# CX4240 Spring 2026
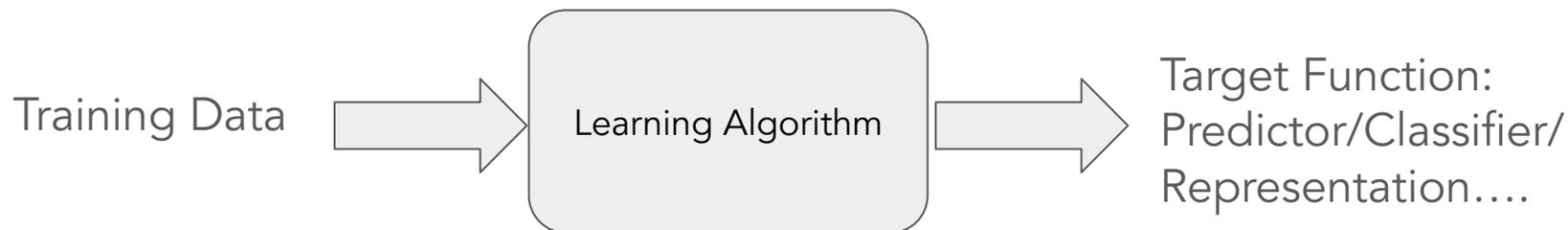# Multi-Class Logistic Regression Naive Bayes

Bo Dai
School of CSE, Georgia Tech
bodai@cc.gatech.edu
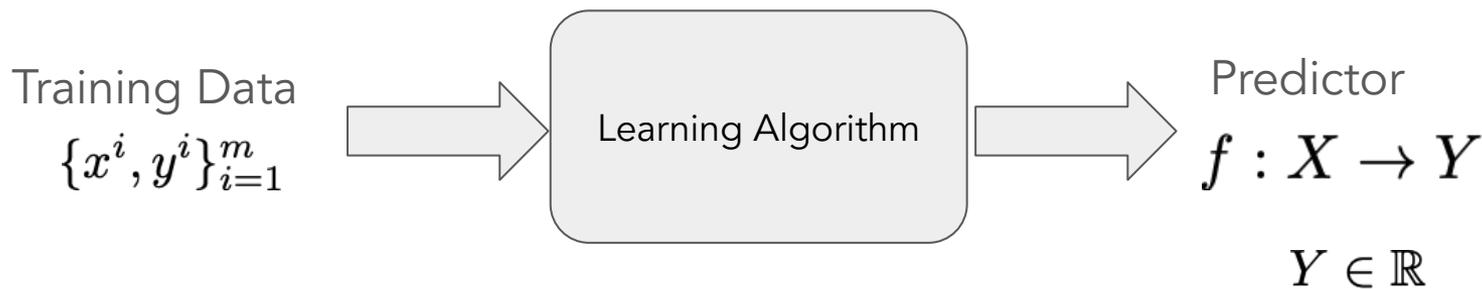
# ML Algorithm Pipeline

Training Data → Learning Algorithm → Target Function: Predictor/Classifier/Representation….

## General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP….)
3. Select optimizer

# Regression Algorithms

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Predictor
$$f : X \rightarrow Y$$

$$Y \in \mathbb{R}$$

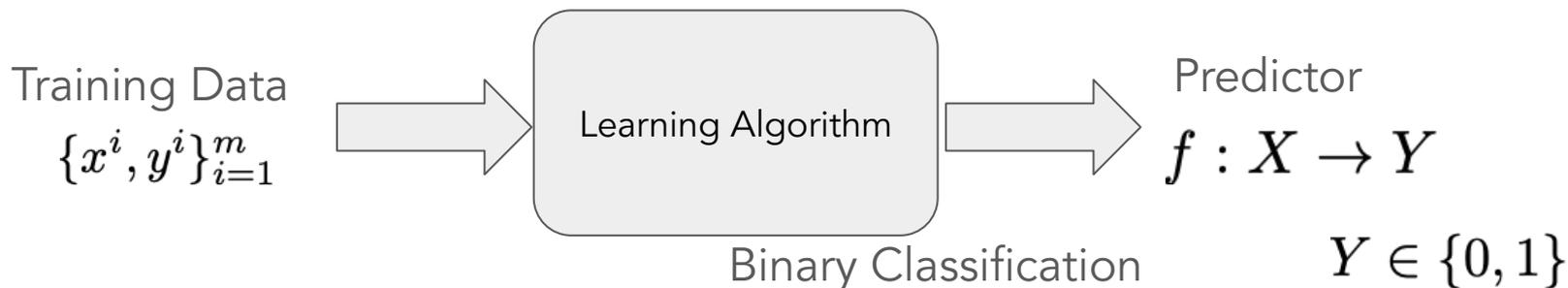## Linear Regression Pipeline

1. Build probabilistic models:
   Gaussian Distribution + Linear Model
2. Derive loss function: MLE and MAP
3. Select optimizer
   Necessary Condition vs. (Stochastic) GD

# Binary Classification Algorithms

Training Data
$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Binary Classification

Predictor
$f : X \to Y$

$Y \in \{0, 1\}$

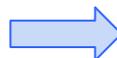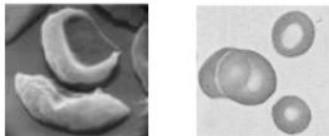## Binary Logistic Regression Pipeline

1. Build probabilistic models:
   Bernoulli Distribution + + Linear Model
2. Derive loss function: MLE and MAP
3. Select optimizer: (Stochastic) Gradient Descent

