DS 4400

Machine Learning and Data Mining I
Spring 2021

Alina Oprea
Associate Professor
Khoury College of Computer Science
Northeastern University

February 16 2021

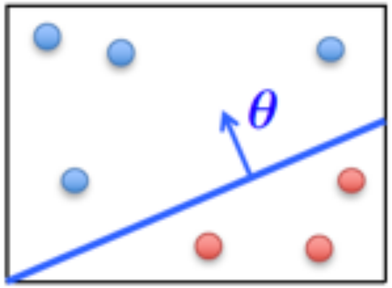
Outline

- Logistic regression
  - Classification based on probability
- Maximum Likelihood Estimation
  - Application to logistic regression
  - Cross-entropy objective
- Gradient descent for logistic regression
- Logistic regression lab
- Evaluation metrics for classifiers

Linear Classifiers

- **Linear classifiers**: represent decision boundary by hyperplane

$$\boldsymbol{\theta} = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \vdots \\ \theta_d \end{bmatrix} \quad \boldsymbol{x}^\mathsf{T} = \begin{bmatrix} 1 & x_1 & \dots & x_d \end{bmatrix}$$



$h_\theta(x) = f(\theta^T x)$ linear function
- If $\theta^T x > 0$ classify "Class 1"
- If $\theta^T x < 0$ classify "Class 0"

All the points x on the hyperplane satisfy: $\theta^T x = 0$

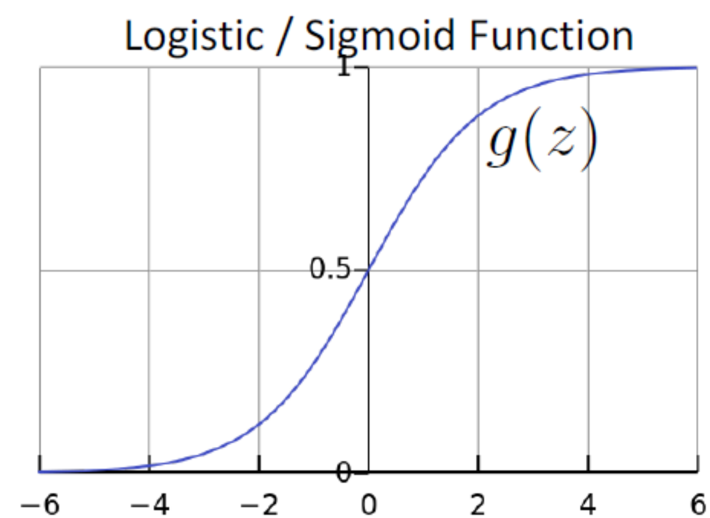
Logistic Regression

- Takes a probabilistic approach to learning discriminative functions (i.e., a classifier)
- $h_{\boldsymbol{\theta}}(\boldsymbol{x})$ should give $P(Y = 1|X; \theta)$
  - Want $0 \leq h_{\boldsymbol{\theta}}(\boldsymbol{x}) \leq 1$
- Logistic regression model:

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}\right)$$

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

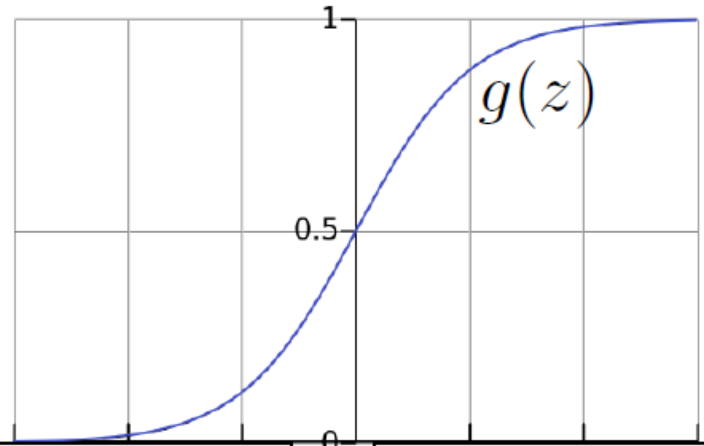
$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{\theta}^{\mathsf{T}}\boldsymbol{x}}}$$



