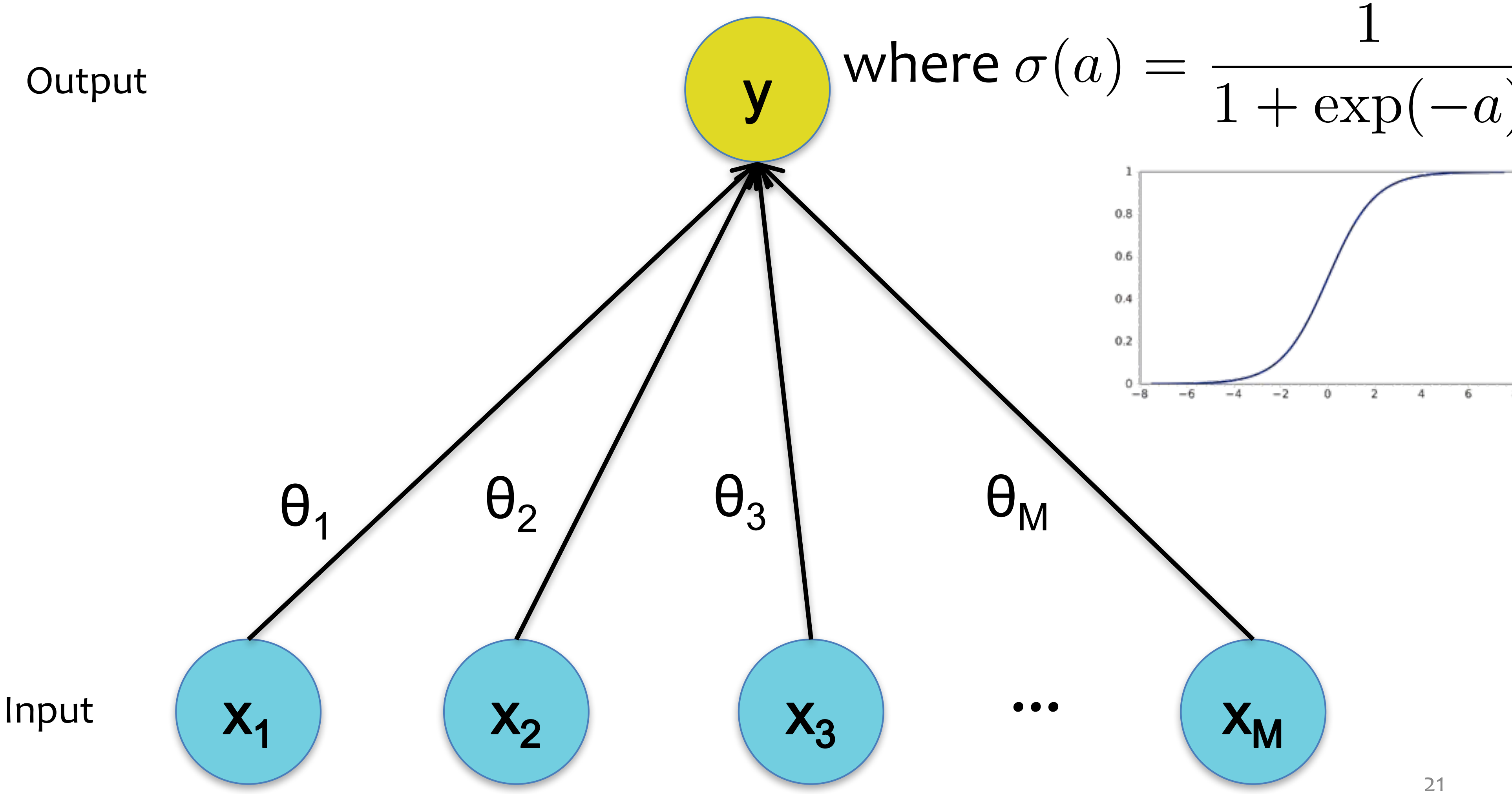
Slide adapted from Eric Xing

Decision Functions

Logistic Regression

$$y = h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \sigma(\boldsymbol{\theta}^T \boldsymbol{x})$$

where $\sigma(a) = \dfrac{1}{1 + \exp(-a)}$



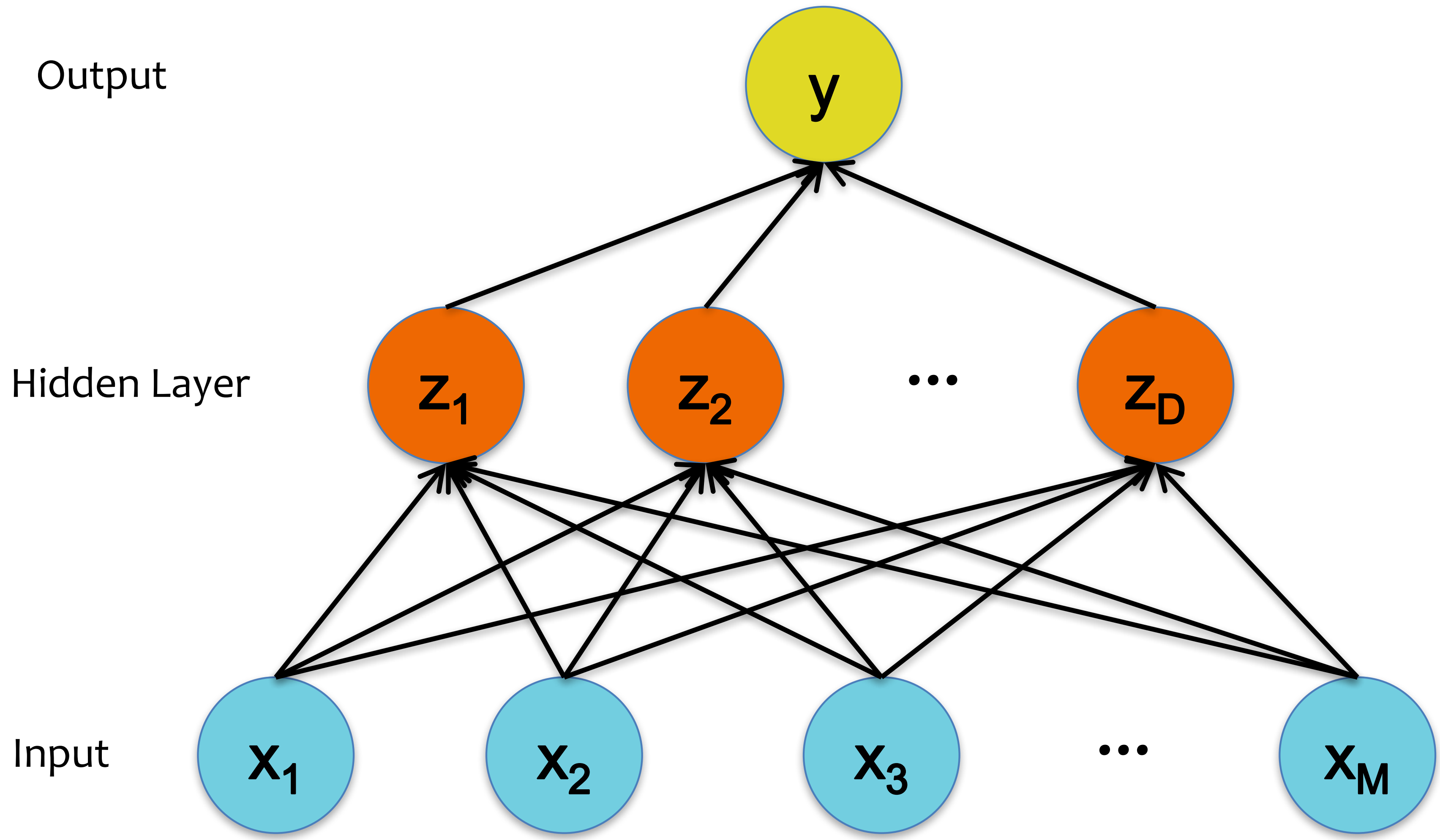
Neural Networks

Chalkboard

- Example: Neural Network w/1 Hidden Layer
- Example: Neural Network w/2 Hidden Layers
- Example: Feed Forward Neural Network

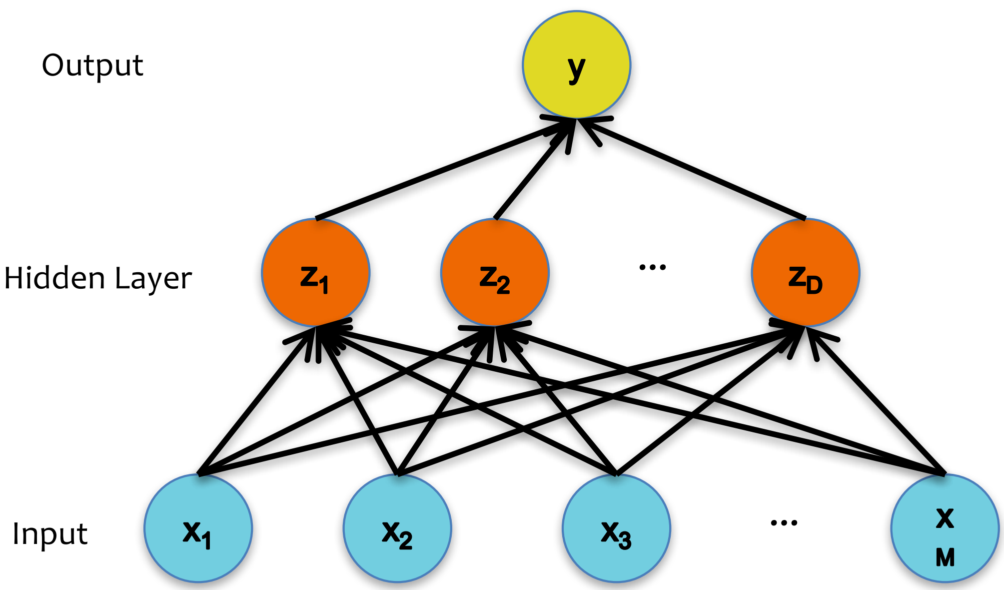
Decision Functions

Neural Network



Decision Functions

Neural Network



(E) **Output (sigmoid)**

$$y = \frac{1}{1+\exp(-b)}$$

(D) **Output (linear)**

$$b = \sum_{j=0}^{D} \beta_j z_j$$

(C) **Hidden (sigmoid)**

$$z_j = \frac{1}{1+\exp(-a_j)}, \ \ \forall j$$

(B) **Hidden (linear)**

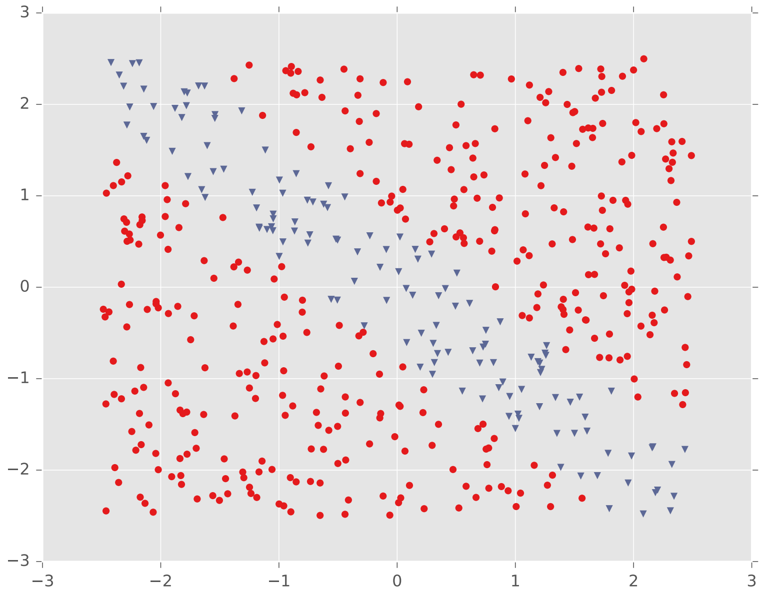
$$a_j = \sum_{i=0}^{M} \alpha_{ji} x_i, \ \ \forall j$$