# Classification Tasks

**Feature, X**                    **Label, Y**

Diagnosing sickle cell anemia

Anemic cell
Healthy cell     $Y \in \{0, 1\}$

Tax Fraud Detection

Cheat
?     $Y \in \{0, 1\}$
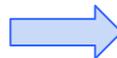
Web Classification

Sports
Science
News     $Y \in \{0, 1, 2, \ldots, k\}$

Image Classification

Airplane
Automobile
Bird
…     $Y \in \{0, 1, 2, \ldots, k\}$

# Multiclass Logistic Regression Algorithms
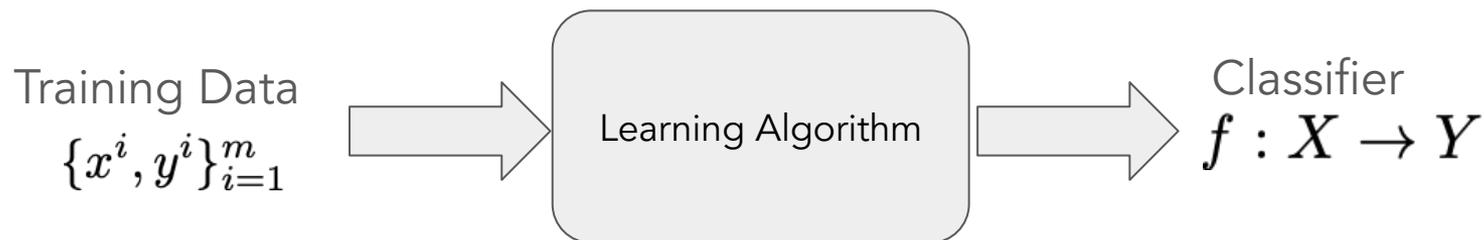
Training Data
$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Classifier
$f : X \to Y$

# Multiclass Logistic Regression Algorithms

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Classifier
$$f : X \to Y$$

Multiclass Classification

$$Y \in \{0, 1, \dots, k\}$$



airplane
automobile
bird
cat
deer
dog
frog
horse
ship
truck

# Multiclass Logistic Regression Algorithms

Training Data
$$\{x^i, y^i\}_{i=1}^{m}$$

Learning Algorithm

Classifier
$$f : X \to Y$$

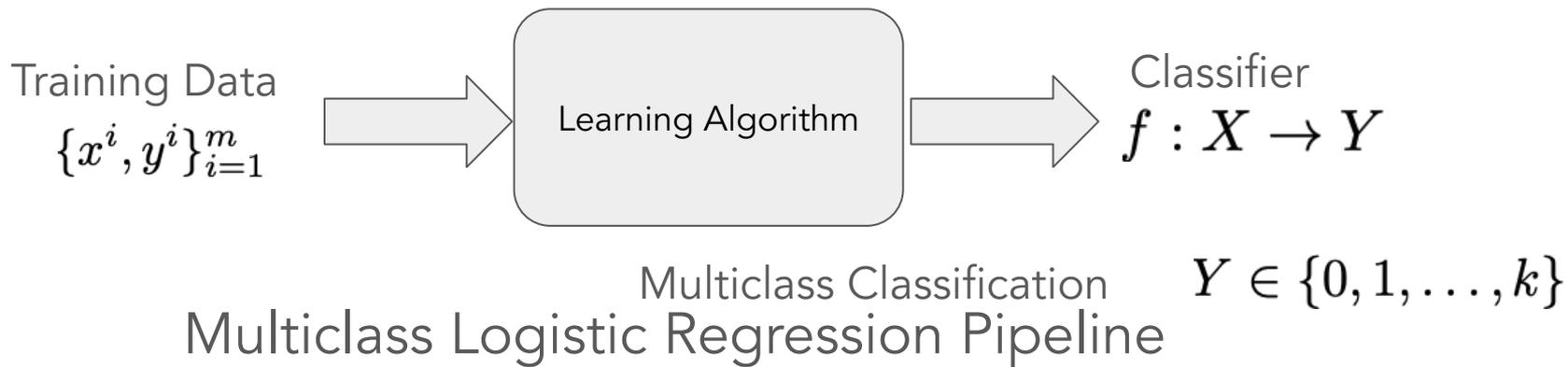Multiclass Classification $\quad Y \in \{0, 1, \ldots, k\}$
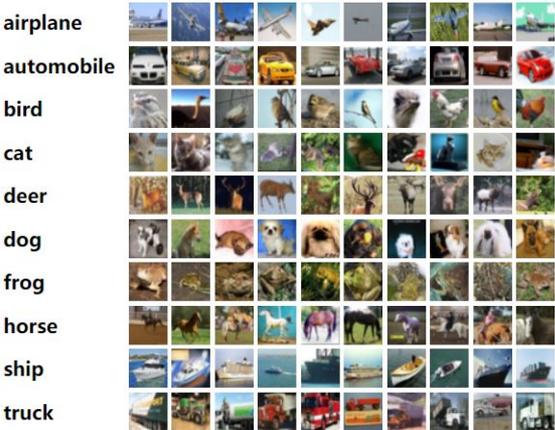
## Multiclass Logistic Regression Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