Logistic Regression

$$h_{\boldsymbol{\theta}}(\boldsymbol{x}) = g\left(\boldsymbol{\theta}^\intercal \boldsymbol{x}\right)$$

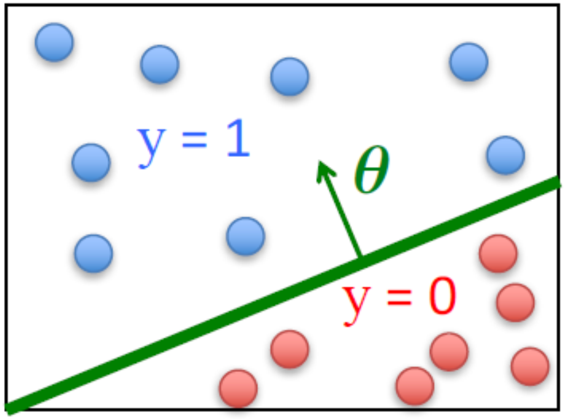
$$g(z) = \frac{1}{1+e^{-z}}$$



$\boldsymbol{\theta}^\intercal \boldsymbol{x}$ should be large negative values for negative instances

$\boldsymbol{\theta}^\intercal \boldsymbol{x}$ should be large positive values for positive instances

- Assume a threshold and...
  - Predict $Y = 1$ if $h_{\boldsymbol{\theta}}(\boldsymbol{x}) \geq 0.5$
  - Predict $Y = 0$ if $h_{\boldsymbol{\theta}}(\boldsymbol{x}) < 0.5$



Logistic Regression is a linear classifier!

Maximum Likelihood Estimation (MLE)

Given training data $X = \{x_1, \dots, x_N\}$ with labels $\mathrm{Y} = \{y_1, \dots, y_N\}$

What is the likelihood of training data for parameter θ?

Define likelihood function

$$Max_\theta \, L(\theta) = P[Y|X;\theta]$$

Assumption: training labels are conditionally independent

$$L(\theta) = \prod_{i=1}^{N} P[Y = y_i | X = x_i; \theta]$$

General probabilistic method for classifier training

Log Likelihood

- Max likelihood is equivalent to maximizing log of likelihood

$$L(\theta) = \prod_{i=1}^{N} P[Y = y_i | X = x_i; \theta]$$

$$\log L(\theta) = \sum_{i=1}^{N} \log P[Y = y_i | X = x_i; \theta]$$

- They both have the same maximum θ_{MLE}

MLE for Logistic Regression

$$\mathrm{P}(Y = y_i | X = x_i; \theta) = h_\theta(x_i)^{y_i}\left(1 - h_\theta(x_i)\right)^{1-y_i}$$

$$\theta_{MLE} = \mathrm{argmax}_\theta \sum_{i=1}^{N} \log P[Y = y_i | X = x_i; \theta]$$
$$= \mathrm{argmax}_\theta \sum_{i=1}^{N} y_i \log h_\theta(x_i) + (1 - y_i) \log\left(1 - h_\theta(x_i)\right)$$

Logistic regression objective

$$\min_\theta J(\theta)$$

$$J(\theta) = -\sum_{i=1}^{N} [y_i \log h_\theta(x_i) + (1 - y_i) \log\left(1 - h_\theta(x_i)\right)]$$

Cross-Entropy Objective

$$J(\theta) = -\sum_{i=1}^{N}[y_{\mathrm{i}}\log h_\theta(x_i) + (1-y_i)\log\left(1-h_\theta(x_i)\right)]$$

- Cost of a single instance:

$$\text{cost}\,(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1-h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$