(A) **Input**

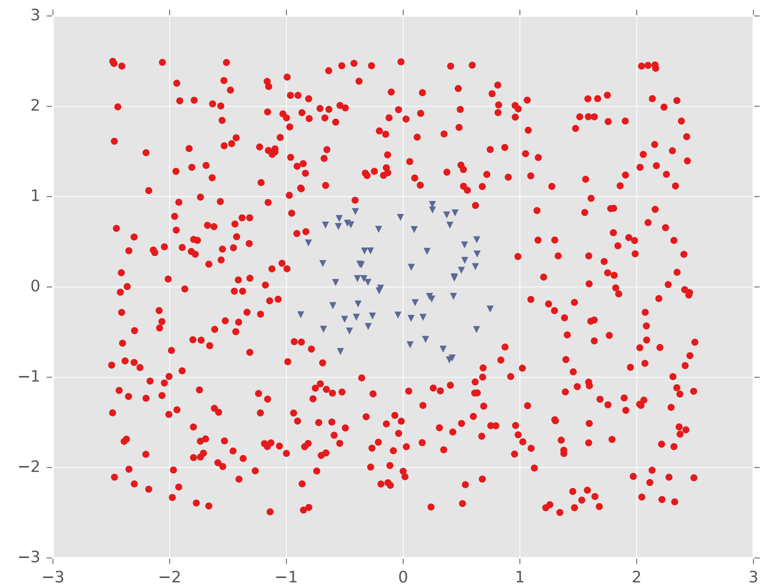
Given $x_i, \ \ \forall i$

DECISION BOUNDARY EXAMPLES

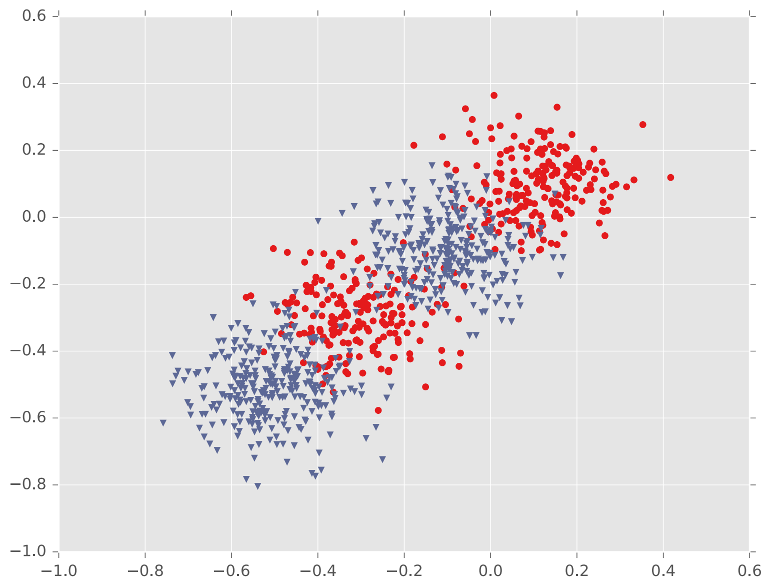
Example #1: Diagonal Band



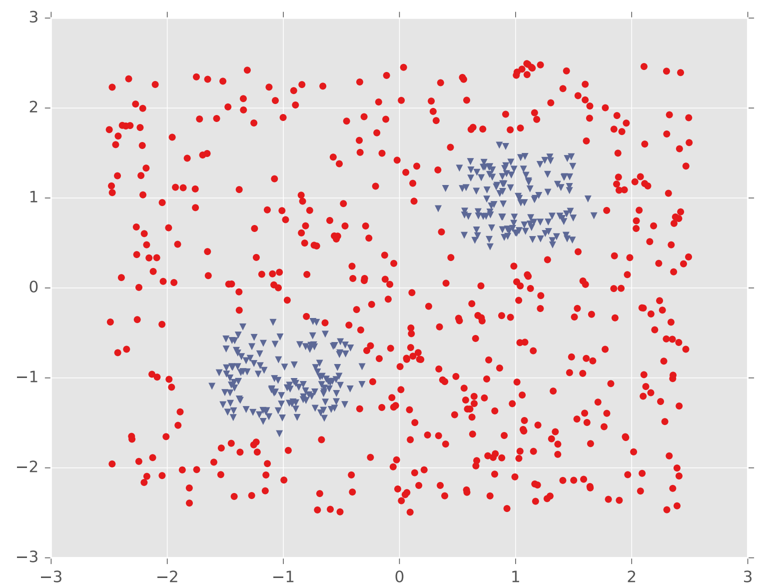
Example #2: One Pocket



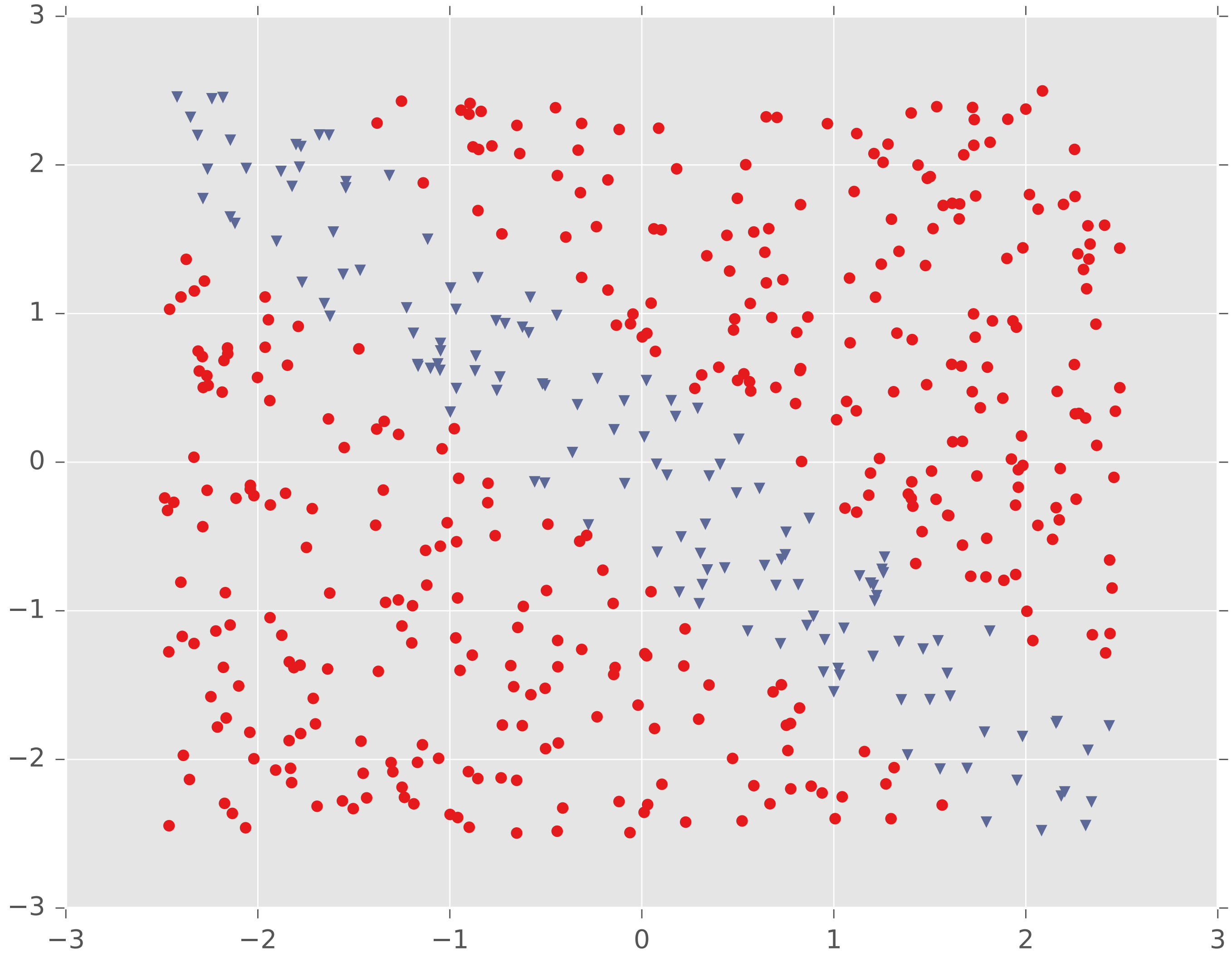
Example #3: Four Gaussians



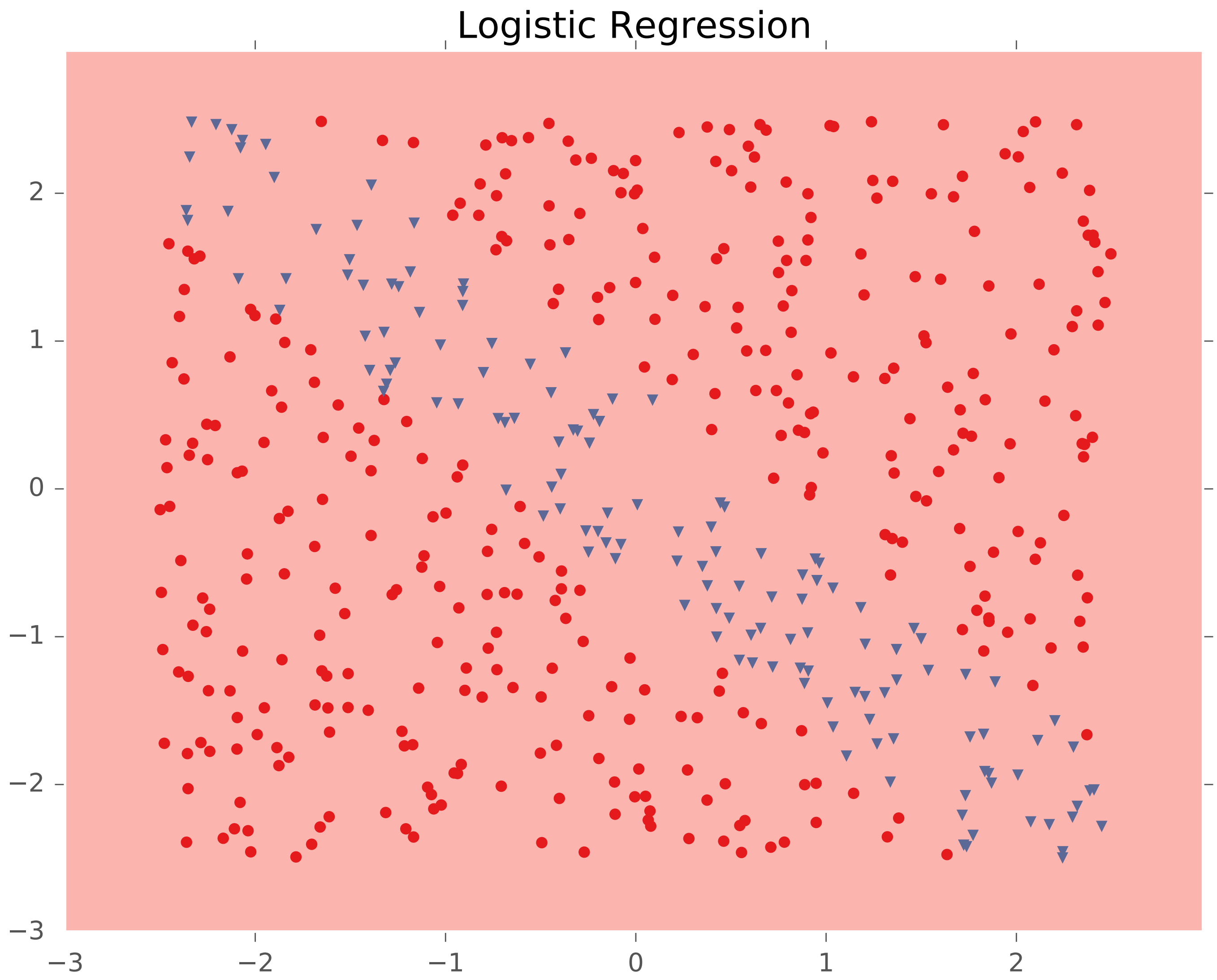
Example #4: Two Pockets



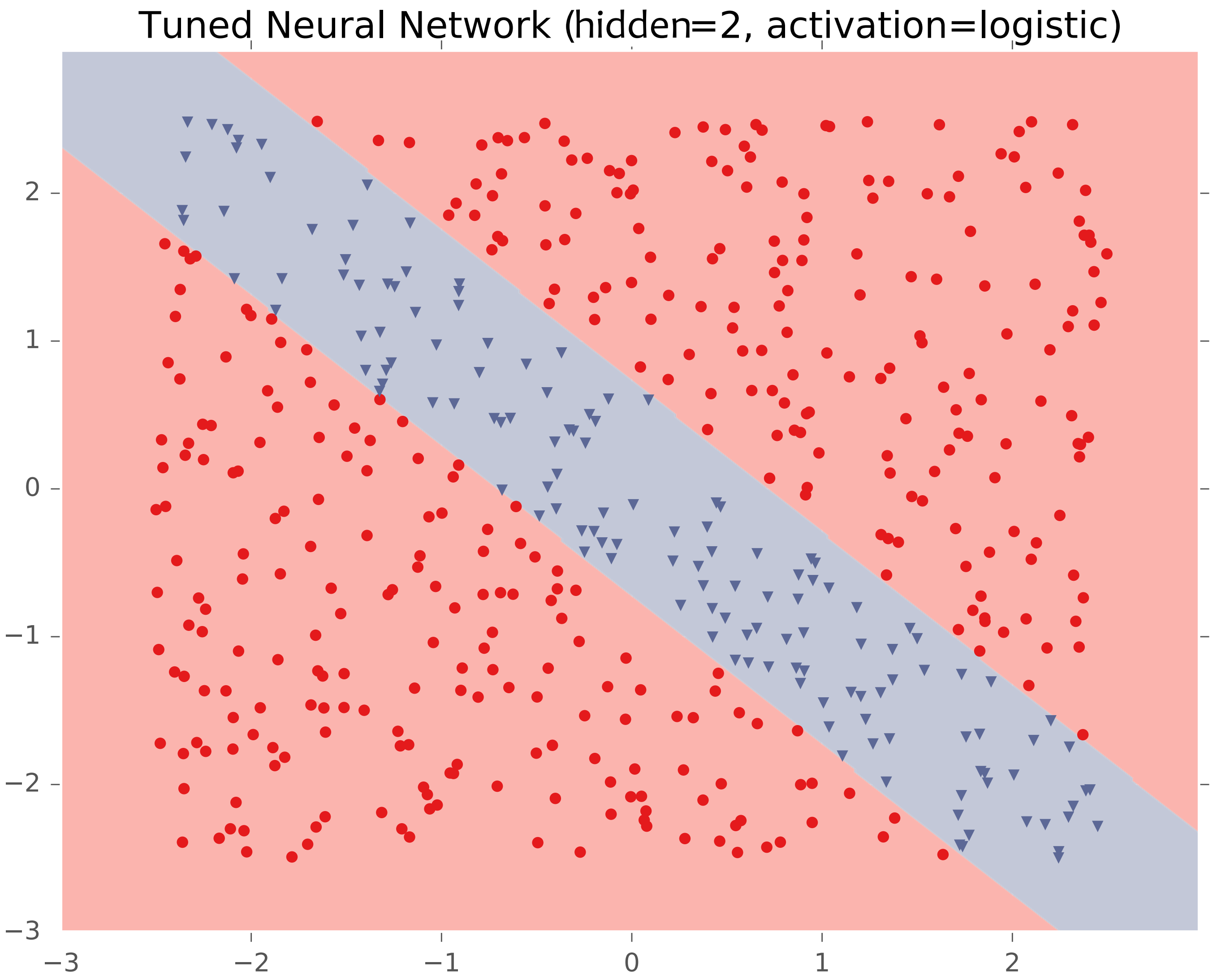
Example #1: Diagonal Band