# Probabilistic Model in Binary Classification:
## Bernoulli Likelihood

$$\begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{if } y = 0 \end{cases} \qquad p \in [0, 1]$$
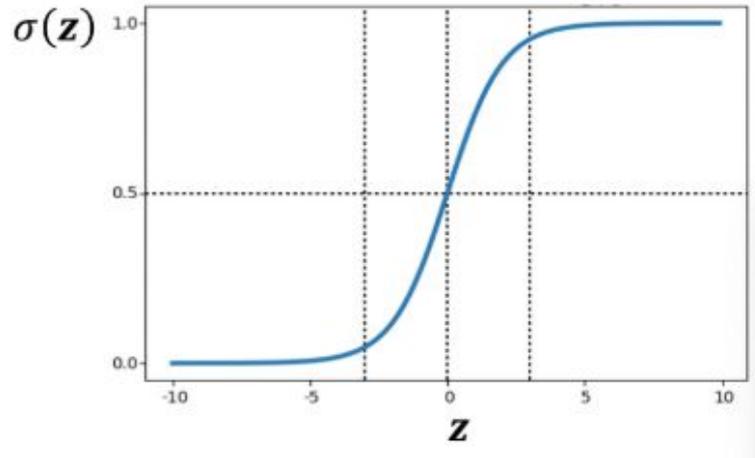
$$p(y) = p^y (1 - p)^{(1-y)}$$

# Probabilistic Model in Binary Classification:
## Bernoulli Likelihood

$$p(y) = p^y (1-p)^{(1-y)}$$

$$p(y|x; \theta) = p(y = 1|\theta^\top x)^y \{1 - p(y = 1|\theta^\top x)\}^{(1-y)}$$

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



$$p(y = 1|\theta^\top x) = \sigma(\theta^\top x) \in [0, 1]$$

# Probabilistic Model in Multiclass Classification:
## Categorical Likelihood

$$p(y = i) = p_i, \quad \sum_{i=1}^{k} p_i = 1, \quad p_i \geq 0$$

$$p(y) = \prod_{i=1}^{k} p_i^{y_i}$$

$$p = (p_1, p_2, \ldots, p_k)$$

$$y = (y_1, y_2, \ldots, y_k), \quad y_i \in 0, 1, \quad \sum_{i=1}^{k} y_i = 1$$

1-of-k code

# Probabilistic Model in Multiclass Classification:
## Categorical Likelihood

$$p(y) = \prod_{i=1}^{k} p_i^{y_i}$$

$$p = (p_1, p_2, \ldots, p_k)$$

$$p(y | \{\theta_i^\top x\}_{i=1}^{k}) = \prod_{i=1}^{k} p(y_i = 1 | \theta_i^\top x)^{y_i}$$
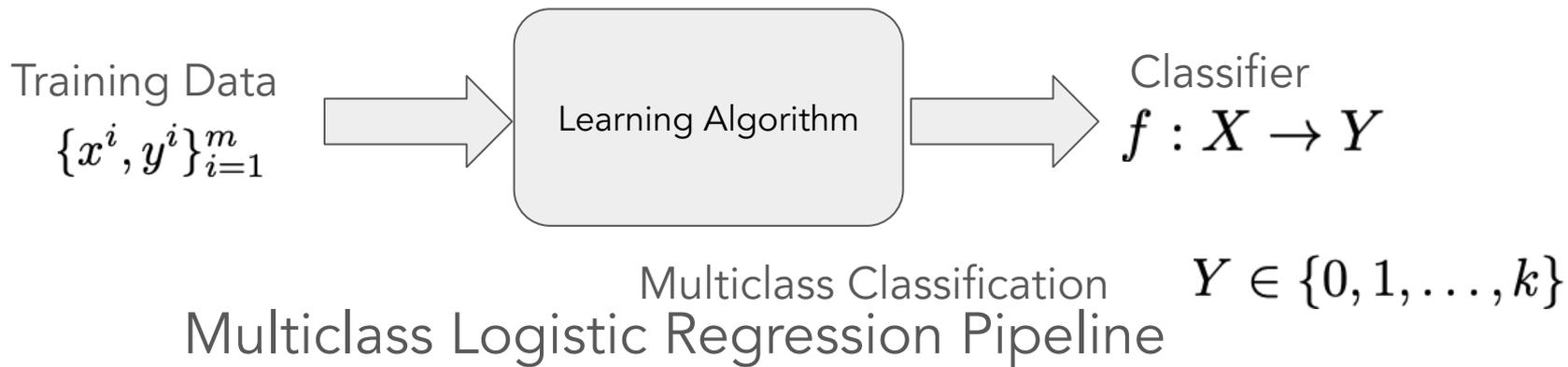
# Softmax Parametrization

$$p(y_i = 1 | \theta_i^\top x) \in (0, 1), \quad \sum_{i=1}^{k} p(y_i = 1 | \theta_i^\top x) = 1$$

Positivity

$$p(y_i = 1 | \theta_i^\top x) \propto \exp(\theta_i^\top x)$$

Normalization

$$p(y_i = 1 | \theta_i^\top x) = \frac{\exp(\theta_i^\top x)}{\sum_{i=1}^{k} \exp(\theta_i^\top x)}$$

# Multiclass Logistic Regression Algorithms

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Classifier
$$f : X \rightarrow Y$$

Multiclass Classification    $Y \in \{0, 1, \ldots, k\}$

## Multiclass Logistic Regression Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

# MLE

- Given all input data $\{x^i, y^i\}_{i=1}^m$

$$p(y^i|\theta^\top x^i) = \prod_{j=1}^k p(y_j^i = 1|\theta^\top x^i)^{y_j^i}$$

- Log-likelihood

$$\ell(\theta) = \sum_{i=1}^m \sum_{j=1}^k y_j^i \log p(y_j^i = 1|\theta^\top x^i)$$

