CS 760: Machine Learning
Parametric modeling

Misha Khodak

University of Wisconsin-Madison

17 September 2025

Logistics

- **Announcements**:
  - HW 1 will be out right after class
    - available as hw1.zip on Canvas
    - due October 1st on Gradescope
  - Roadmap after today:
    - Regression & optimization (2 lectures)
    - Unsupervised learning (2 lectures)
    - Neural networks (5 lectures)
    - Midterm (in class)

Outline

- **Parametric modeling**
  - Comparison with non-parametric models
  - Generative vs. discriminative modeling
- **Logistic regression**
  - Maximum likelihood estimation
  - Setup
- **Naïve Bayes**
  - Bayesian modeling, MAP vs. MLE
  - Motivation / training / inference / smoothing
  - Examples (Bernoulli, Multiclass, Gaussian)

Outline

- **Parametric modeling**
  - Comparison with non-parametric models
  - Generative vs. discriminative modeling
- **Logistic regression**
  - Maximum likelihood estimation
  - Setup
- **Naïve Bayes**
  - Bayesian modeling, MAP vs. MLE
  - Motivation / training / inference / smoothing
  - Examples (Bernoulli, Multiclass, Gaussian)

Supervised Learning: Review

Problem setting

- Set of possible instances
- Unknown *target function*
- Set of *models* (a.k.a. *hypotheses*)

$$\mathcal{X}$$

$$f : \mathcal{X} \rightarrow \mathcal{Y}$$

$$\mathcal{H} = \{h | h : \mathcal{X} \rightarrow \mathcal{Y}\}$$

Get

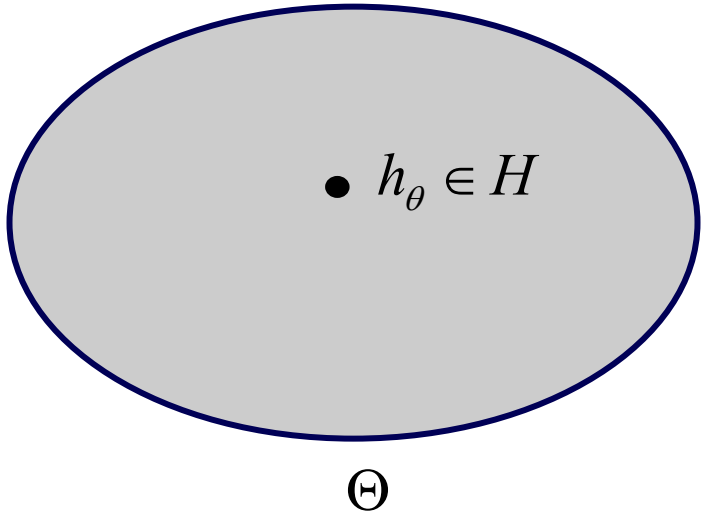
- Training set of instances for unknown target function *f*,

$$(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})$$

Goal: model *h* that best approximates *f*

Parametric Learning

- A way to categorize learning techniques
  - Parametric: hypotheses indexed by a **parameter**
  - Learning: find parameter yielding model that best approximates the target
  - **Ex**: linear models, neural networks



- Nonparametric methods:
  - Instance-based methods (KNN)
  - Decision trees

Discriminative Models

- **Idea**: hypothesis h directly predicts the label (given features)
  - y = h(x) or p(y|x) = h(x)

- Includes everything we've covered so far
  - k-NN
  - decision trees
  - linear models