- Can re-write objective function as

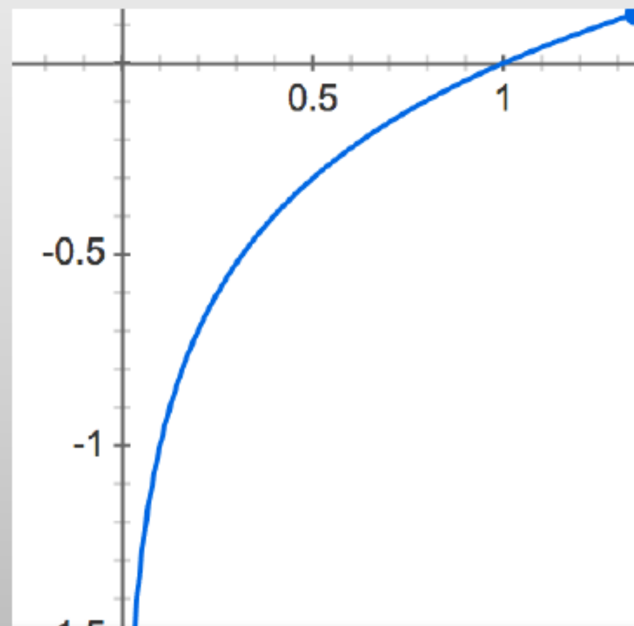
$$J(\boldsymbol{\theta}) = \sum_{i=1}^{n} \underbrace{\text{cost}\left(h_\theta(x_i), y_i \right)}_{\text{Cross-entropy loss}}$$

Cross-entropy loss

Intuition

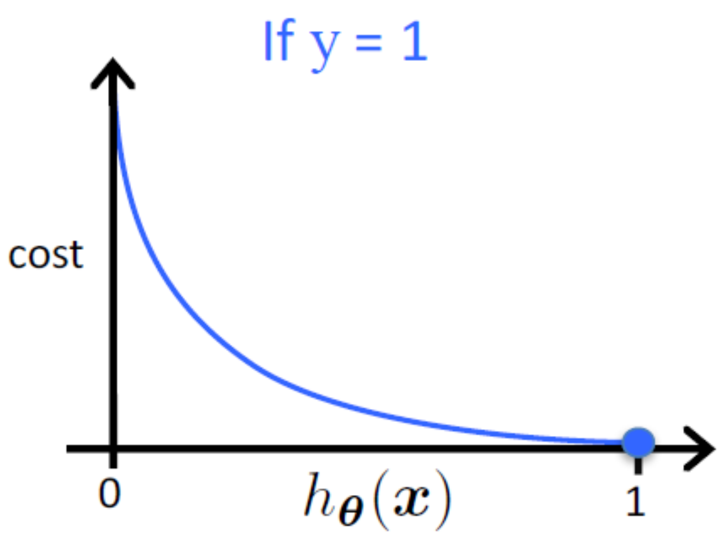
$$\text{cost}\left(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y\right) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$

Aside: Recall the plot of log(z)



Intuition

$$\text{cost}\,(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$

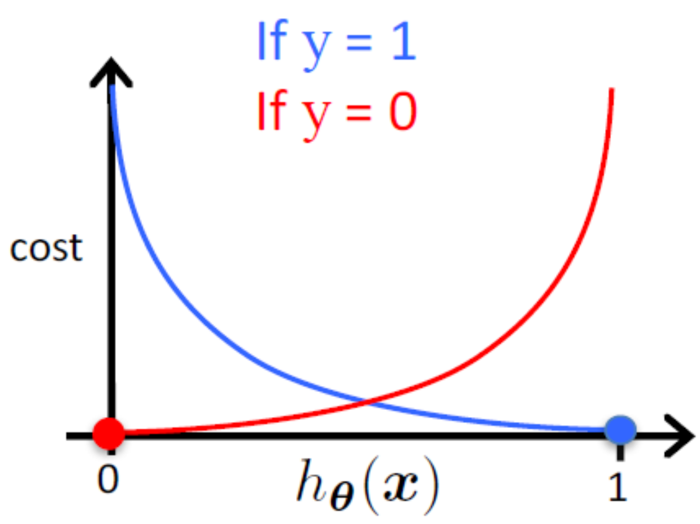


If y = 1

- Cost = 0 if prediction is correct
- As $h_{\boldsymbol{\theta}}(\boldsymbol{x}) \to 0, \text{cost} \to \infty$
- Captures intuition that larger mistakes should get larger penalties
  - e.g., predict $h_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0$, but y = 1

Intuition

$$\text{cost}\,(h_{\boldsymbol{\theta}}(\boldsymbol{x}), y) = \begin{cases} -\log(h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 1 \\ -\log(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) & \text{if } y = 0 \end{cases}$$



If y = 0

- Cost = 0 if prediction is correct
- As $(1 - h_{\boldsymbol{\theta}}(\boldsymbol{x})) \rightarrow 0, \text{cost} \rightarrow \infty$
- Captures intuition that larger mistakes should get larger penalties