# MLE

- Given all input data $\{x^i, y^i\}_{i=1}^m$

$$p(y^i|\theta^\top x^i) = \prod_{j=1}^k p(y_j^i = 1|\theta^\top x^i)^{y_j^i}$$

- Log-likelihood

$$\ell(\theta) = \sum_{i=1}^m \sum_{j=1}^k y_j^i \log p(y_j^i = 1|\theta^\top x^i)$$

$$= \sum_{i=1}^m \sum_{j=1}^k y_j^i \log \frac{\exp(\theta_j^\top x^i)}{\sum_{c=1}^k \exp(\theta_c^\top x^i)}$$

$$= \sum_{i=1}^m \sum_{j=1}^k y_j^i (\theta_j^\top x^i) - \sum_{i=1}^m \log \sum_{c=1}^k \exp(\theta_c^\top x^i)$$

# MLE

- Given all input data $\{x^i, y^i\}_{i=1}^m$

$$p(y^i|\theta^\top x^i) = \prod_{j=1}^k p(y^i_j = 1|\theta^\top x^i)^{y^i_j}$$

- Log-likelihood

$$\ell(\theta) = \sum_{i=1}^m \sum_{j=1}^k y^i_j \log p(y^i_j = 1|\theta^\top x^i) \quad \text{cross-entropy}$$

$$= \sum_{i=1}^m \sum_{j=1}^k y^i_j \log \frac{\exp(\theta_j^\top x^i)}{\sum_{c=1}^k \exp(\theta_c^\top x^i)}$$

$$= \sum_{i=1}^m \sum_{j=1}^k y^i_j (\theta_j^\top x^i) - \sum_{i=1}^m \log \sum_{c=1}^k \exp(\theta_c^\top x^i)$$

# MAP

- Likelihood

$$p(y = j | x, \theta) = \frac{\exp(\theta_j^\top x)}{\sum_{c=1}^{k} \exp(\theta_c^\top x)}$$

- Prior

$$p(\theta) \propto \exp(-\lambda \|\theta\|_2^2)$$

$$\max_{\theta} \log p(\theta | \{x^i, y^i\}_{i=1}^{m}) = \log L(\theta) + \log p(\theta)$$

$$= \sum_{i=1}^{m} \sum_{j=1}^{k} y_j^i \theta_j^\top x^i - \sum_{i=1}^{m} \log \sum_{c=1}^{k} \exp(\theta_c^\top x^i) - \lambda \|\theta\|_2^2$$

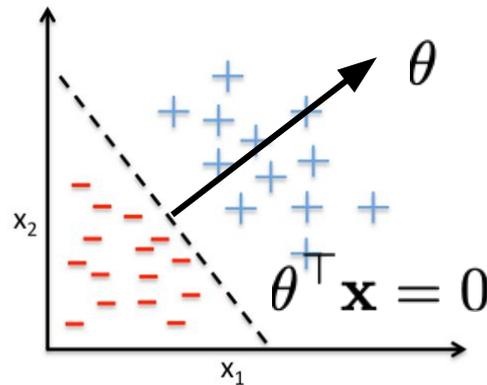# Logistic Regression is a Linear Classifier

- **Decision boundaries for Logistic Regression?**
  - At the decision boundary, label 1/0 are equiprobable.

$$P(y = 1 | \mathbf{x}, \theta) = \frac{1}{1 + e^{-\theta^\top \mathbf{x}}}, \qquad P(y = 0 | \mathbf{x}, \theta) = \frac{1}{1 + e^{\theta^\top \mathbf{x}}}$$

to be equal: $e^{-\theta^\top \mathbf{x}} = e^{\theta^\top \mathbf{x}}$, whose only solution is $\theta^\top \mathbf{x} = 0$.

✓ ⇒ Decision boundary is linear.

✓ ⇒ Logistic regression is a <u>probabilistic linear classifier</u>.

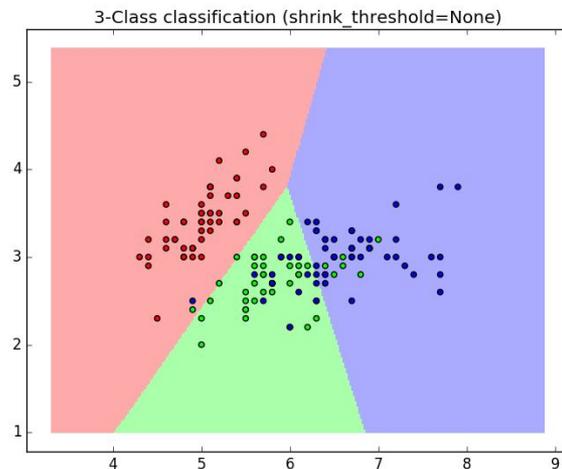# Multiclass Logistic Regression is a Linear Classifier
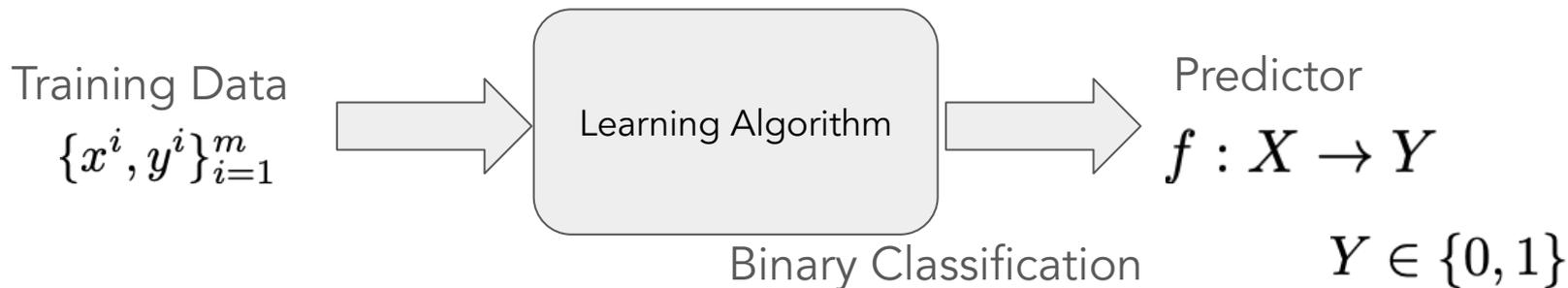
- Decision boundaries for Multiclass Logistic Regression?