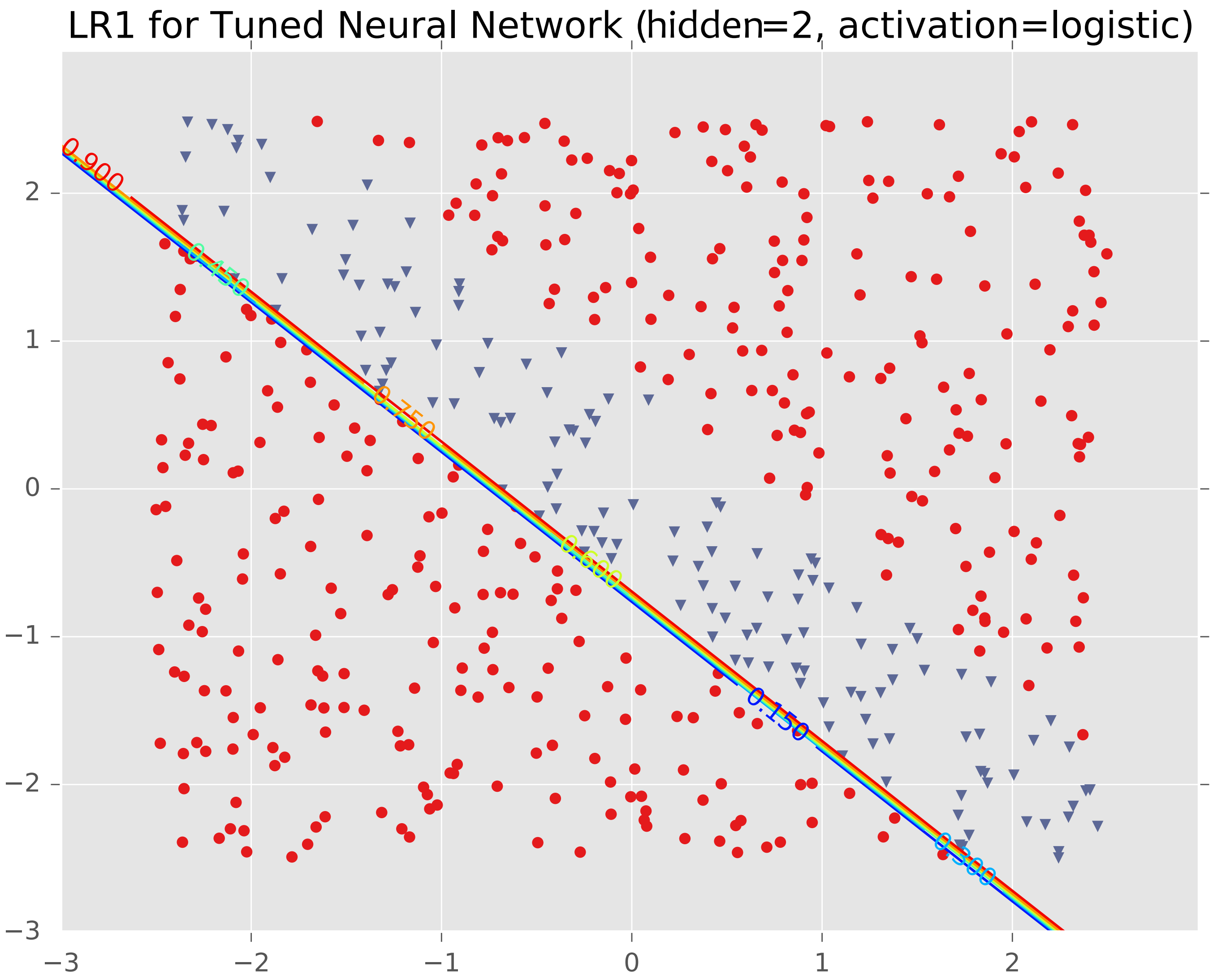
Example #1: Diagonal Band



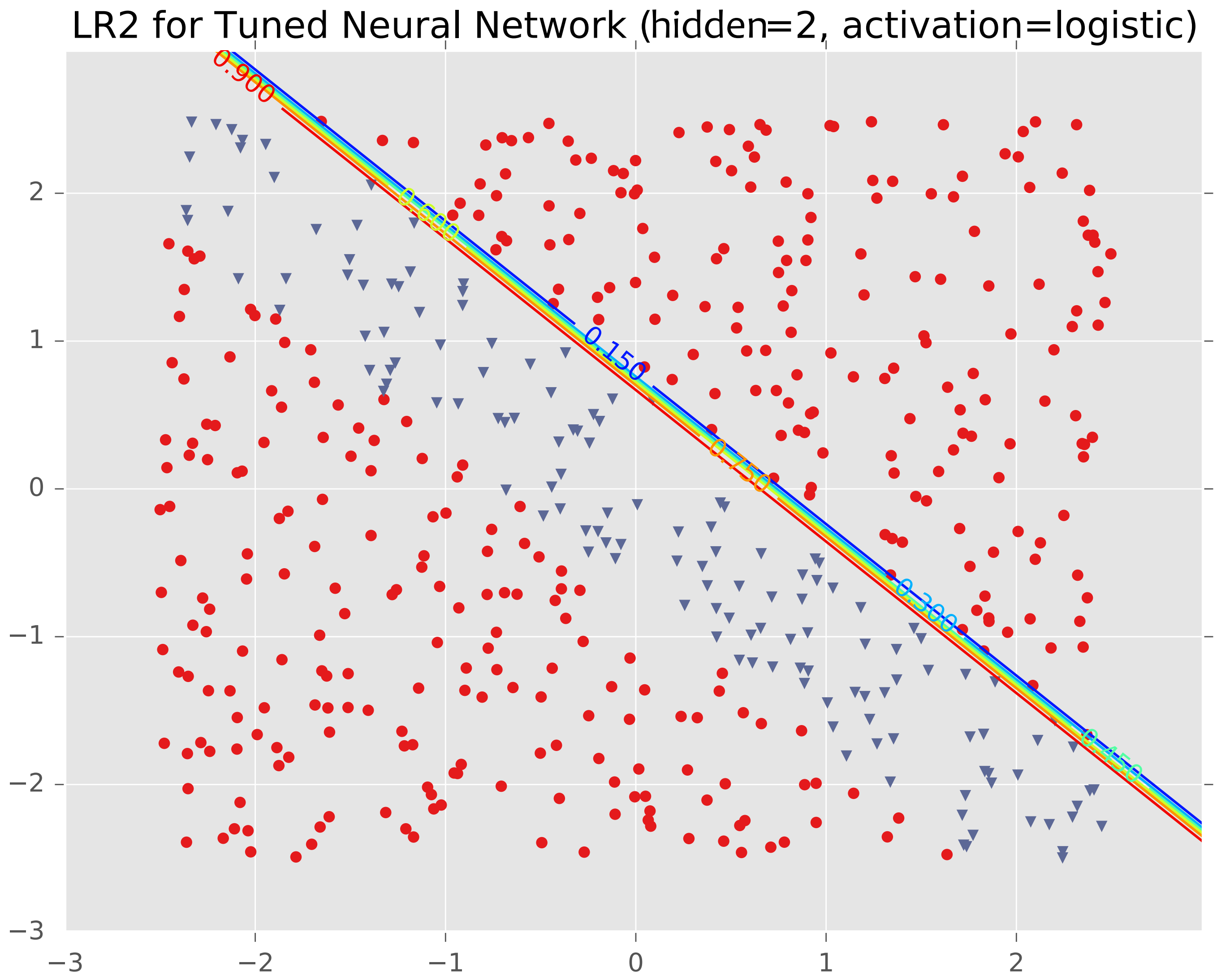
Example #1: Diagonal Band



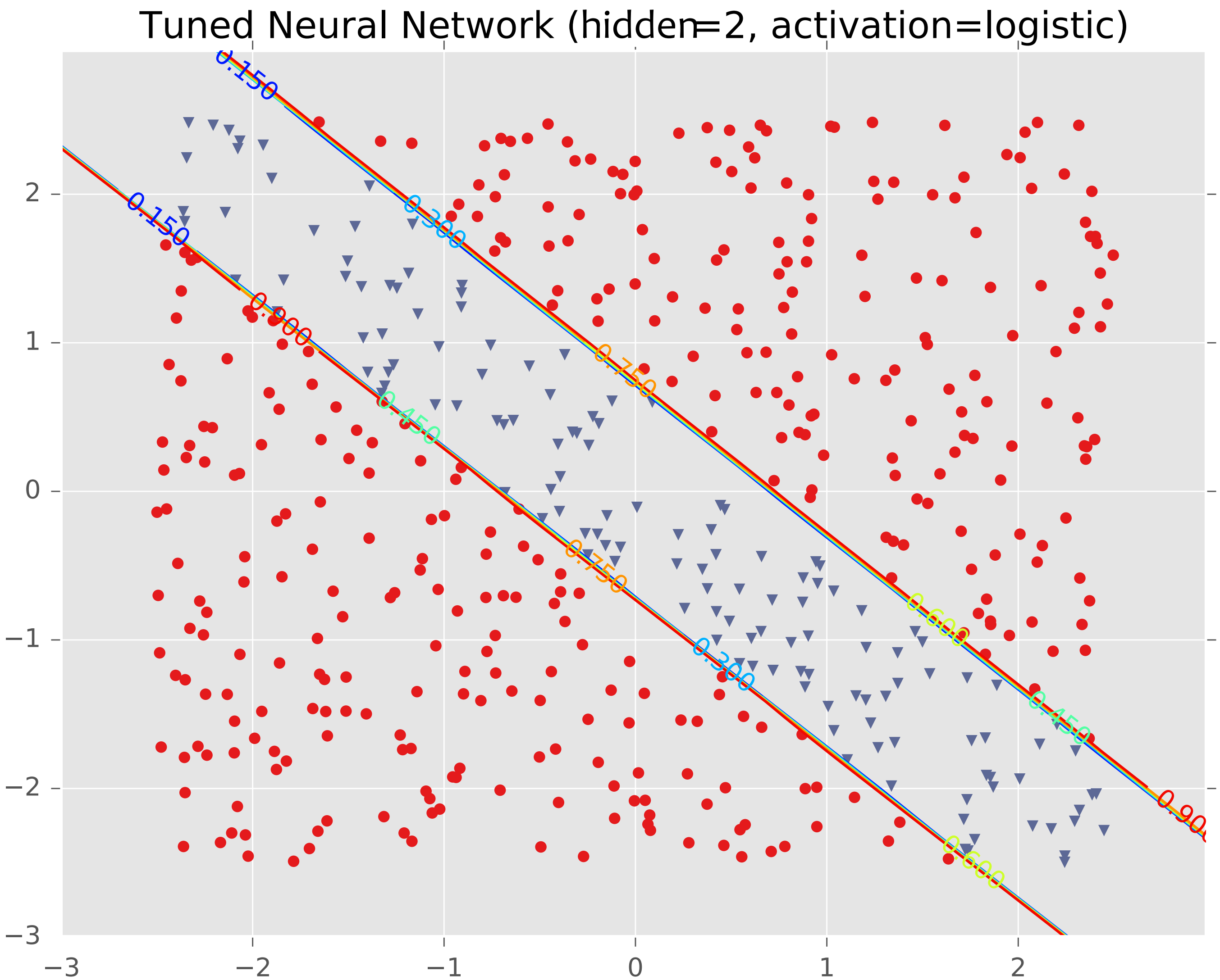
Example #1: Diagonal Band



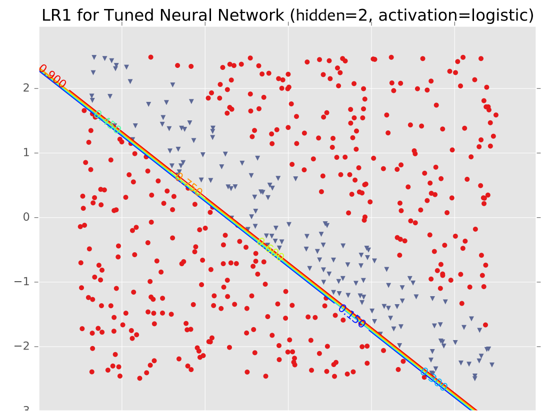
Example #1: Diagonal Band

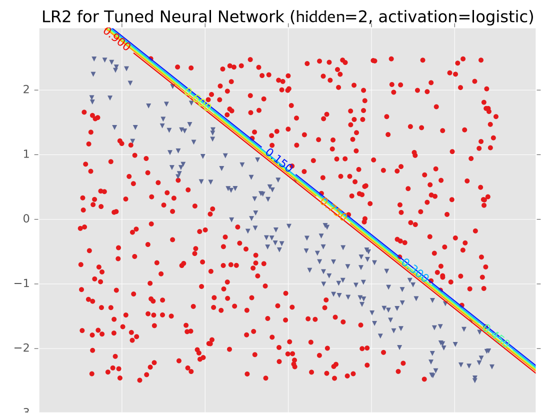


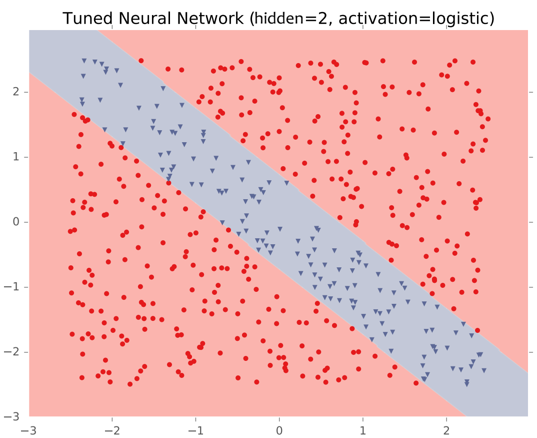
Example #1: Diagonal Band

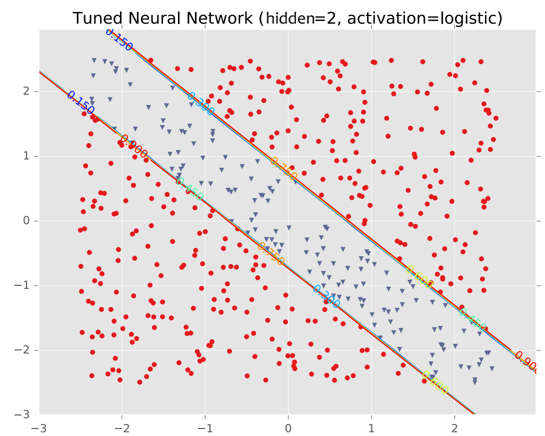


Example #1: Diagonal Band

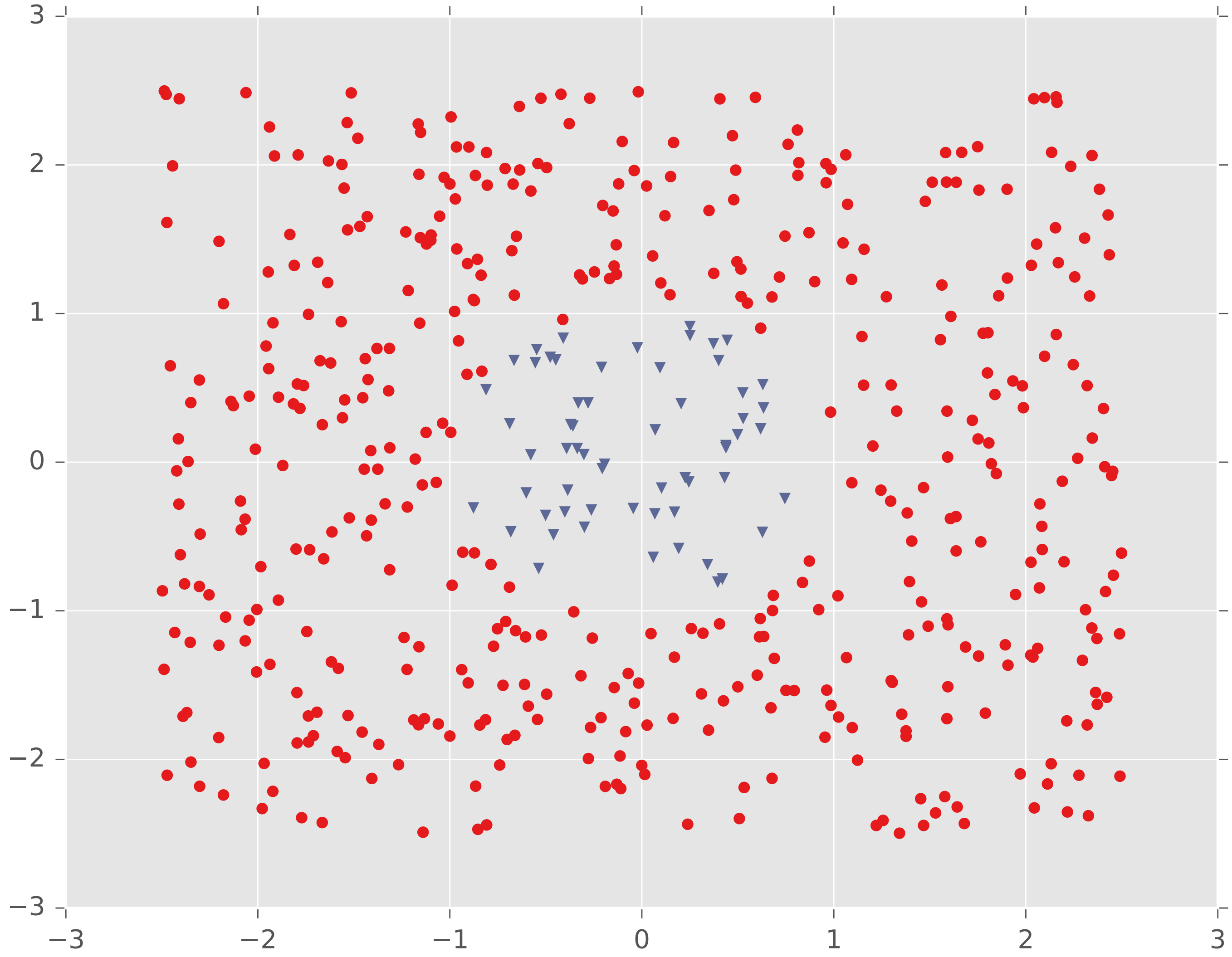