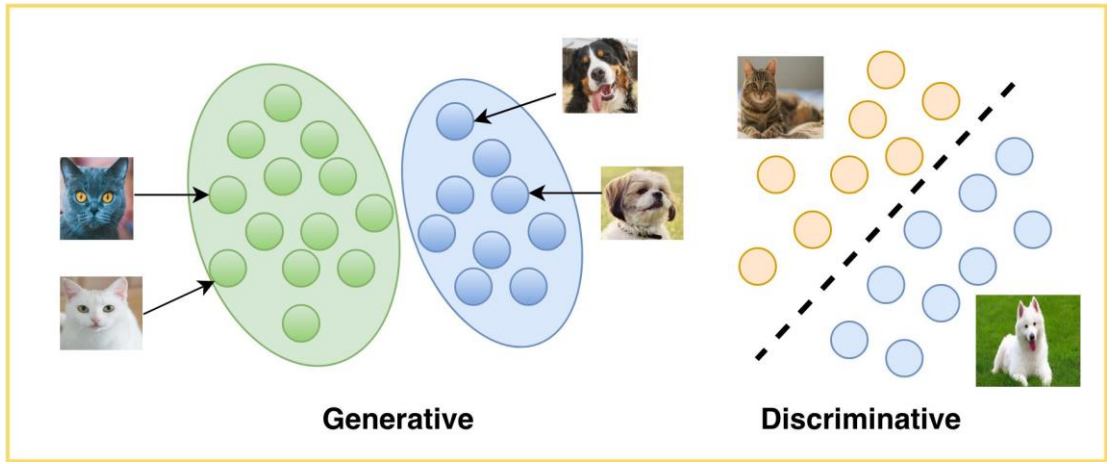
$$h_\theta(x) = \sum_{i=0}^{d} \theta_i x_i$$

Generative Models

- Hypothesis h specifies a **generative story** for how the data was created
  - h(x,y) = p(x,y) or h(x) = p(x) ⟵ **Note: supervised or unsupervised**
- Select a hypothesis via ML (or MAP)
  - Ex: roll a die. Weights for each side define data generation
  - Observe training data to learn hypothesis

Discriminative vs Generative

- Can define both for supervised/unsupervised learning
  - k-means (discriminative-like) vs mixture-of-Gaussians (generative)
- When should we use one over the other?
  - Not always obvious
  - Discriminative models:
    - Often easier to optimize
    - Targets exact performance measure
  - Generative models:
    - Handling missing data
    - Generation via sampling



LearnOpenCV

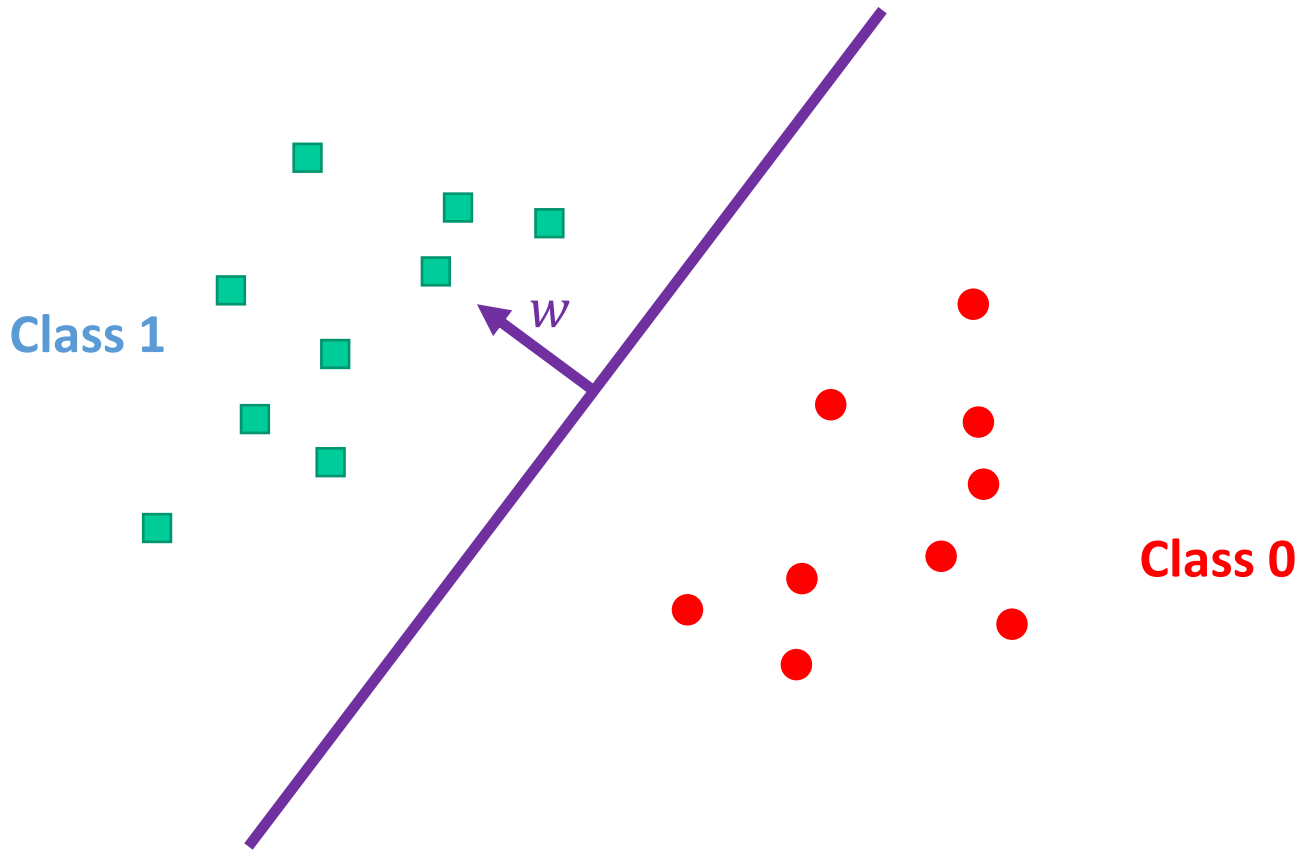
- Typical examples:
  - Discriminative: linear regression, logistic regression, SVM, many neural networks (not all!)
  - Generative: Naïve Bayes, Bayesian Networks, …