Gradient Descent for Logistic Regression

$$J(\theta) = -\sum_{i=1}^{N}[y_\mathrm{i}\log h_\theta(x_i) + (1 - y_i)\log\left(1 - h_\theta(x_i)\right)]$$

Want $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\boldsymbol{\theta})$$

simultaneous update for j = 0 ... d

Computing Gradients

- Derivative of sigmoid
  - $g(z) = \frac{1}{1+e^{-z}}$; $g'(z) = \frac{e^{-z}}{(1+e^{-z})^2} = g(z)(1 - g(z))$
- Derivative of hypothesis
  - $h_\theta(x_i) = g(\theta^T x_i) = g(\theta_j x_{ij} + \sum_{k \neq j} \theta_k x_{ik})$
  - $\frac{\partial h_\theta(x_i)}{\partial \theta_j} = \frac{\partial g(\theta^T x_i)}{\partial \theta_j} x_{ij} = g(\theta^T x_i)\left(1 - g(\theta^T x_i)\right) x_{ij}$
- Derivation of C_i
  - $\frac{\partial C_i}{\partial \theta_j} = y_i \frac{1}{h_\theta(x_i)} \mathrm{g}(\theta^T x_i)\left(1 - \mathrm{g}(\theta^T x_i)\right) x_{ij}$ -

    $(1 - y_i) \frac{1}{1 - h_\theta(x_i)} \mathrm{g}(\theta^T x_i)\left(1 - \mathrm{g}(\theta^T x_i)\right) x_{ij}$

    $= \left(y_i - h_\theta(x_i)\right) x_{ij}$

Gradient Descent for Logistic Regression

$$J(\theta) = -\sum_{i=1}^{N}[y_\mathrm{i}\log h_\theta(x_i) + (1 - y_i)\log\left(1 - h_\theta(x_i)\right)]$$

Want $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence (simultaneous update for j = 0 ... d)

$$\theta_0 \leftarrow \theta_0 - \alpha\sum_{i=1}^{N}(h_\theta(x_i) - y_i)$$

$$\theta_j \leftarrow \theta_j - \alpha\sum_{i=1}^{N}(h_\theta(x_i) - y_i)x_{ij}$$

Gradient Descent for Logistic Regression

Want $\min_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$

- Initialize $\boldsymbol{\theta}$
- Repeat until convergence (simultaneous update for j = 0 ... d)

$$\theta_0 \leftarrow \theta_0 - \alpha \sum_{i=1}^{N} (h_\theta(x_i) - y_i)$$

$$\theta_j \leftarrow \theta_j - \alpha \sum_{i=1}^{N} (h_\theta(x_i) - y_i) x_{ij}$$

This looks IDENTICAL to Linear Regression!

- However, the form of the model is very different:

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

Regularized Logistic Regression

$$J(\theta) = -\sum_{i=1}^{N}[y_{\mathrm{i}} \log h_\theta(x_i) + (1-y_i)\log\left(1-h_\theta(x_i)\right)]$$

- We can regularize logistic regression exactly as before:

$$J_{\text{regularized}}(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) + \lambda \sum_{j=1}^{d} \theta_j^2$$
$$= J(\boldsymbol{\theta}) + \lambda \|\boldsymbol{\theta}_{[1:d]}\|_2^2$$

L2 regularization

Logistic Regression
Lab Example

Classifier Evaluation

- Classification is a supervised learning problem
  - Prediction is binary or multi-class
- Classification techniques
  - Linear classifiers
    - Perceptron (online or batch mode)
    - Logistic regression (probabilistic interpretation)
  - Instance learners
    - kNN: need to store entire training data
- Cross-validation should be used for parameter selection and estimation of model error