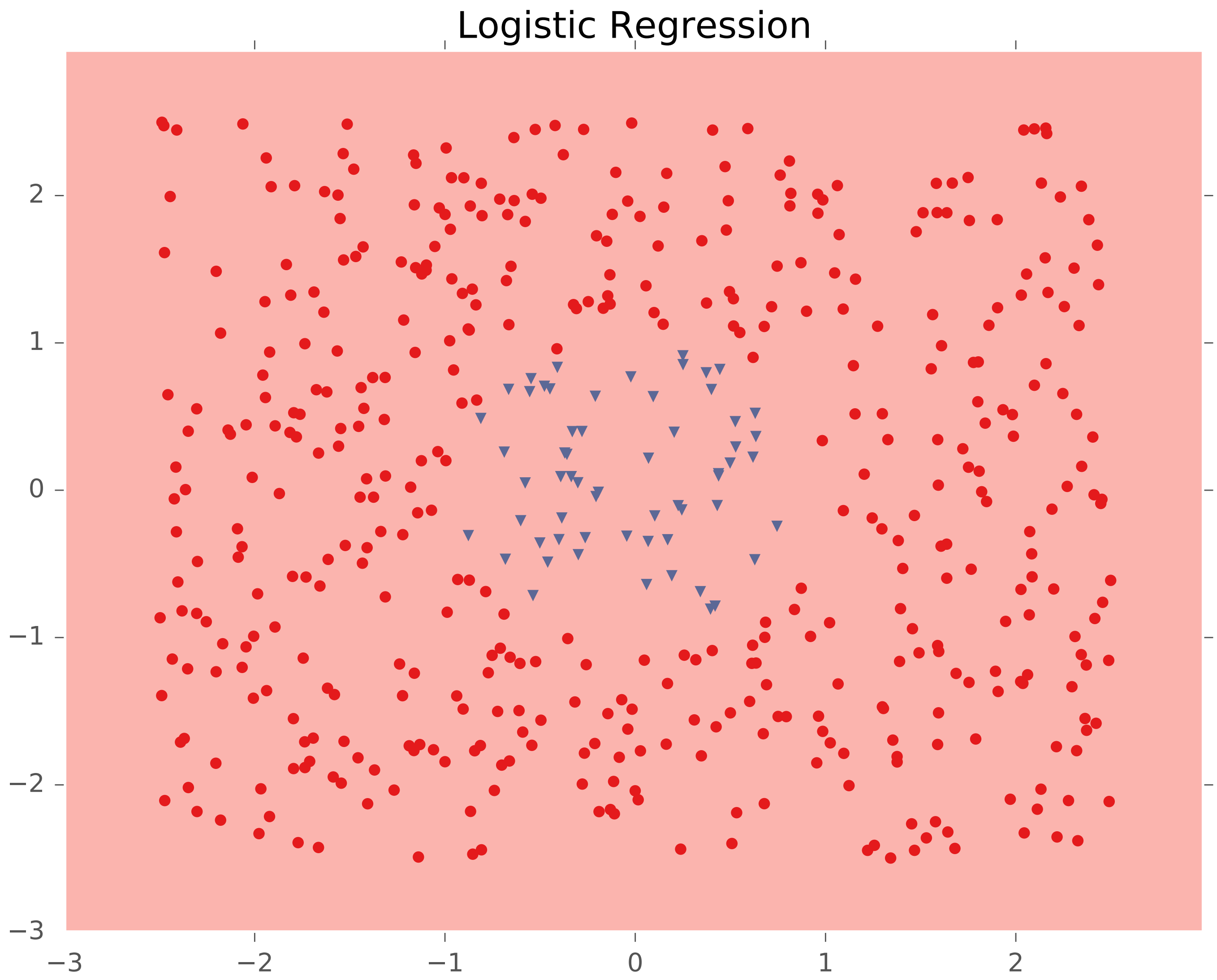




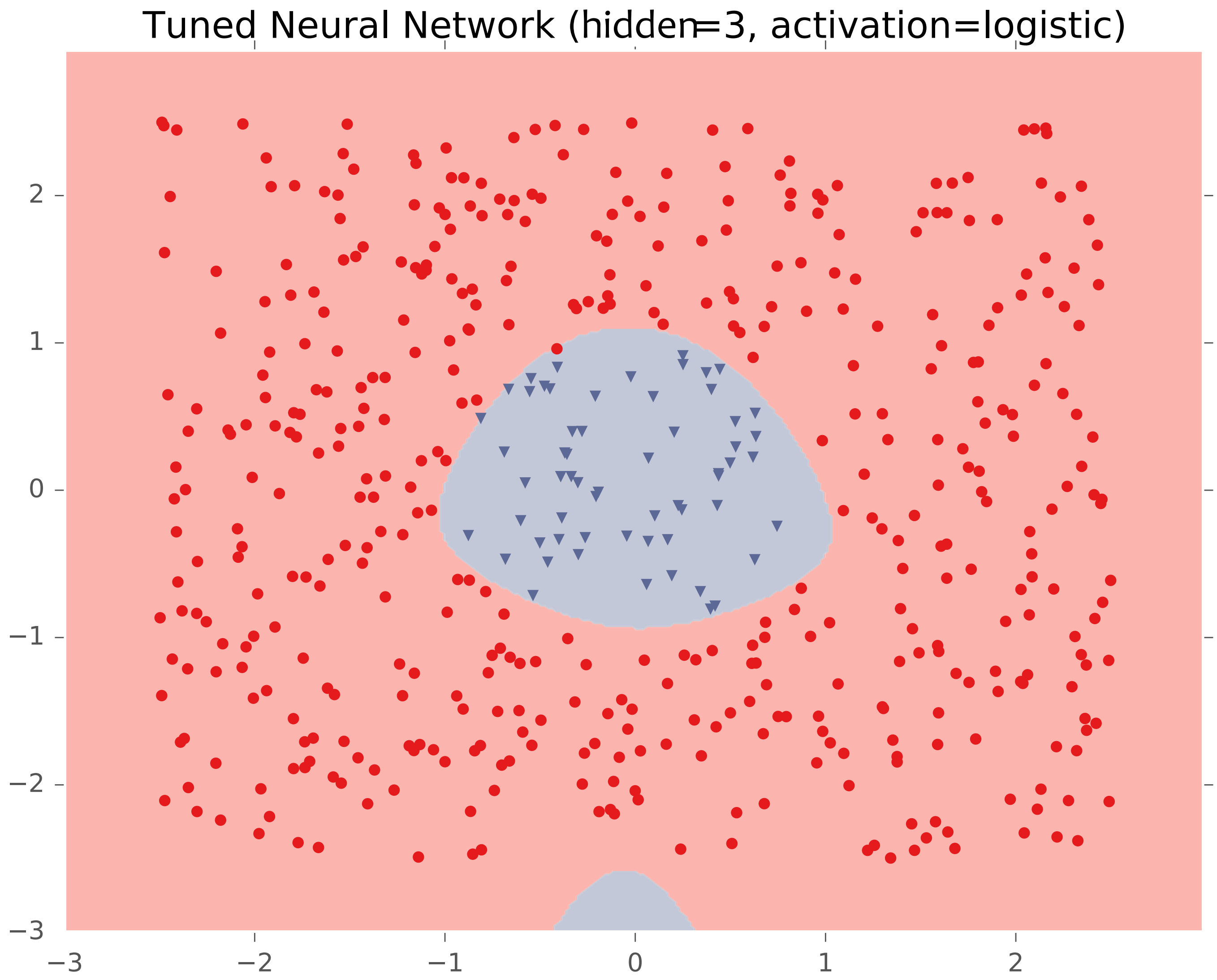
Example #2: One Pocket



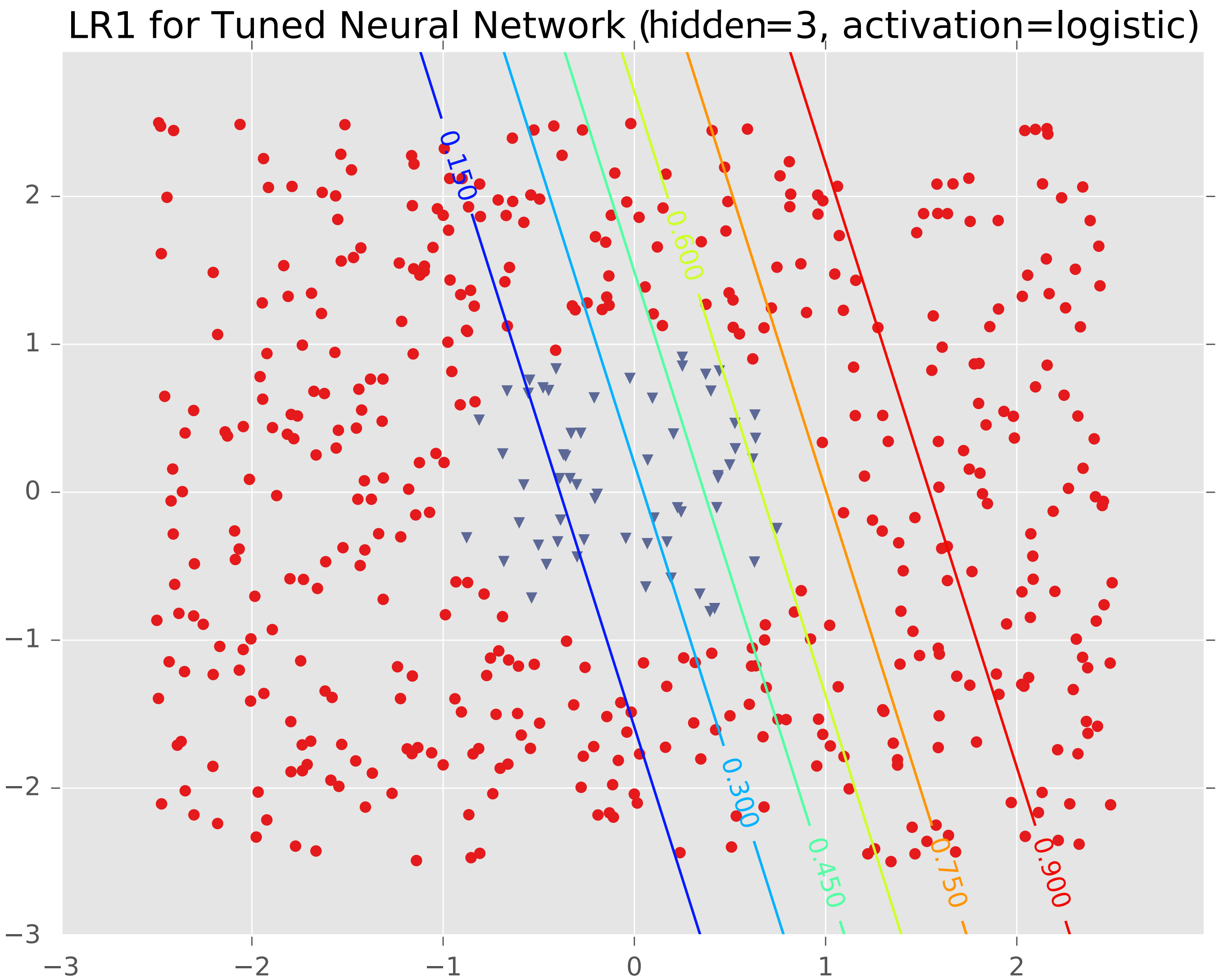
Example #2: One Pocket



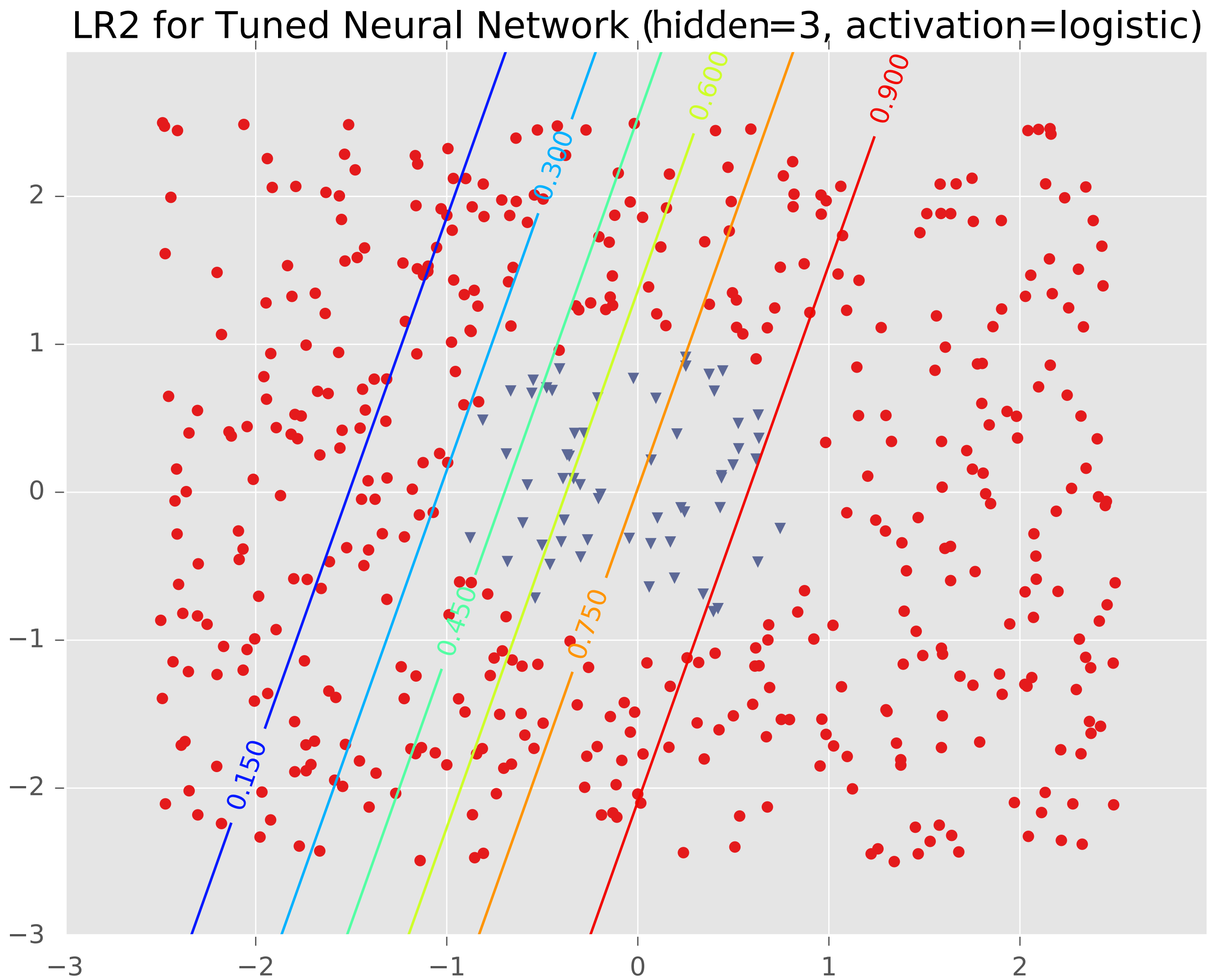
Example #2: One Pocket



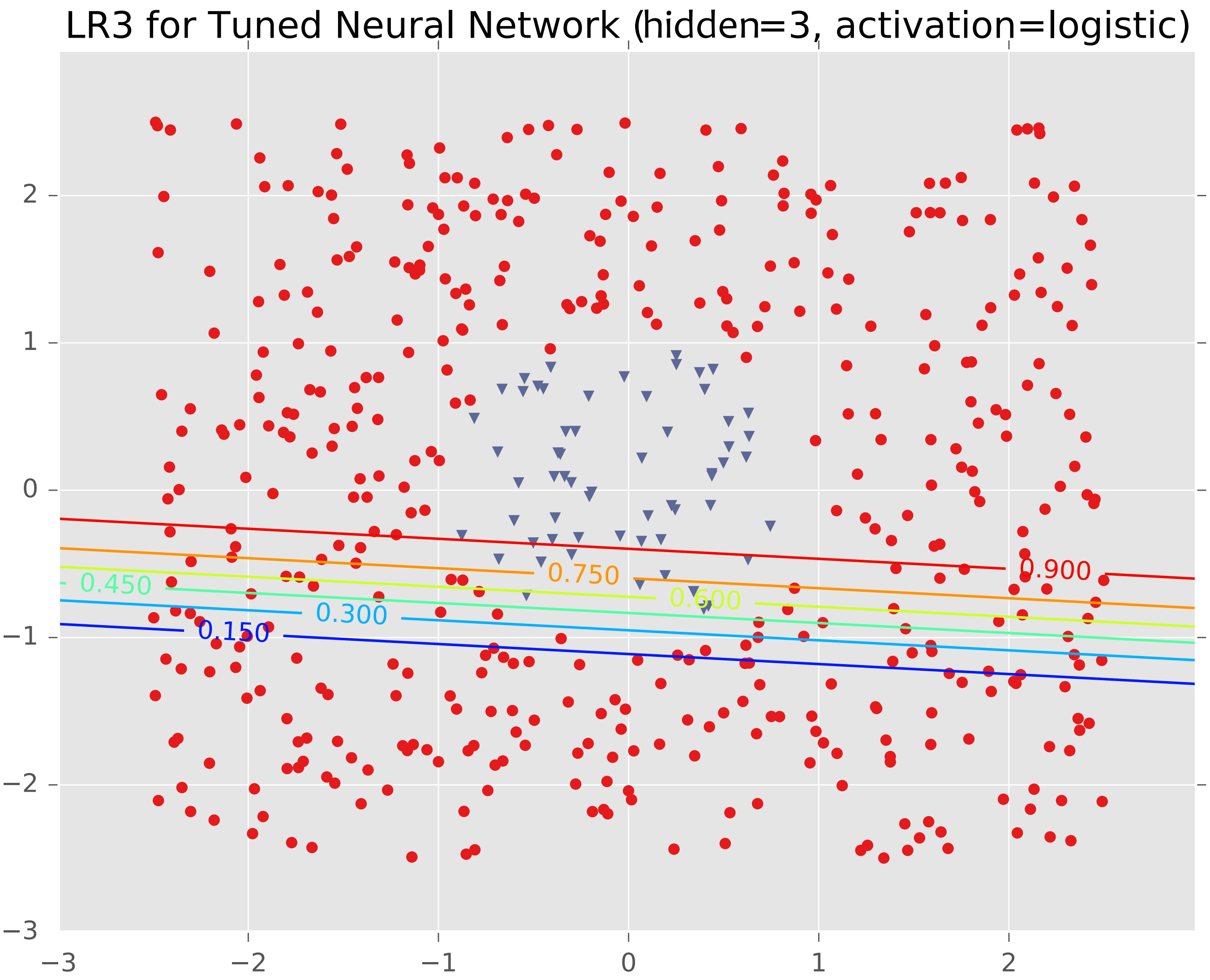
Example #2: One Pocket



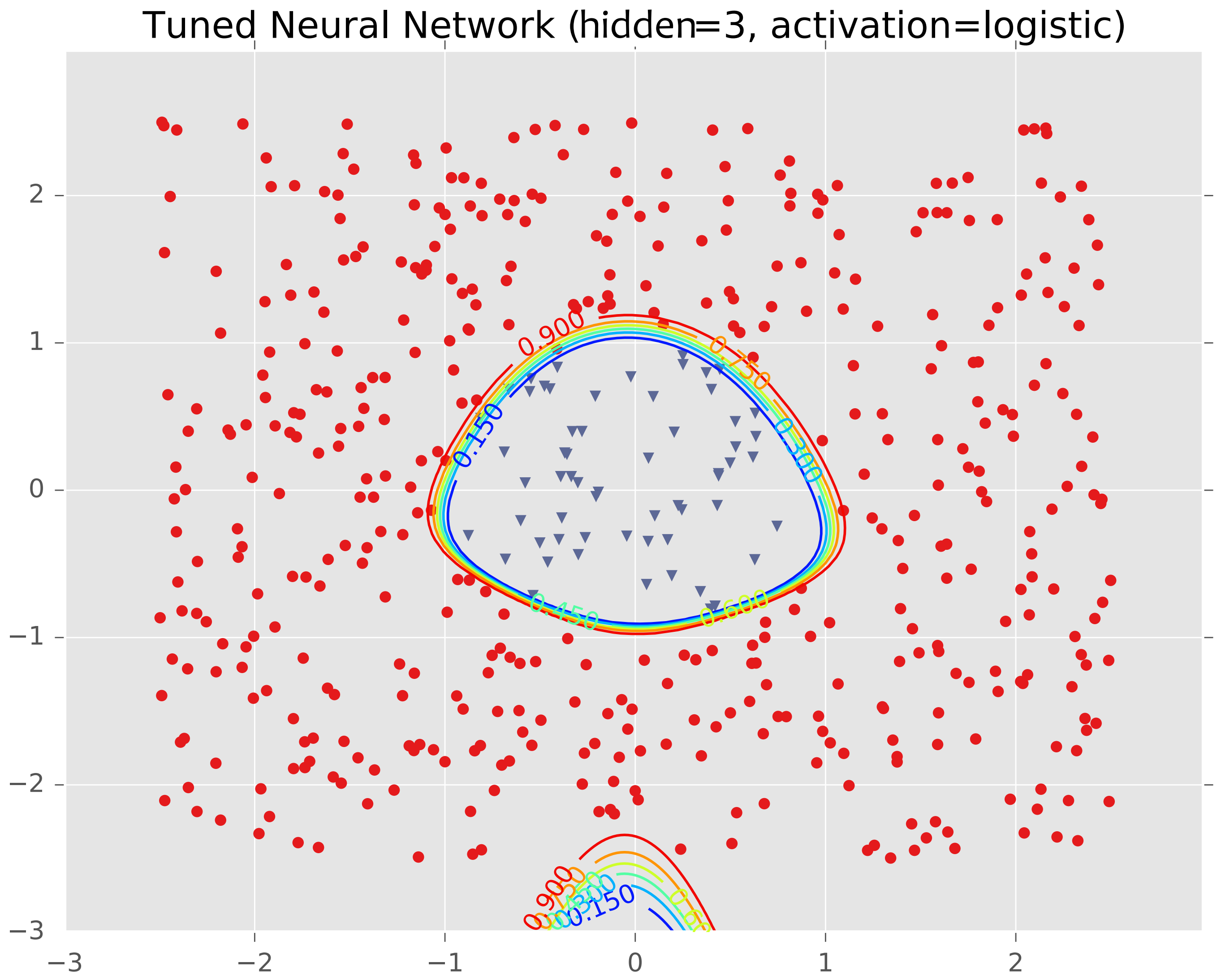
Example #2: One Pocket



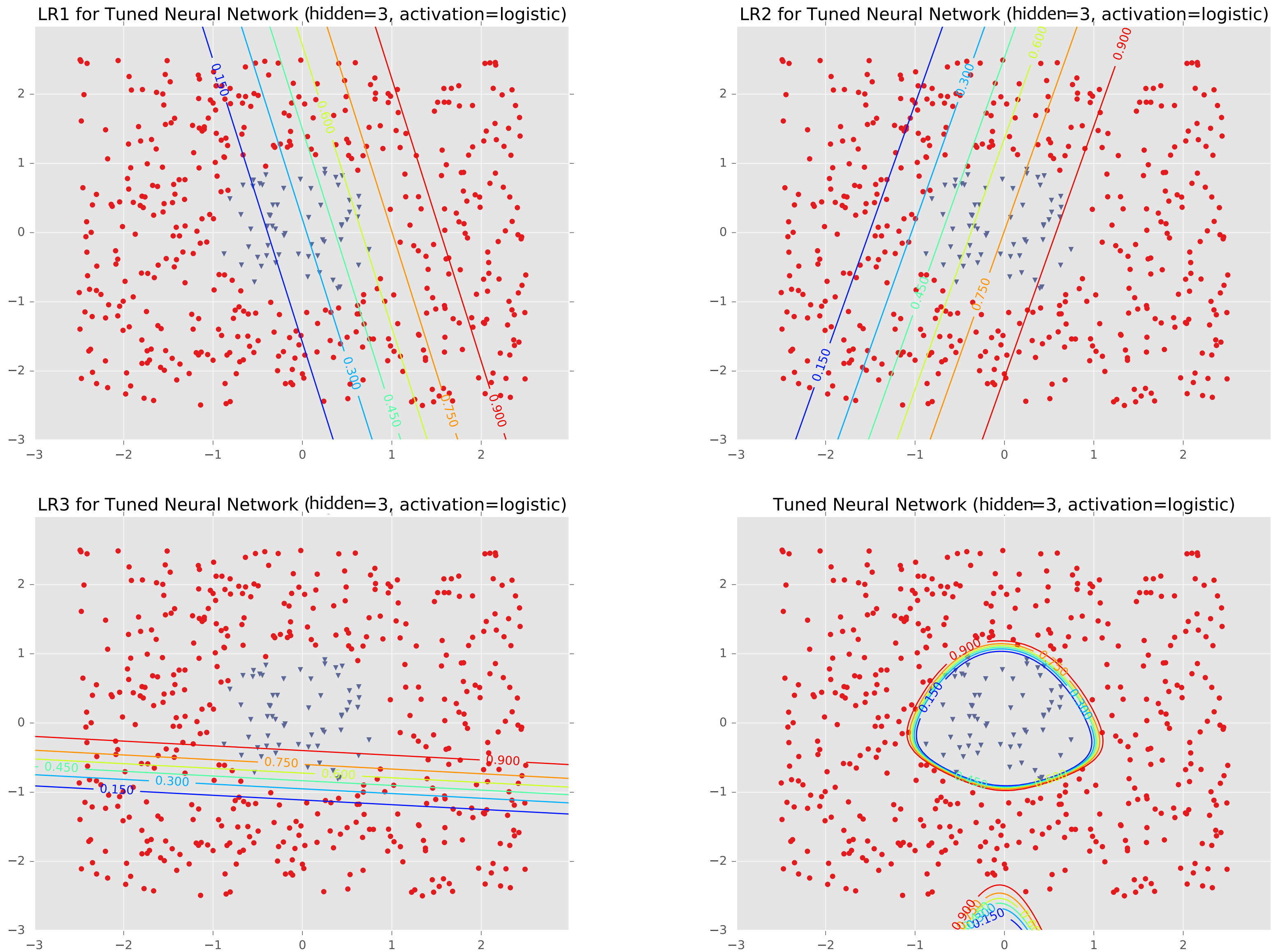
Example #2: One Pocket

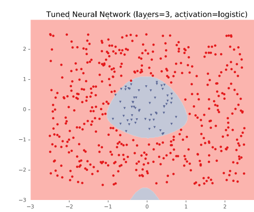