3-Class classification (shrink_threshold=None)

✓ ⇒ Decision boundary is linear.

✓ ⇒ Multiclass Logistic regression is a probabilistic linear classifier.

# Multiclass Logistic Regression Algorithms

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Binary Classification

Predictor
$$f : X \to Y$$

$$Y \in \{0, 1\}$$

## Multiclass Logistic Regression Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

# Select Optimizer

$$\ell(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{k} y_j^i (\theta_j^\top x^i) - \sum_{i=1}^{m} \log \sum_{c=1}^{k} \exp(\theta_c^\top x^i)$$

- Necessary Condition

- (Stochastic) Gradient Descent

# Necessary Condition?

$$p(y = j | x, \theta) = \frac{\exp(\theta_j^\top x)}{\sum_{c=1}^{k} \exp(\theta_c^\top x)}$$

$$\frac{\partial \log L(\theta)}{\partial \theta} = \sum_{i=1}^{m} \sum_{j=1}^{k} (y_j^i - p(y_j^i = 1 | x^i, \theta)) x^i = 0$$

Nonlinear Equation!
Does NOT admit a closed-form solution

# Gradient Calculation of MLE

$$\max_{\theta} \log L(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{k} y_j^i \theta_j^\top x^i - \sum_{i=1}^{m} \log \sum_{c=1}^{k} \exp(\theta_c^\top x^i)$$

$$\frac{\partial \log L(\theta)}{\partial \theta} = \sum_{i=1}^{m} \sum_{j=1}^{k} (y_j^i - p(y_j^i = 1 | x^i, \theta)) x^i$$

$$p(y_j^i = 1 | x^i, \theta) = \frac{\exp(\theta_j^\top x^i)}{\sum_{c=1}^{k} \exp(\theta_c^\top x^i)}$$

# Gradient Calculation of MAP

$$\max_{\theta} \log L(\theta) = \sum_{i=1}^{m} \sum_{j=1}^{k} y_j^i \theta_j^\top x^i - \sum_{i=1}^{m} \log \sum_{c=1}^{k} \exp(\theta_c^\top x^i) - \lambda ||\theta||_2^2$$
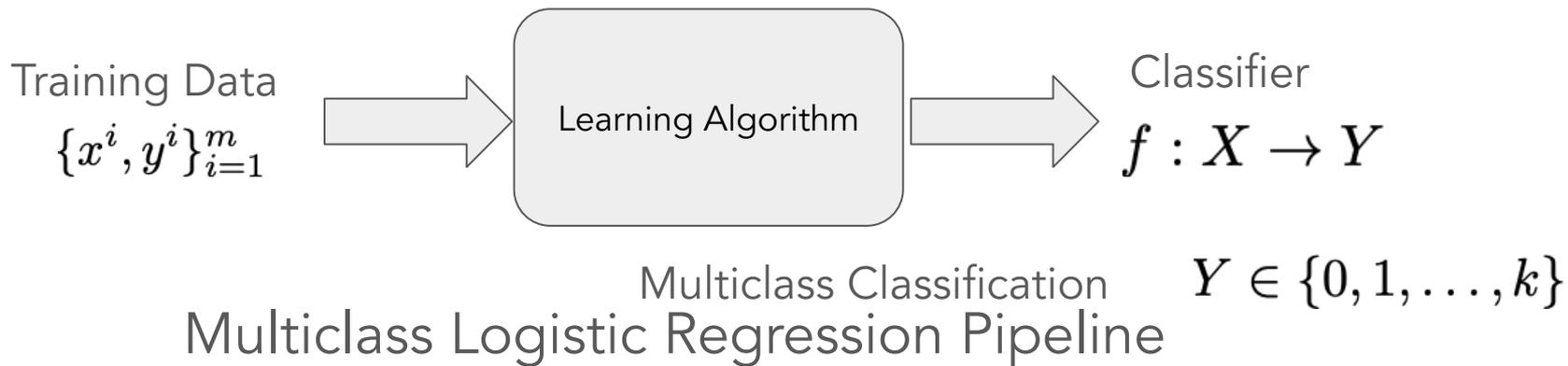
$$\frac{\partial \log L(\theta)}{\partial \theta} = \sum_{i=1}^{m} \sum_{j=1}^{k} (y_j^i - p(y_j^i = 1 | x^i, \theta)) x^i - 2\lambda\theta$$

# (Stochastic) Gradient Descent

- Initialize parameter $\theta^0$

- Do

$$\theta^{t+1} \leftarrow \theta^t + \eta \sum_{i=1}^{m} \sum_{j=1}^{k} (y_j^i - p(y_j^i = 1 | x^i, \theta)) x^i \left[ -2\lambda\theta \right]$$