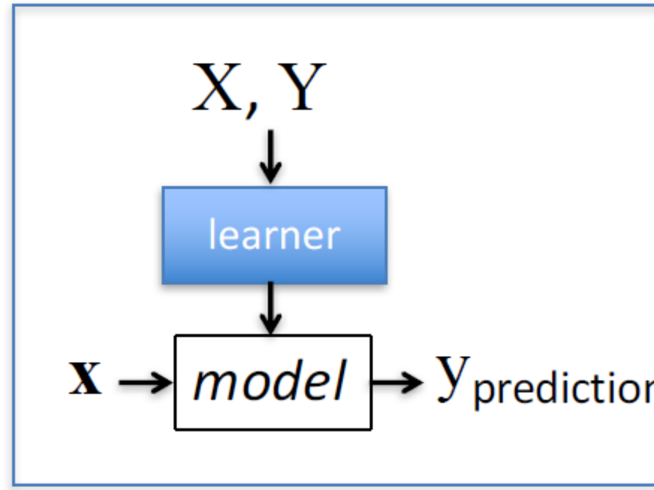
Evaluation of classifiers

Given: labeled training data $X, Y = \{\langle \boldsymbol{x}_i, y_i \rangle\}_{i=1}^{n}$

- Assumes each $\boldsymbol{x}_i \sim \mathcal{D}(\mathcal{X})$

Train the model:

model ← *classifier*.train(X, Y)



Apply the model to new data:

- Given: new unlabeled instance $\boldsymbol{x} \sim \mathcal{D}(\mathcal{X})$

$y_{prediction}$ ← *model*.predict($\mathbf{x}$)

Classification Metrics

$$\text{accuracy} = \frac{\#\ \text{correct predictions}}{\#\ \text{test instances}}$$

$$\text{error} = 1 - \text{accuracy} = \frac{\#\ \text{incorrect predictions}}{\#\ \text{test instances}}$$

- Training set accuracy and error
- Testing set accuracy and error

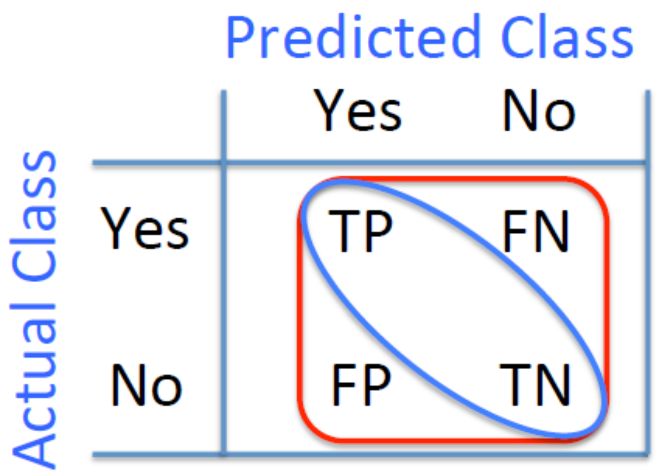
Confusion Matrix

Given a dataset of P positive instances and N negative instances:

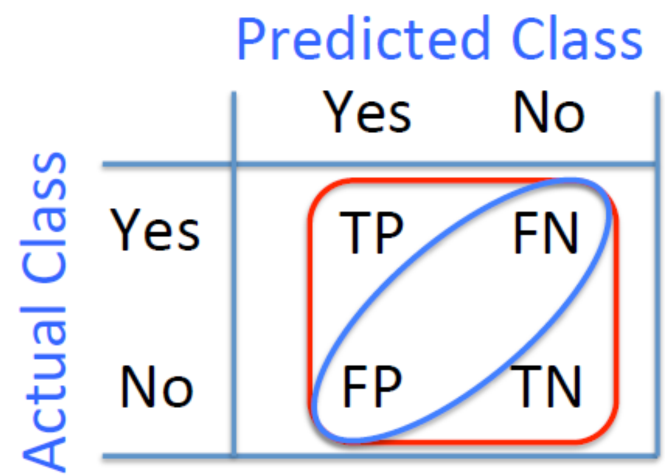
		Predicted Class	
		Yes	No
Actual Class	Yes	TP	FN
	No	FP	TN

Accuracy and Error

Given a dataset of P positive instances and N negative instances:



$$\text{accuracy} = \frac{TP + TN}{P + N}$$



$$\text{error} = 1 - \frac{TP + TN}{P + N} = \frac{FP + FN}{P + N}$$

Review

- Maximum Likelihood Estimation (MLE) is a general statistical method for parameter estimation
- Logistic regression is a linear classifier that predicts class probability
  - Cross-entropy objective derived with MLE
- Can be trained with Gradient Descent

Acknowledgements

- Slides made using resources from:
  - Andrew Ng
  - Eric Eaton
  - David Sontag
- Thanks!