Example #2: One Pocket

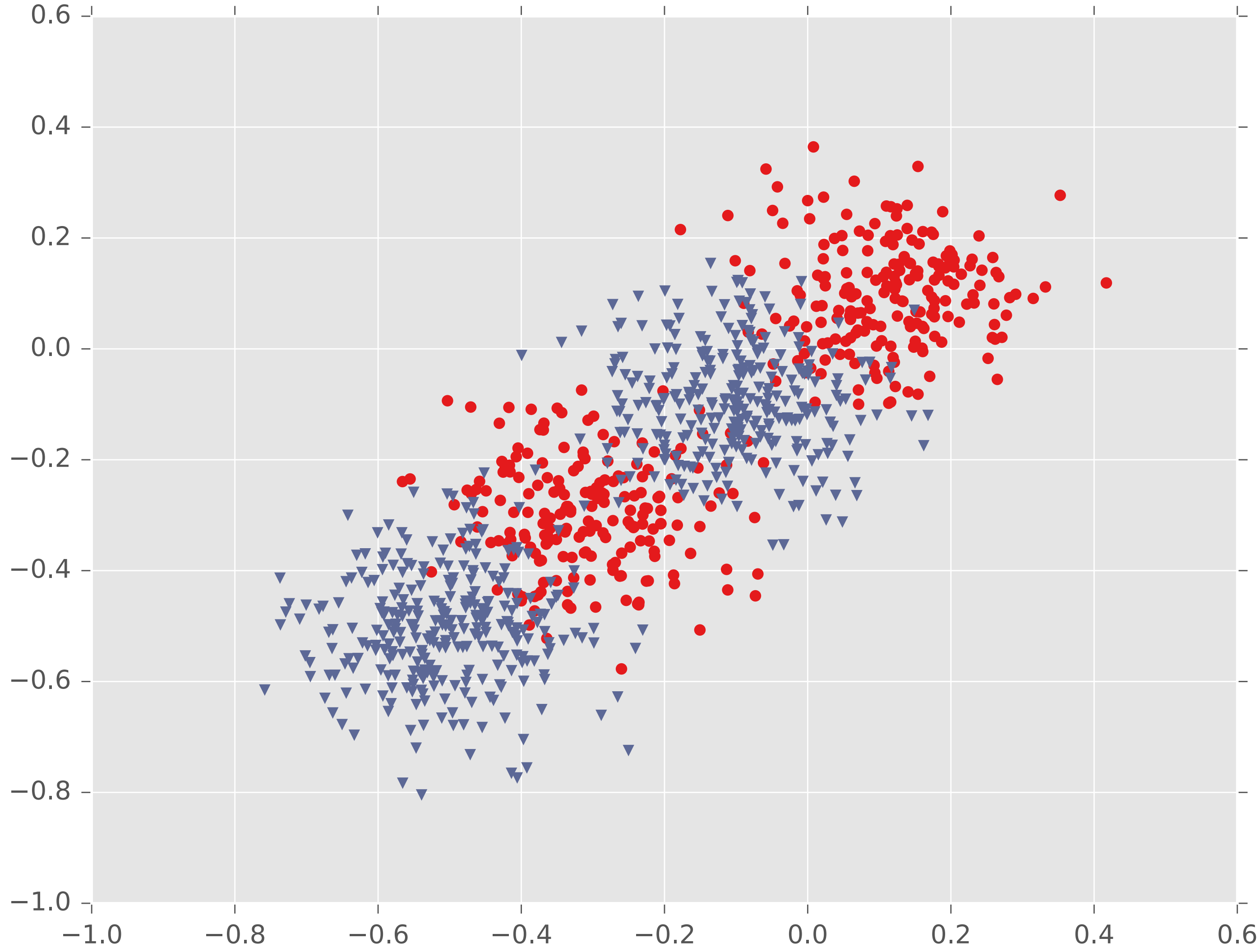


Example #2: One Pocket

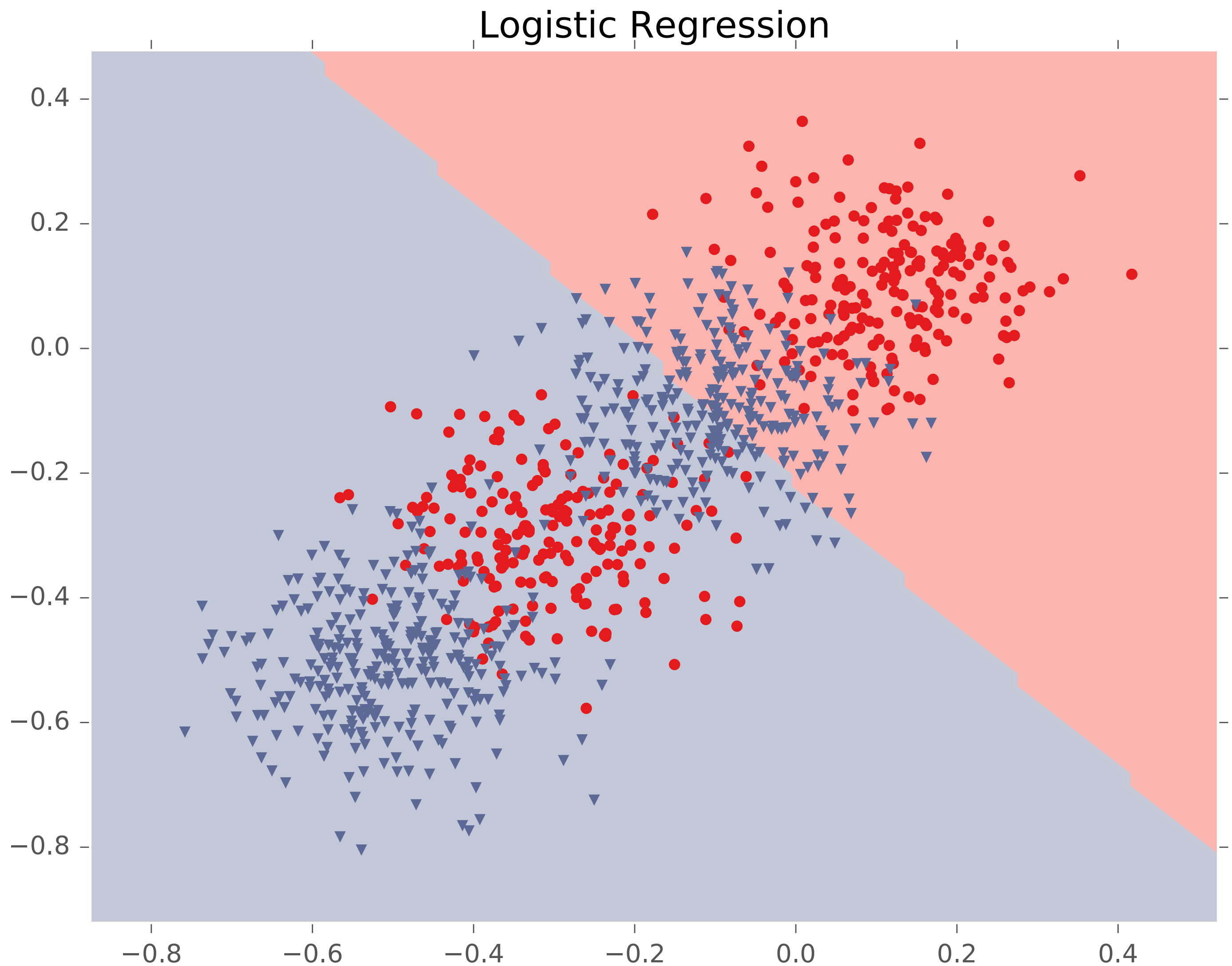




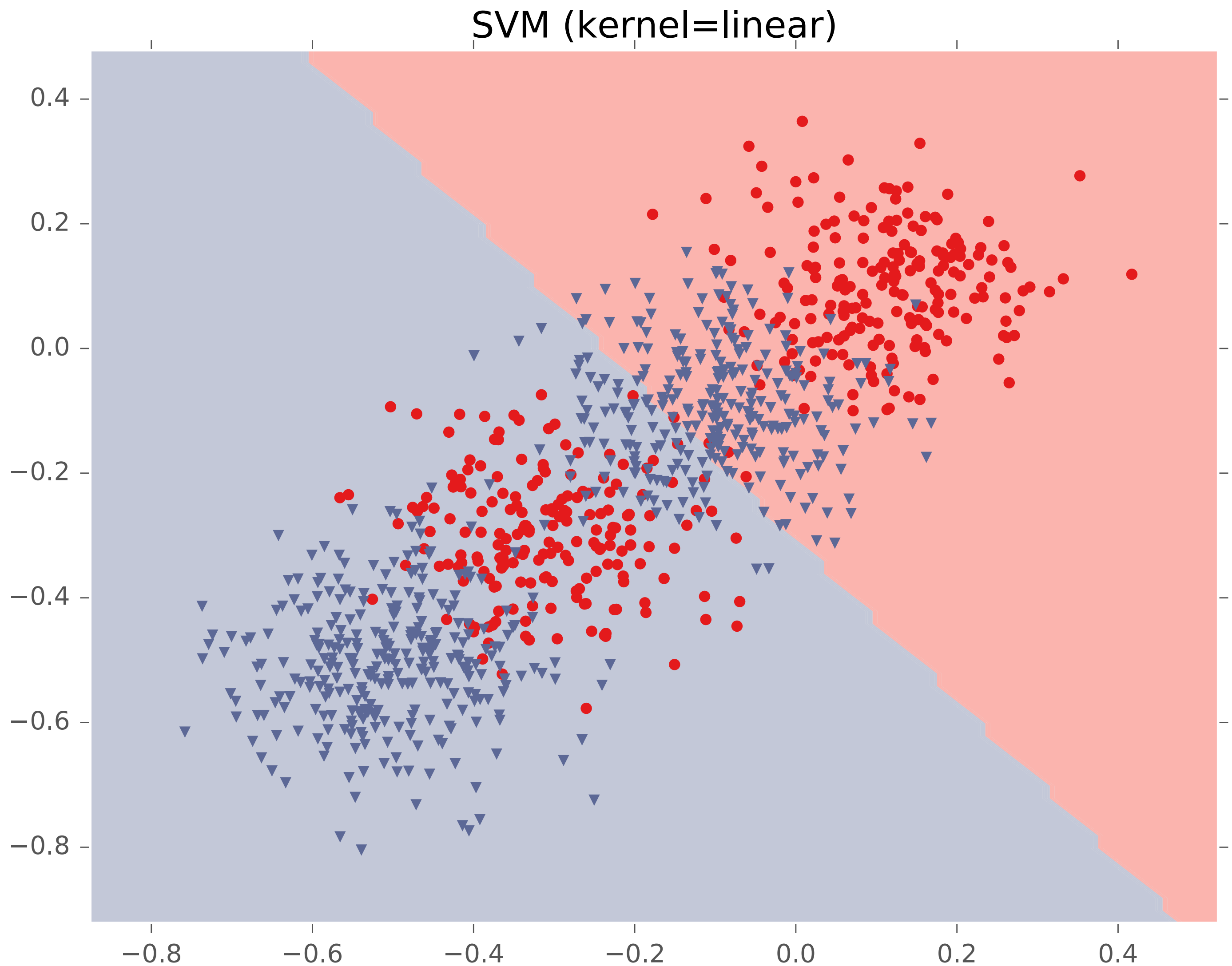
Example #3: Four Gaussians



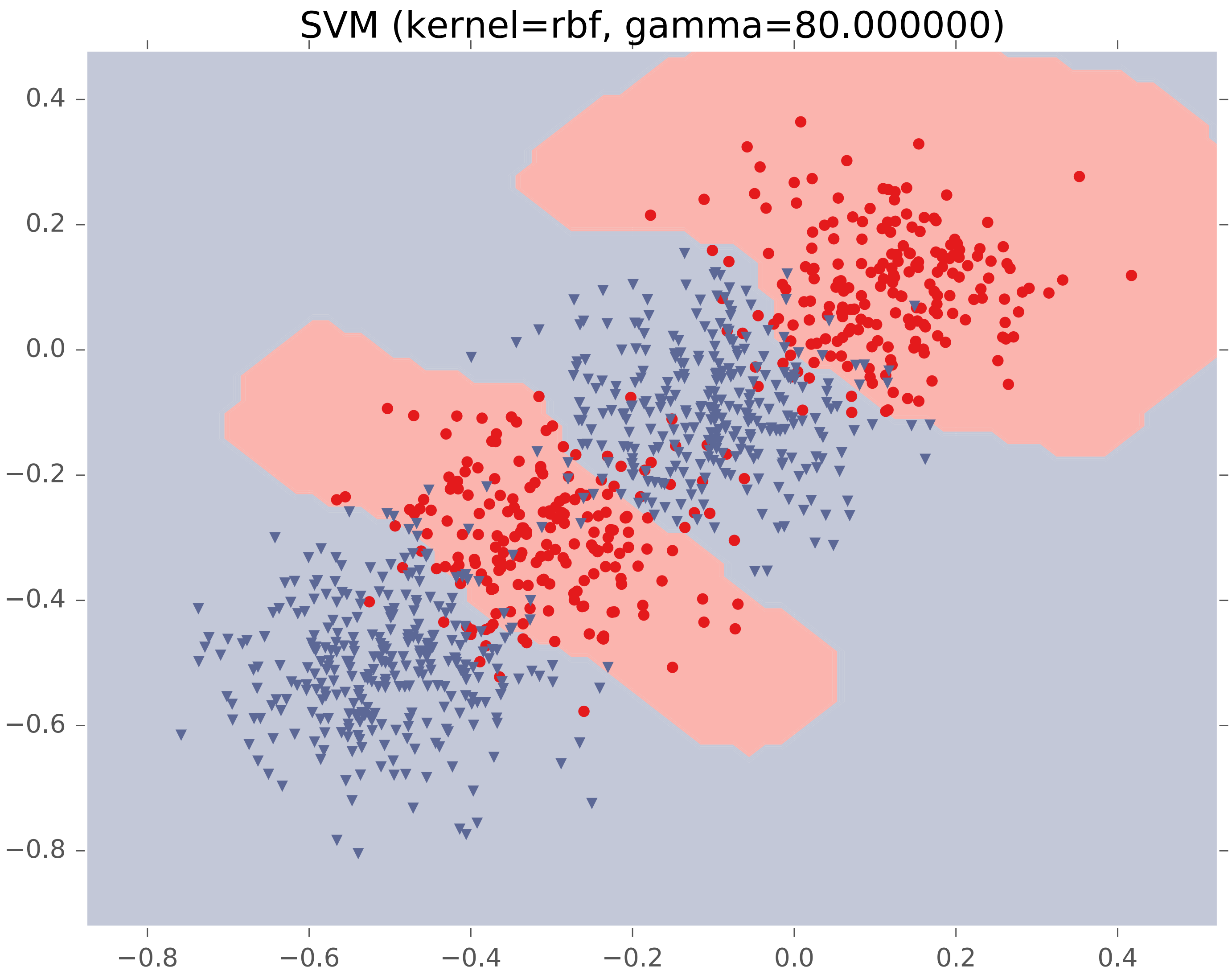
Example #3: Four Gaussians



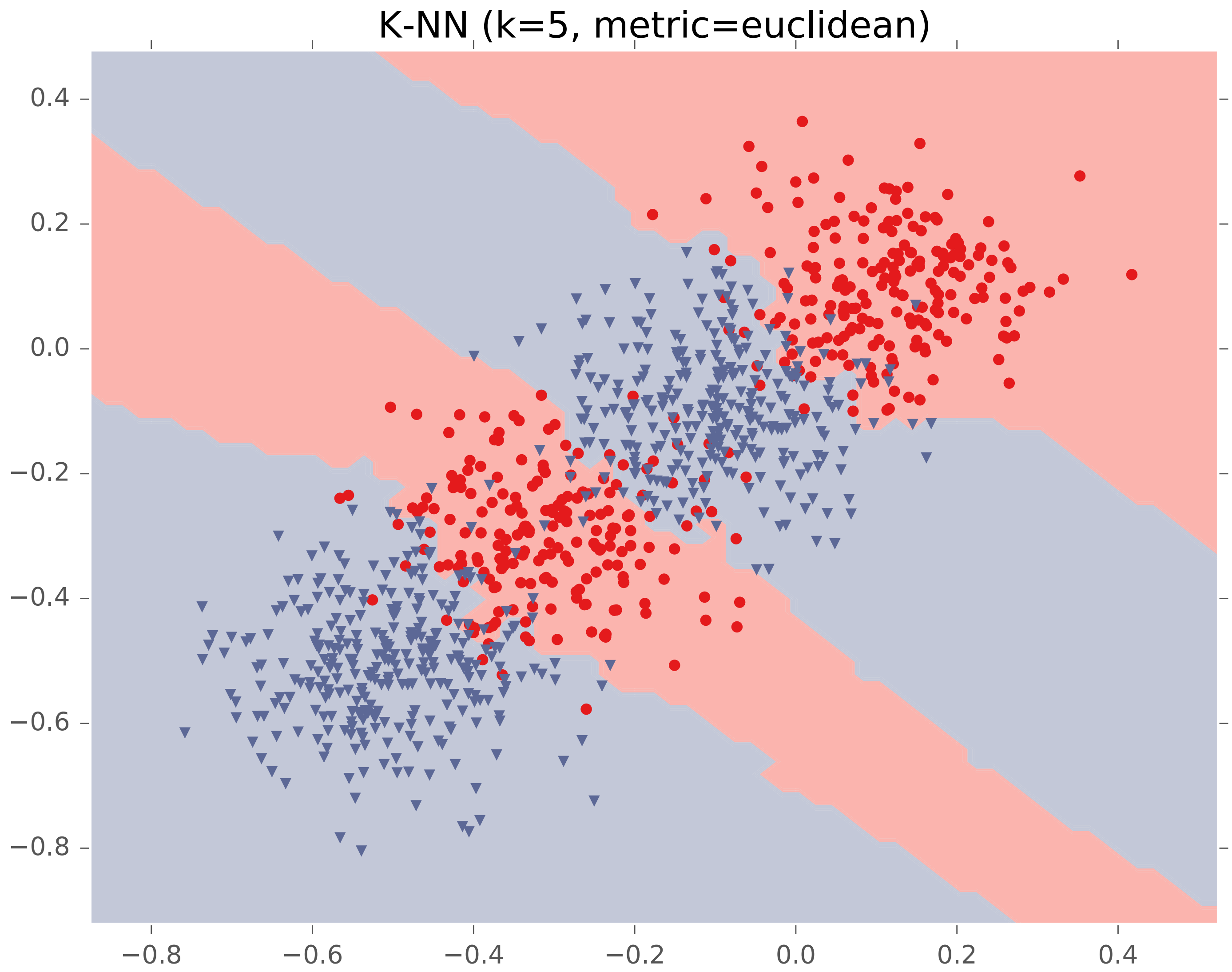
Example #3: Four Gaussians



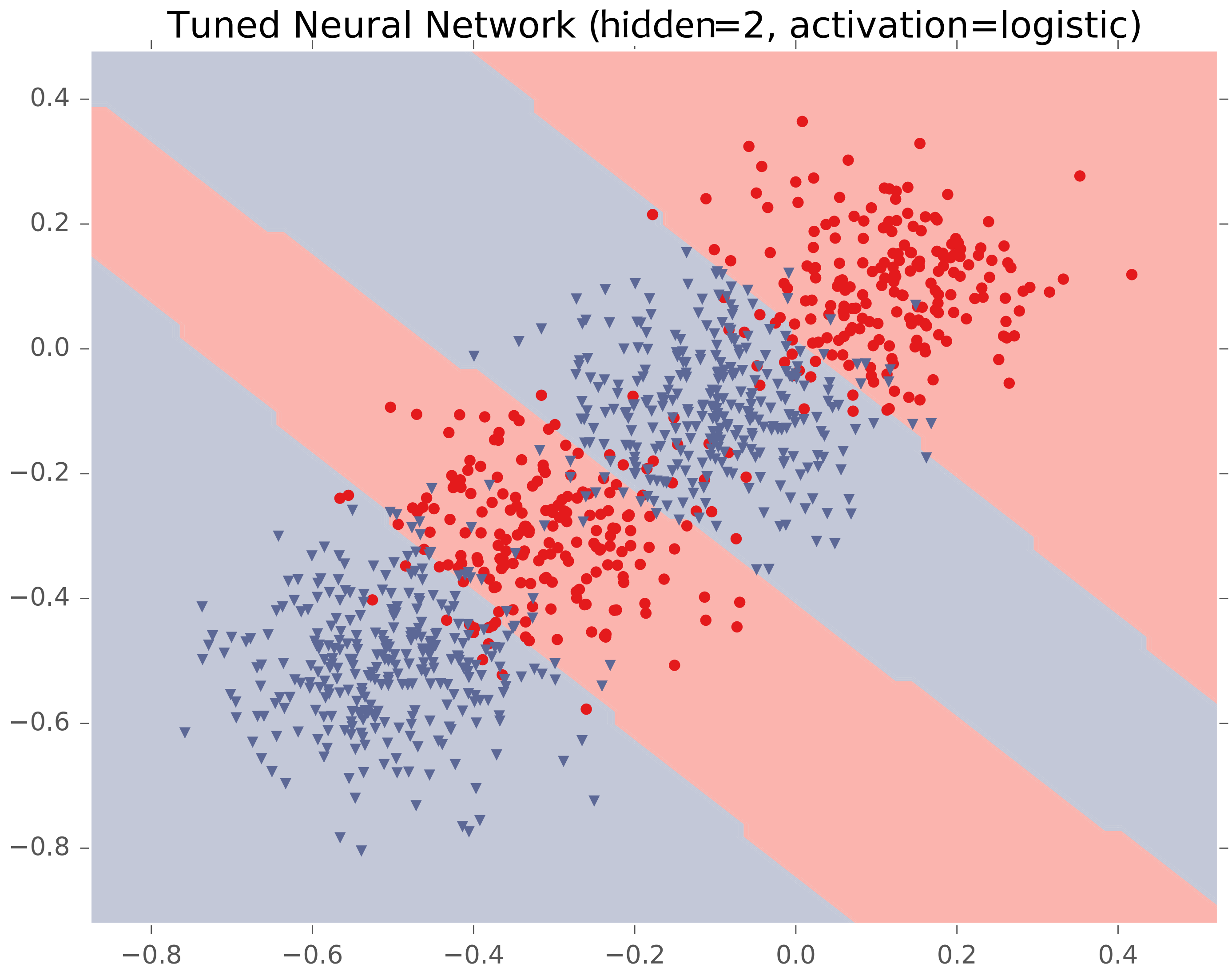
Example #3: Four Gaussians



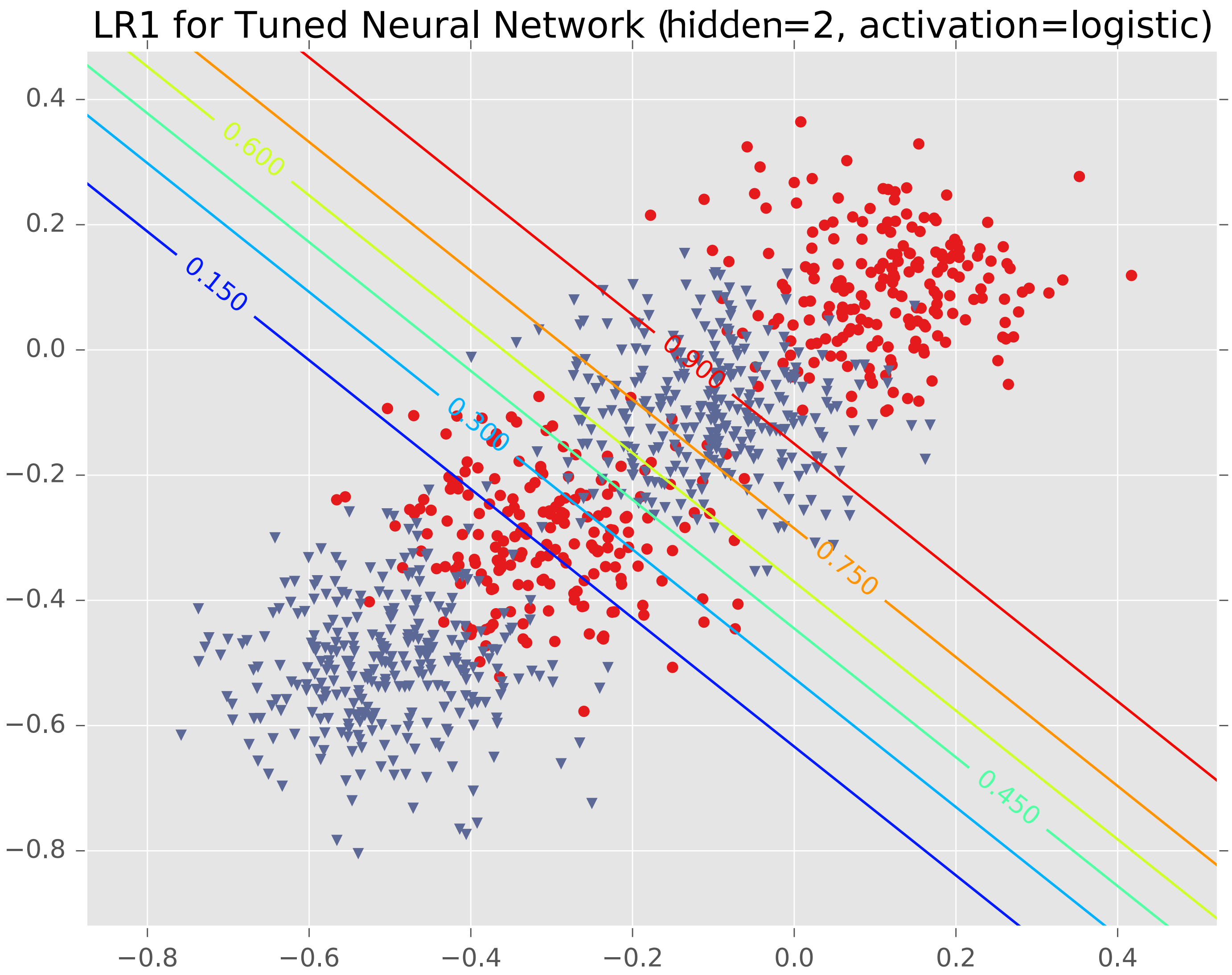