# Multiclass Logistic Regression Algorithms

Training Data

$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Classifier

$f : X \rightarrow Y$

Multiclass Classification     $Y \in \{0, 1, \ldots, k\}$

## Multiclass Logistic Regression Pipeline

1. Build probabilistic models:
   Categorical Distribution + Linear Model
2. Derive loss function: MLE and MAP
3. Select optimizer: (Stochastic) Gradient Descent

# Naive Bayes Classification

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Classifier
$$f : X \to Y$$

Multiclass Classification

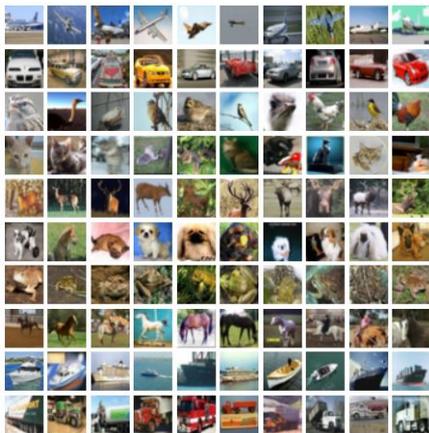$$Y \in \{0, 1, \ldots, k\}$$

airplane
automobile
bird
cat
deer
dog
frog
horse
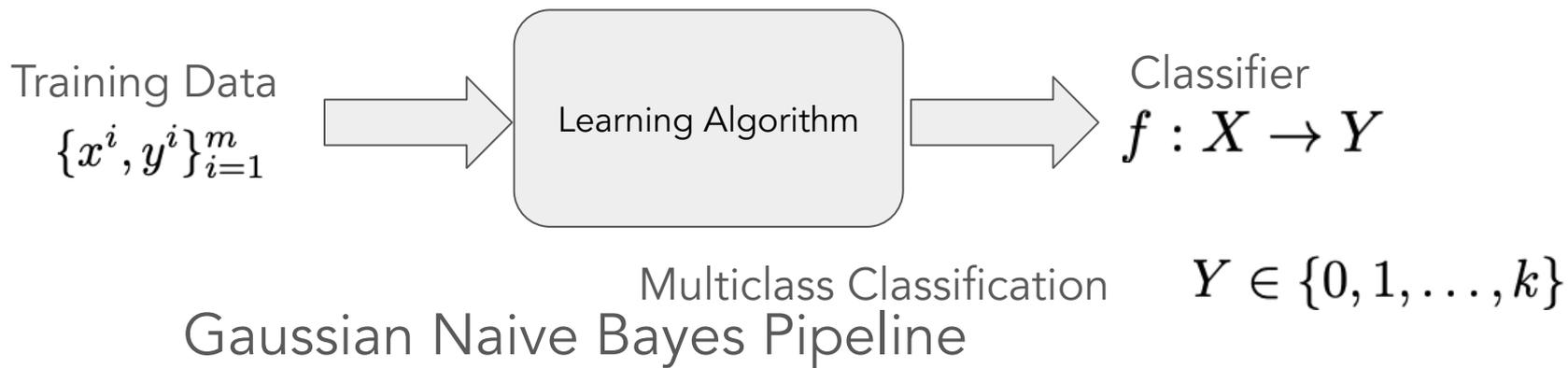ship
truck

# Naive Bayes Classification

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Classifier
$$f : X \to Y$$

Multiclass Classification

$$Y \in \{0, 1, \ldots, k\}$$

Gaussian Naive Bayes Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

# Bayes' Rule

Softmax in Multiclass
Classification

$$p(y_i = 1 | \theta_i^\top x) = \frac{\exp(\theta_i^\top x)}{\sum_{i=1}^{k} \exp(\theta_i^\top x)}$$

likelihood          Prior

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)} = \frac{P(x,y)}{\sum_z P(x,y)}$$

posterior          normalization constant

# Bayes' Rule

Prior

posterior

normalization constant

$$P(y|x) = \frac{P(x|y)P(y)}{P(x)} = \frac{P(x,y)}{\sum_z P(x,y)}$$

Prior: $P(y)$ $\quad \pi = (\pi_1, \pi_2, \ldots, \pi_k), \quad \sum_{i=1}^{k} \pi_i = 1, \pi_i \geq 0$

Likelihood (class conditional distribution : $p(x|y) = \mathcal{N}(x|\mu_y, \Sigma_y)$

Posterior: $P(y|x) = \dfrac{P(y)\mathcal{N}(x|\mu_y, \Sigma_y)}{\sum_y P(y)\mathcal{N}(x|\mu_y, \Sigma_y)}$