Example #3: Four Gaussians



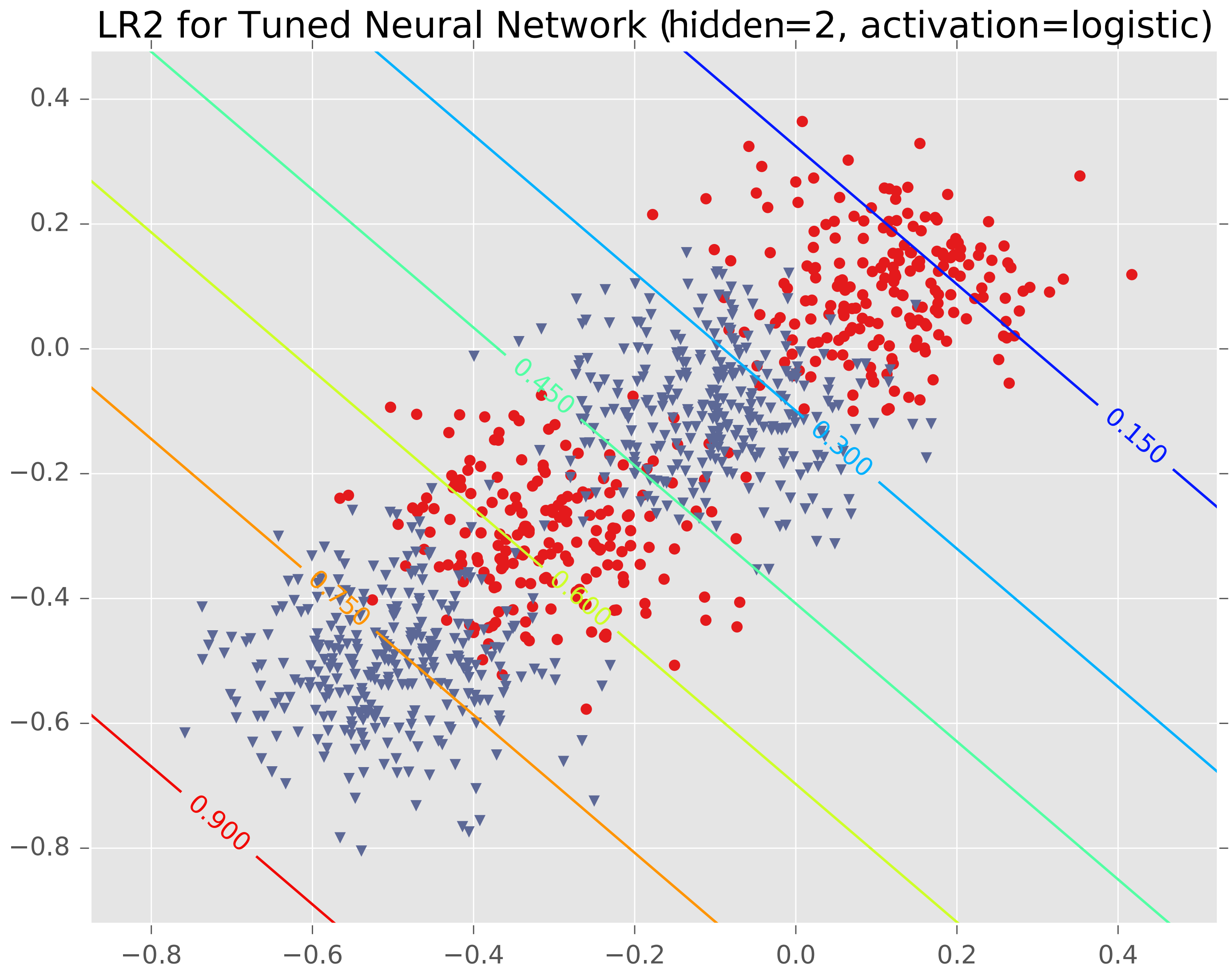
Example #3: Four Gaussians



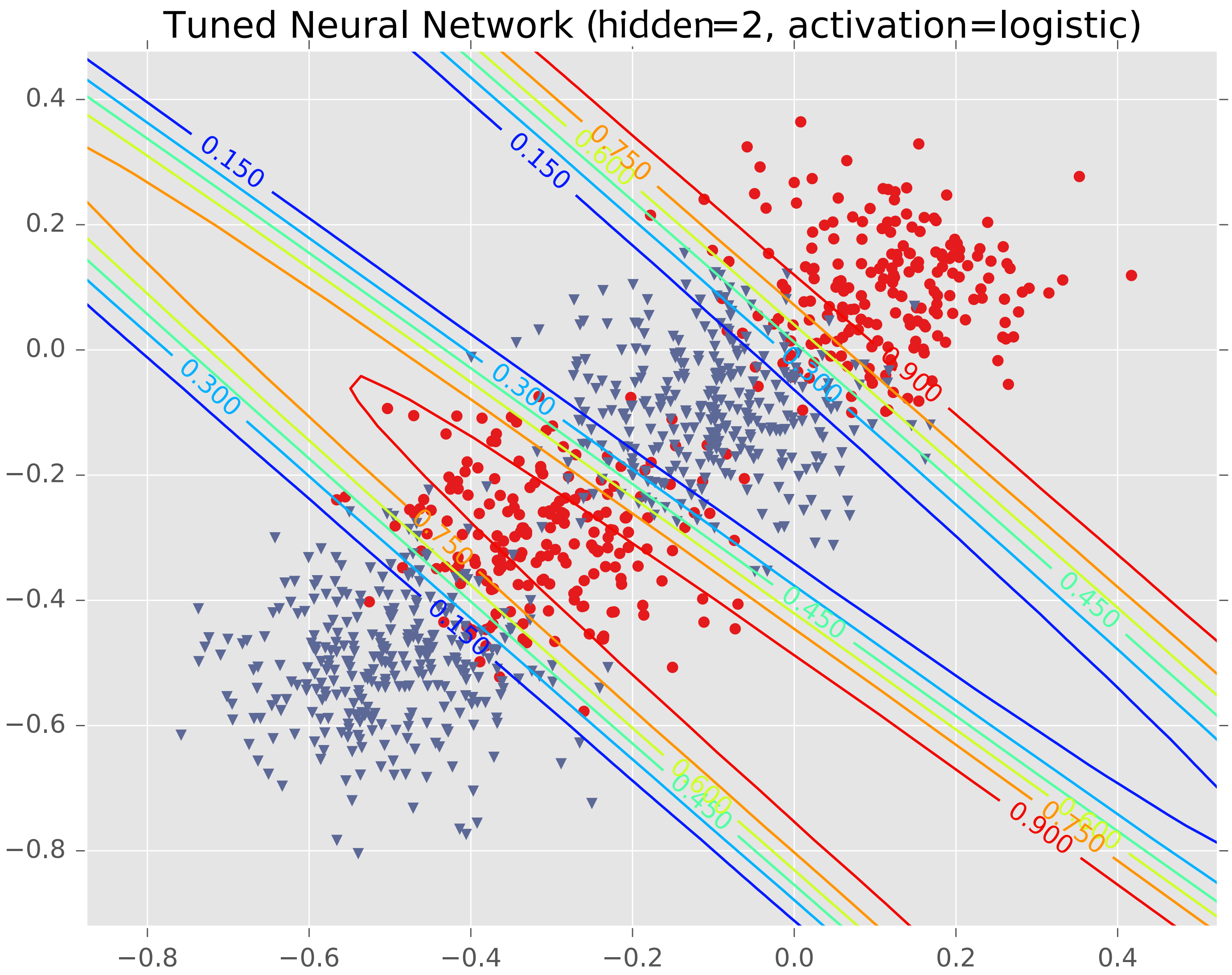
Example #3: Four Gaussians



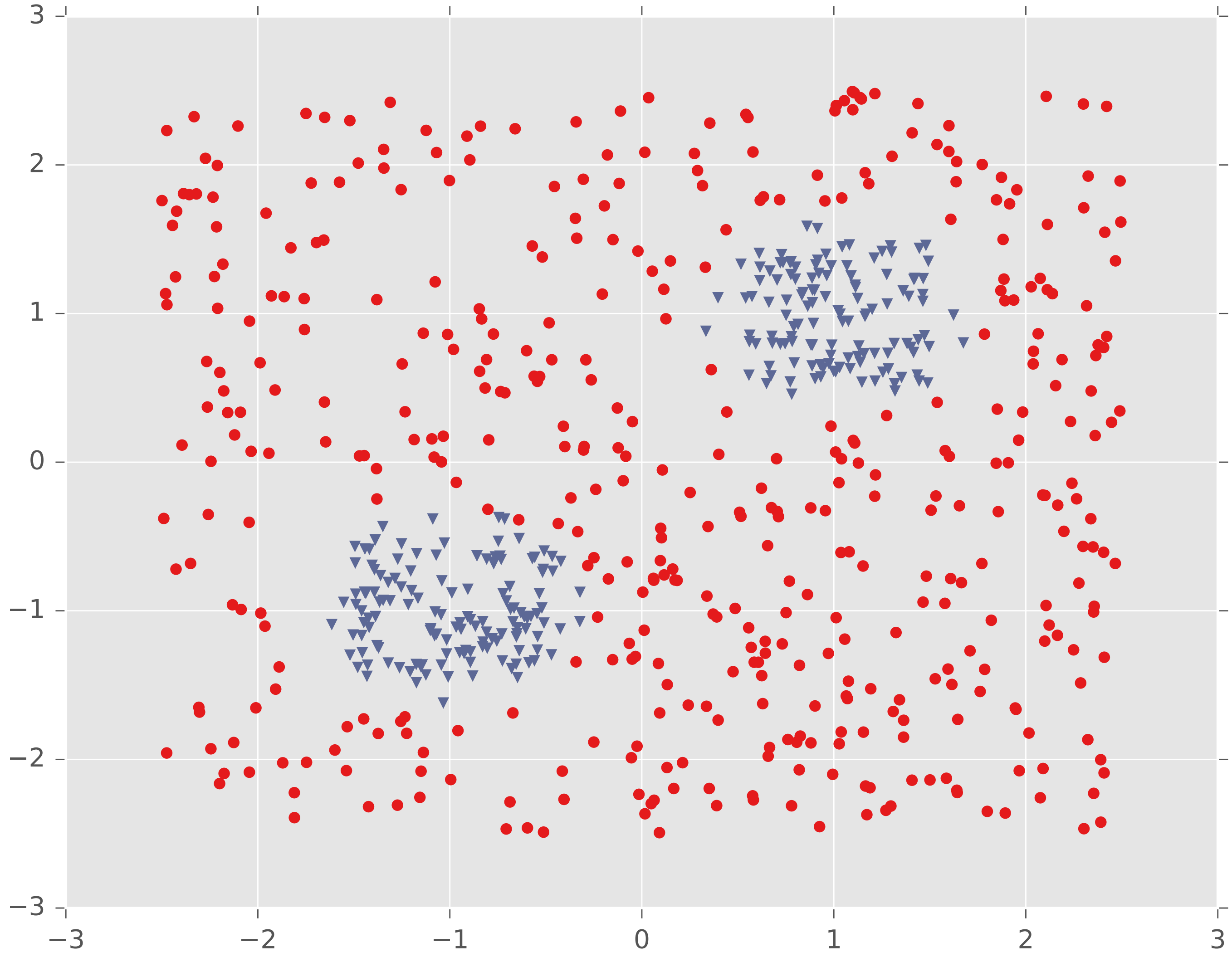
Example #3: Four Gaussians



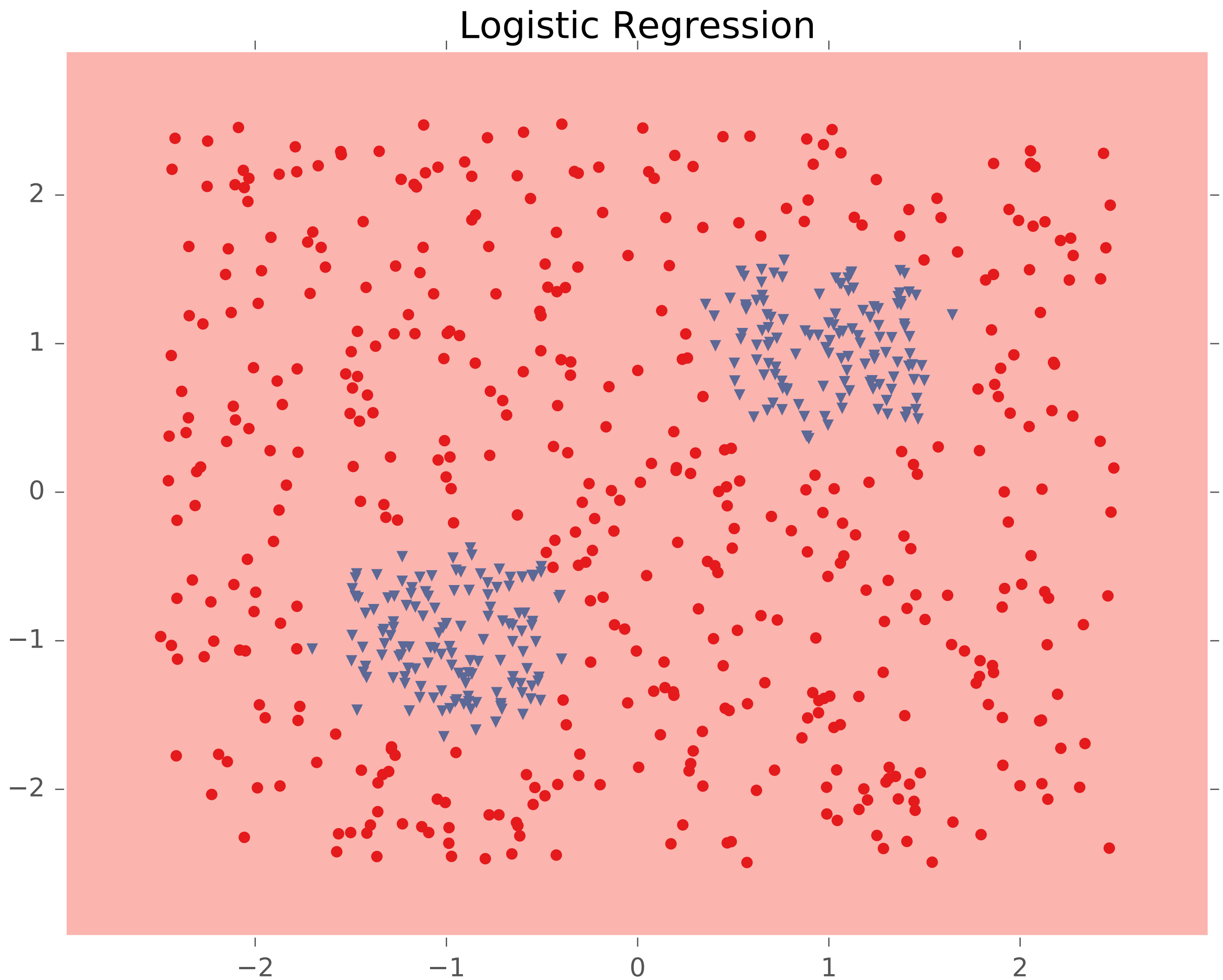
Example #3: Four Gaussians



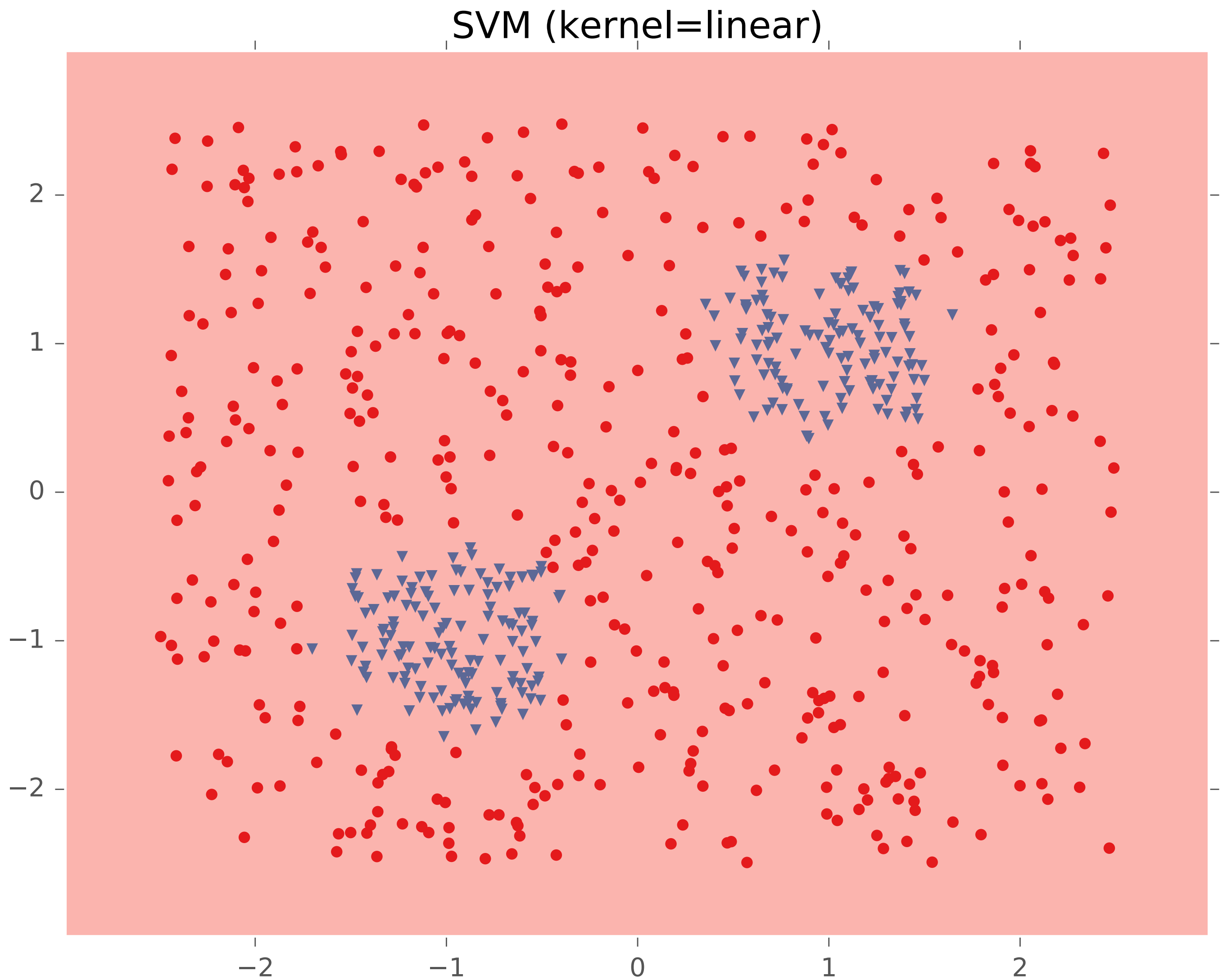
Example #4: Two Pockets



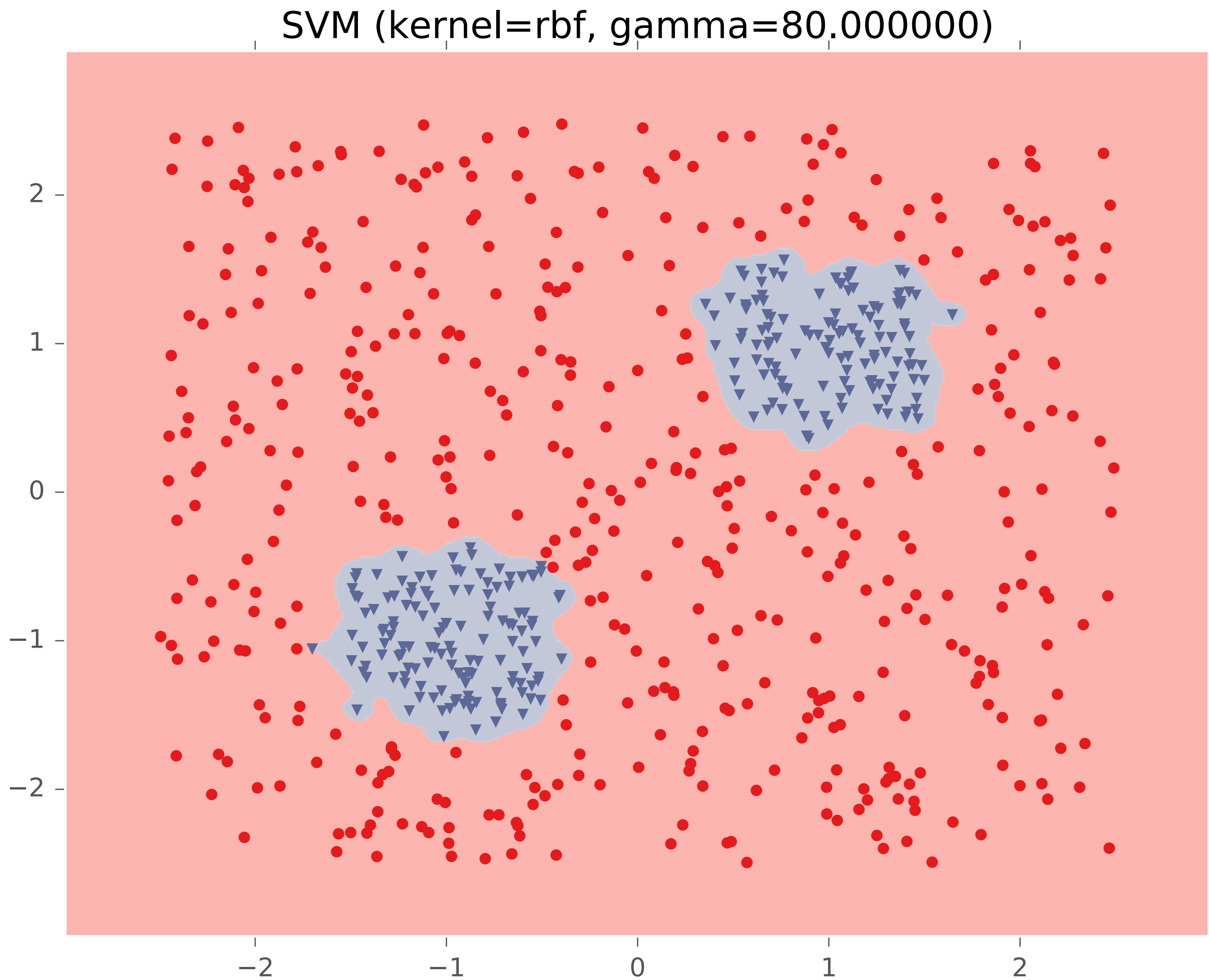
Example #4: Two Pockets



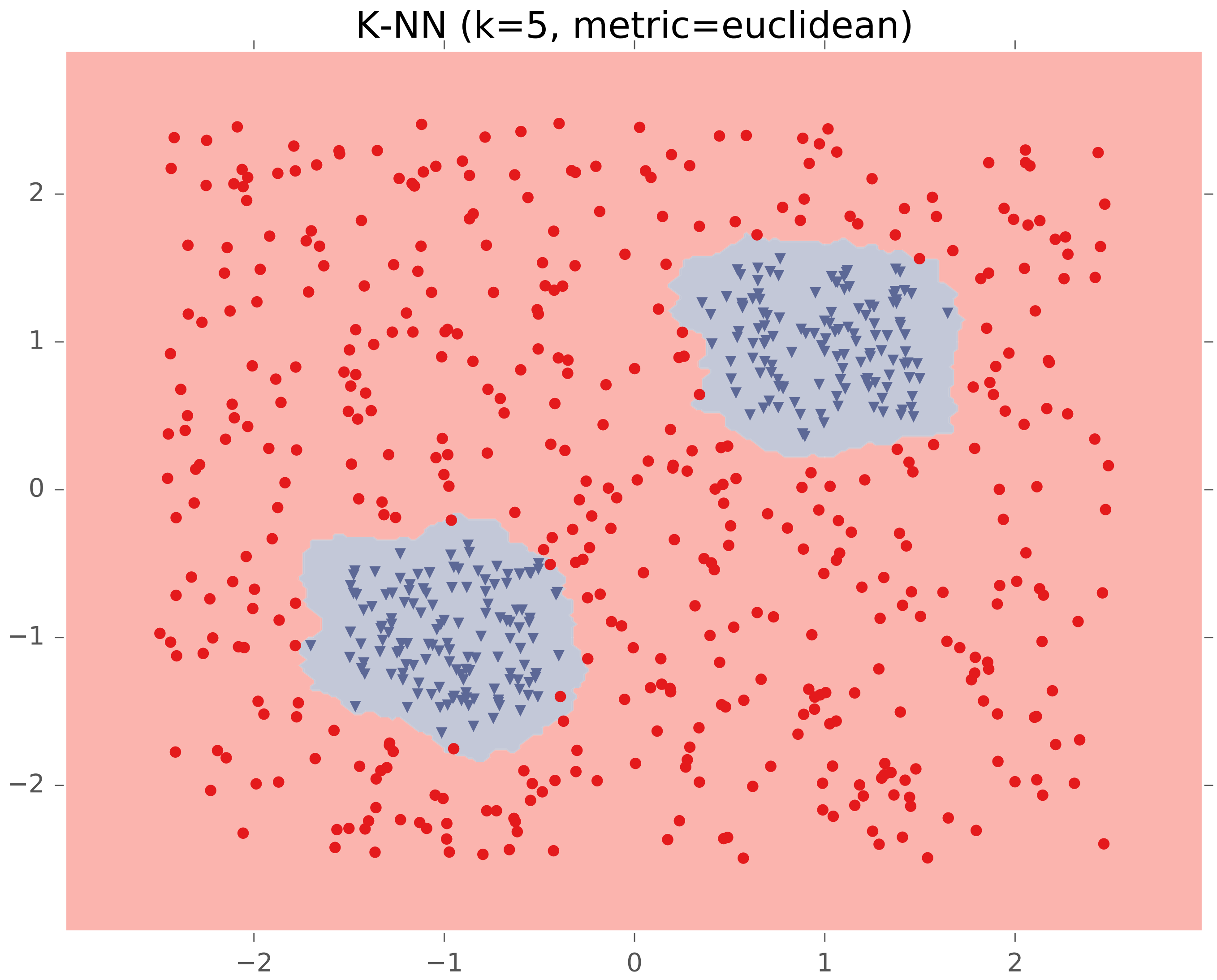