# Decision with Bayes' Rule

- The posterior probability of a test point

$$q_i(x) := P(y = i | x) = \frac{P(x|y)P(y)}{P(x)}$$

- Bayes decision rule:
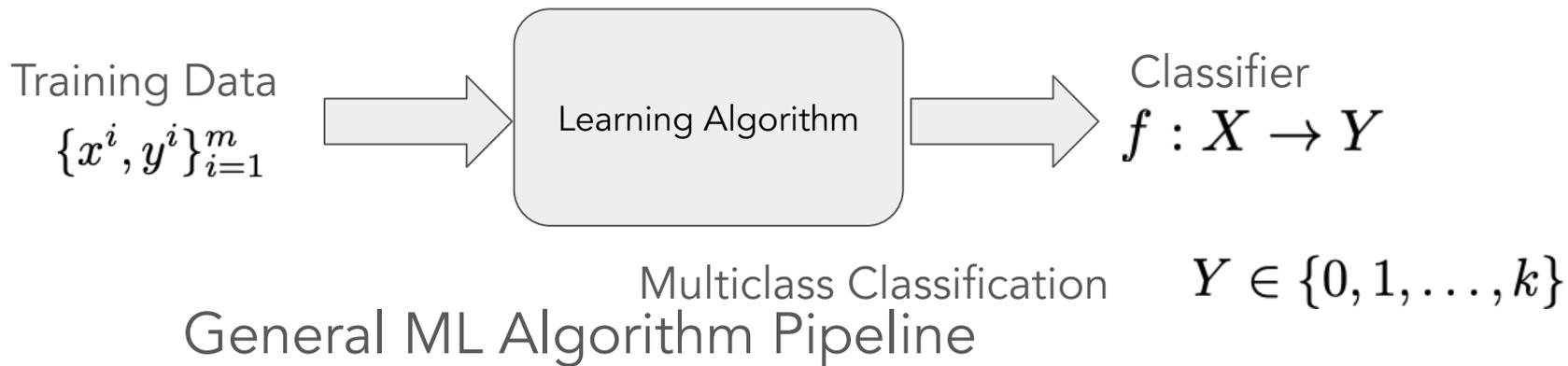  - If $q_i(x) > q_j(x)$, then $y = i$, otherwise $y = j$

- Alternatively:
  - If ratio $l(x) = \dfrac{P(x|y = i)}{P(x|y = j)} > \dfrac{P(y = j)}{P(y = i)}$ , then $y = i$, otherwise $y = j$

  - Or look at the log-likelihood ratio $h(x) = \ln \dfrac{q_i(x)}{q_j(x)}$

# Naive Bayes

Training Data
$\{x^i, y^i\}_{i=1}^m$

Learning Algorithm

Classifier
$f : X \to Y$

Multiclass Classification $Y \in \{0, 1, \ldots, k\}$

## General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

# MLE of Naive Bayes

$$\theta = [\mu, \Sigma, \pi], \quad Z = \sqrt{(2\pi)^D \det(\Sigma)}$$

$$p(x^i | y^i_j = 1, \theta) = \frac{1}{Z} \exp\left(-\frac{1}{2}(x^i - \mu_j)^\top \Sigma_j^{-1}(x^i - \mu_j)\right)$$

$$P(y) \quad \pi = (\pi_1, \pi_2, \ldots, \pi_k), \quad \sum_{i=1}^{k} \pi_i = 1, \pi_i \geq 0$$

# MLE of Naive Bayes

$$\theta = [\mu, \Sigma, \pi], \quad Z = \sqrt{(2\pi)^D \det(\Sigma)}$$

$$p(x^i|y_j^i = 1, \theta) = \frac{1}{Z} \exp\left(-\frac{1}{2}(x^i - \mu_j)^\top \Sigma_j^{-1}(x^i - \mu_j)\right)$$

$$P(y) \quad \pi = (\pi_1, \pi_2, \ldots, \pi_k), \quad \sum_{i=1}^{k} \pi_i = 1, \pi_i \geq 0$$

$$\log L(\theta) = \log p(x, y|\theta)$$

# MLE of Naive Bayes

$$\theta = [\mu, \Sigma, \pi], \quad Z = \sqrt{(2\pi)^D \det(\Sigma)}$$

$$p(x^i | y_j^i = 1, \theta) = \frac{1}{Z} \exp\left(-\frac{1}{2}(x^i - \mu_j)^\top \Sigma_j^{-1}(x^i - \mu_j)\right)$$

$$\log L(\theta) = \log p(x, y | \theta) = \log p(y | \theta) + \log p(x | y, \theta)$$

$$\log L(\theta) = \sum_{i=1}^{N} \sum_{j=1}^{k} y_j^i \log \pi_j - \sum_{i=1}^{N} \log Z - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{k} y_j^i (x^i - \mu_j)^\top \Sigma_j^{-1}(x^i - \mu_j)$$

Want $\arg\max_{\theta} \log L(\theta)$ subject to $\sum_{j=1}^{k} \pi_j = 1$
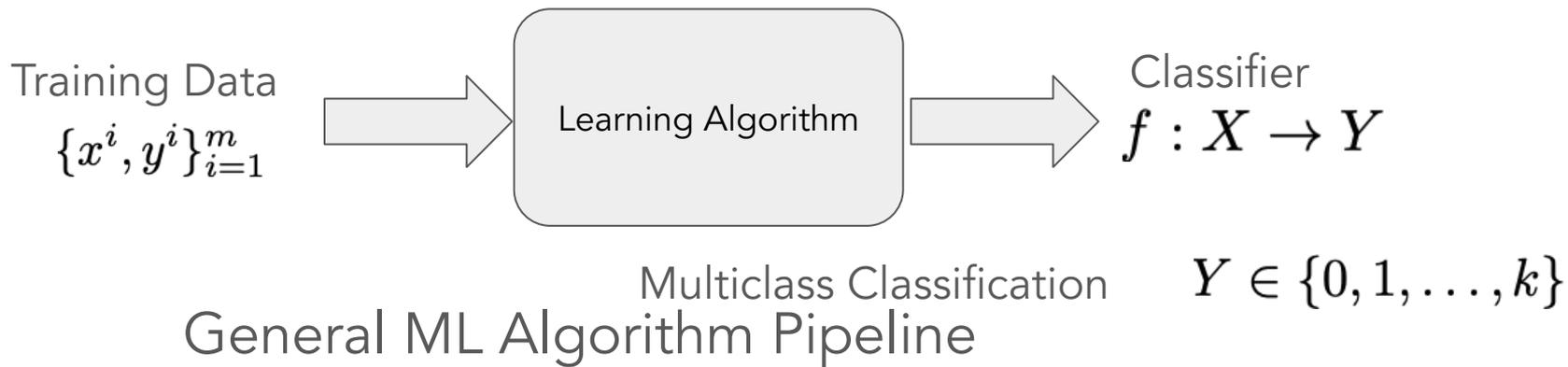
# MAP of Naive Bayes

$$p(\theta) \propto \exp(-\lambda \|\theta\|_2^2)$$

$$\log L(\theta) = \log p(x, y|\theta) = \log p(y|\theta) + \log p(x|y, \theta)$$

$$\max_{\theta} \log p(\theta|\{x^i, y^i\}_{i=1}^m) = \log L(\theta) + \log p(\theta)$$