Example #4: Two Pockets



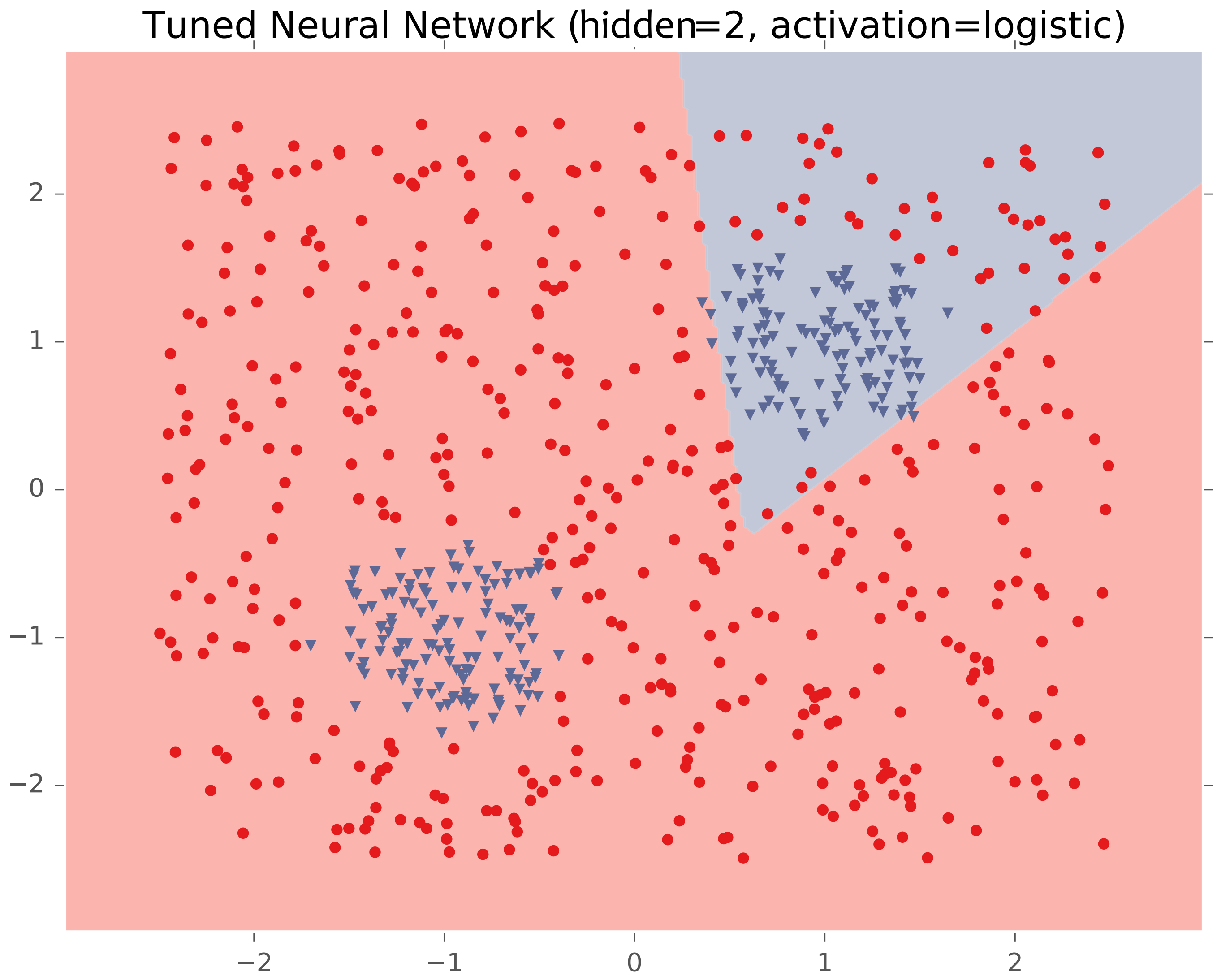
Example #4: Two Pockets



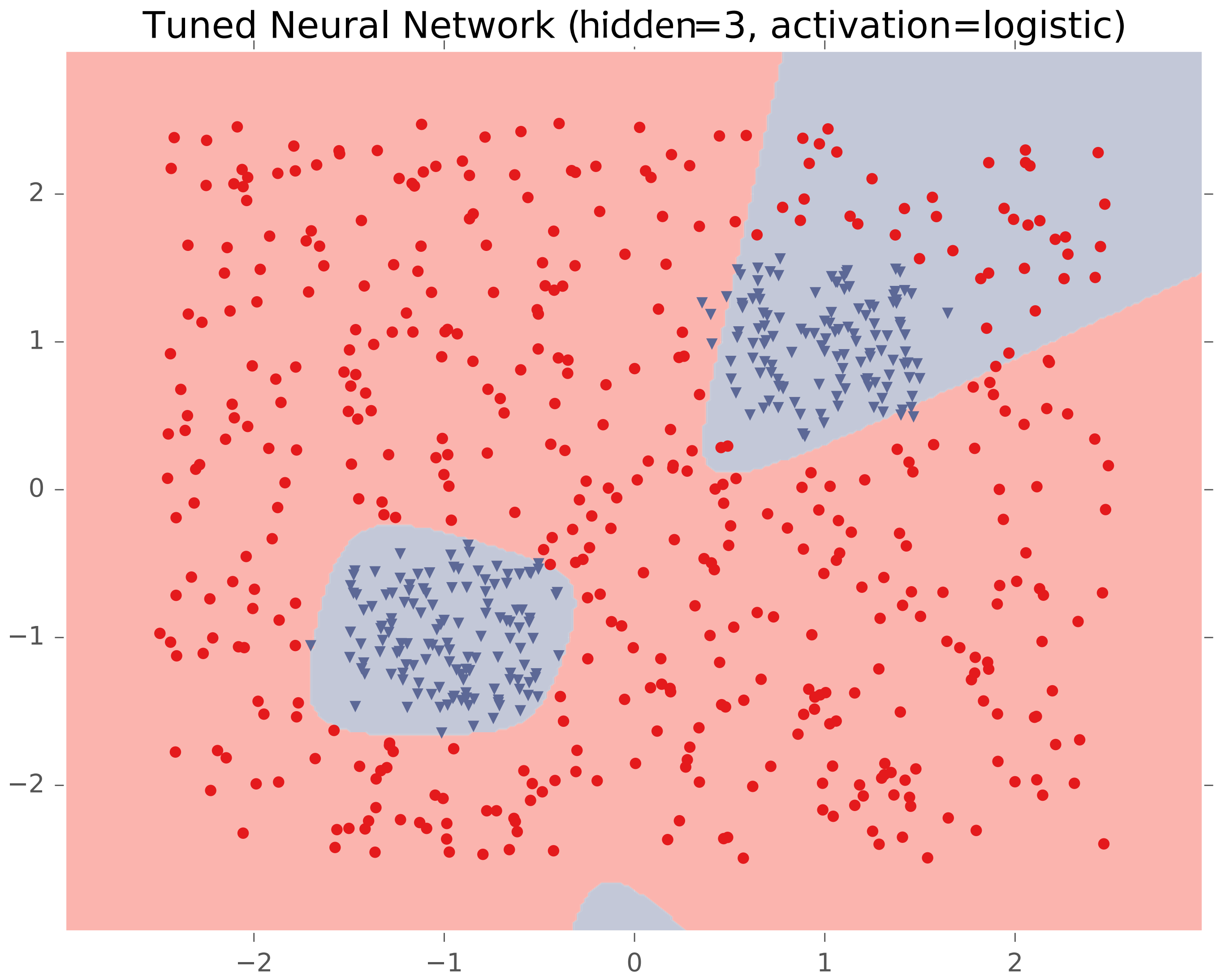
Example #4: Two Pockets



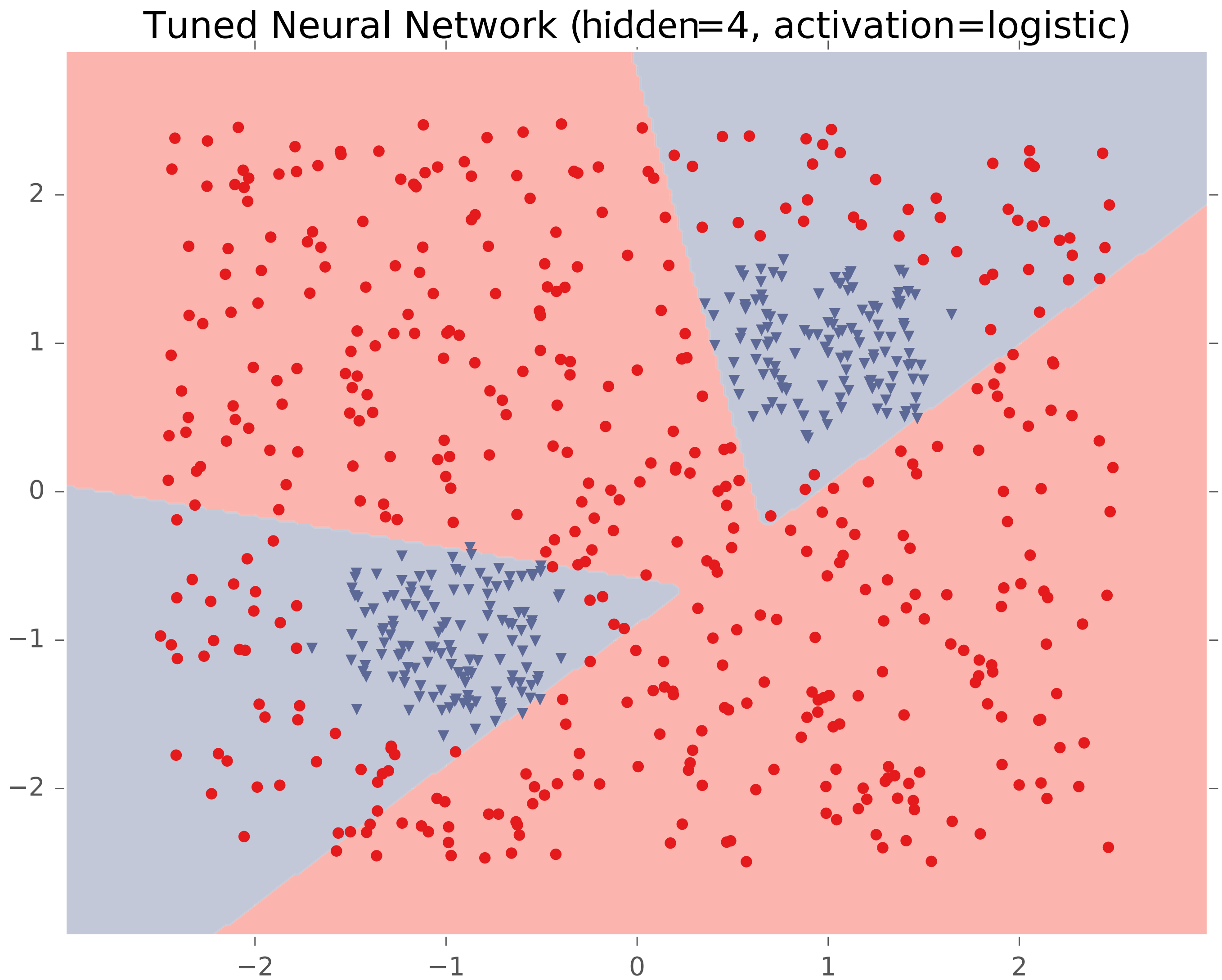
Example #4: Two Pockets



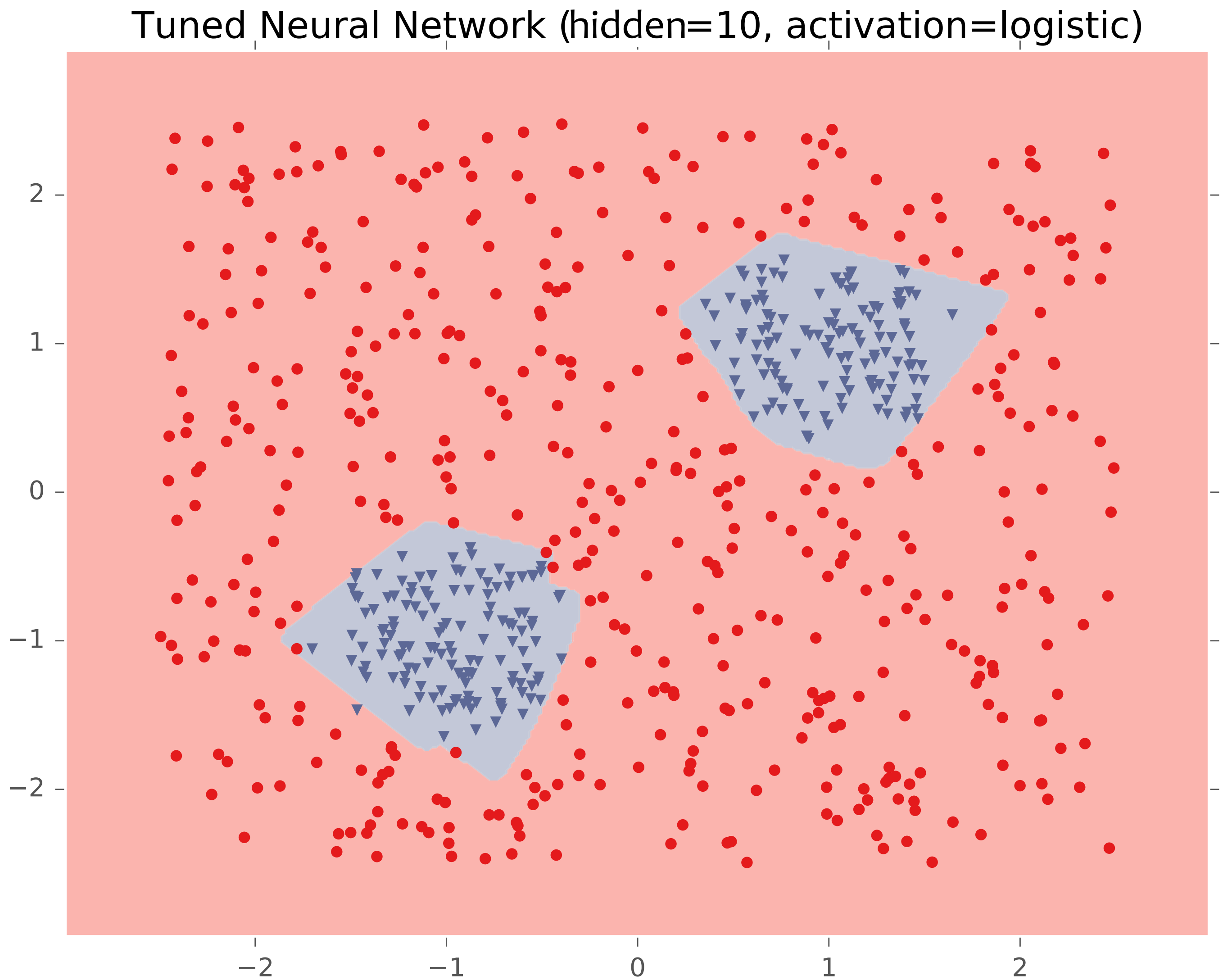
Example #4: Two Pockets



Example #4: Two Pockets



Example #4: Two Pockets



Neural Networks Objectives

You should be able to…

- Explain the biological motivations for a neural network
- Combine simpler models (e.g. linear regression, binary logistic regression, multinomial logistic regression) as components to build up feed-forward neural network architectures
- Explain the reasons why a neural network can model nonlinear decision boundaries for classification
- Compare and contrast feature engineering with learning features
- Identify (some of) the options available when designing the architecture of a neural network
- Implement a feed-forward neural network