# Naive Bayes

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Classifier
$$f : X \to Y$$

Multiclass Classification

$$Y \in \{0, 1, \ldots, k\}$$

## General ML Algorithm Pipeline

1. Build probabilistic models
2. Derive loss function (by MLE or MAP)
3. Select optimizer

# Select Optimizer

$$\sum_{i=1}^{N} \sum_{j=1}^{k} y_j^i \log \pi_j - \log Z - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{k} y_j^i (x^i - \mu_j)^\top \Sigma_j^{-1} (x^i - \mu_j)$$

- Necessary Condition

- (Stochastic) Gradient Descent

# Gradient Calculation of MLE

Take derivative w.r.t $\mu_k$

$$\frac{\partial \log L}{\partial \mu_k} = -\sum_{i=1}^{N} y_k^i \Sigma_k^{-1} (x^i - \mu_k) = 0$$

$$\mu_k = \frac{\sum_{i=1}^{N} y_k^i x^i}{\sum_{i=1}^{N} y_k^i}$$

# Necessary Condition for MLE

Take derivative w.r.t $\Sigma_k^{-1}$ (not $\Sigma_k$)

Note:

$$\frac{\partial \log L}{\partial \Sigma_k^{-1}} = -\sum_{i=1}^{N} y_k^i \left[ -\frac{\partial \log Z_k}{\partial \Sigma_k^{-1}} - \frac{1}{2}(x^i - \mu_k)(x^i - \mu_k)^\top \right] = 0$$

$$\Sigma_k = \frac{\sum_{i=1}^{N} y_k^i (x^i - \mu_k)(x^i - \mu_k)^\top}{\sum_{i=1}^{N} y_k^i}$$

# Necessary Condition for MLE
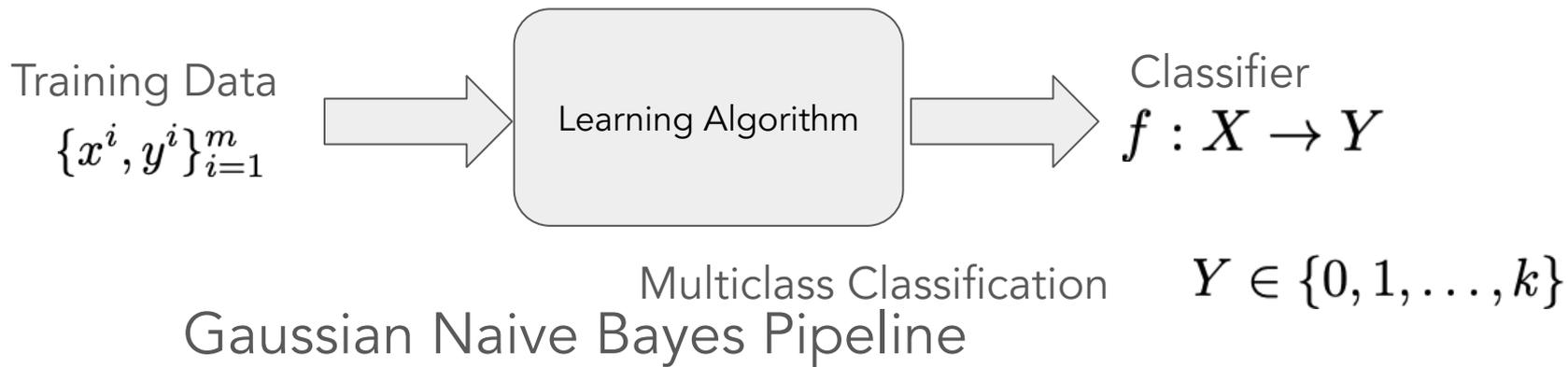
Use Lagrange multiplier to derive $\pi_k$

$$\frac{\partial L(\theta)}{\partial \pi_k} + \lambda \frac{\partial \sum_k \pi_k}{\partial \pi_k} = 0 \Rightarrow \lambda = -\sum_{i=1}^{N} y_k^i \frac{1}{\pi_k}$$

$$\pi_k = -\frac{\sum_{i=1}^{N} y_k^i}{\lambda}$$

Apply constraint: $\sum_k \pi_k = 1 \Rightarrow \lambda = -N$

$$\boxed{\pi_k = \frac{\sum_{i=1}^{N} y_k^i}{N}}$$

# Naive Bayes

Training Data
$$\{x^i, y^i\}_{i=1}^m$$

Learning Algorithm

Classifier
$$f : X \to Y$$

Multiclass Classification     $$Y \in \{0, 1, \ldots, k\}$$

## Gaussian Naive Bayes Pipeline

1. Build probabilistic models: Gaussian Likelihood
2. Derive loss function:  MLE or MAP
3. Select optimizer: Necessary Condition

Q&A

# HW 2 is out
# Team Formation Due: Feb 16th