Outline

- **Parametric modeling**
  - Comparison with non-parametric models
  - Generative vs. discriminative modeling
- **Logistic regression**
  - Maximum likelihood estimation
  - Setup
- **Naïve Bayes**
  - Bayesian modeling, MAP vs. MLE
  - Motivation / training / inference / smoothing
  - Examples (Bernoulli, Multiclass, Gaussian)

Classification: Linear models

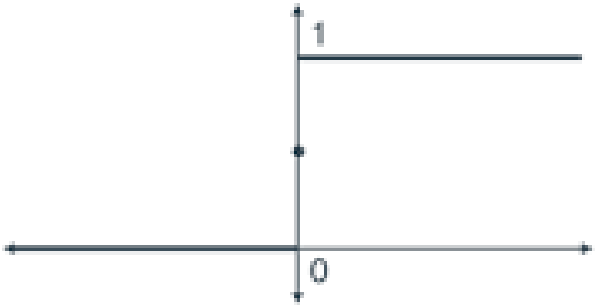
- How do we learn a linear separator between two classes?



Linear Classification: Attempt 1

- Hyperplane: solutions to $\theta^T x = c$
  - note: d-1 dimensional -- d (degree of freedom) – *1* (constraint)
- So… try to use such hyperplanes as separators?
  - Model: $f_\theta(x) = \theta^T x$
  - Predict: *y=1* if $\theta^T x > 0$, y=0 otherwise
  - i.e. $y = \text{step}(f_\theta(x))$
  - Training objective:

step function



difficult loss function to optimize!!

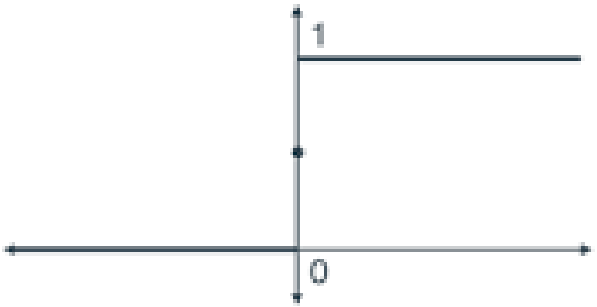
$$\ell(f_\theta) = \frac{1}{m} \sum_{i=1}^{m} \mathbb{1}\left[\text{step}(f_\theta(x^{(i)})) \neq y^{(i)}\right]$$

Linear Classification: Attempt 2

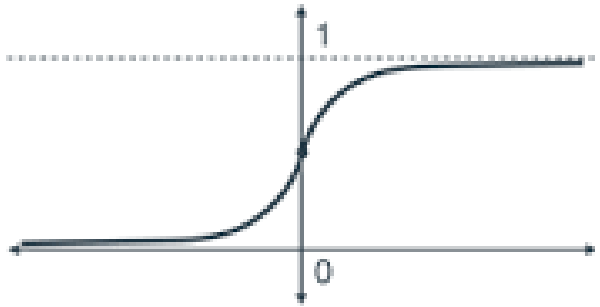
- Let us instead think probabilistically and learn $P_\theta(y|x)$ instead

- How?
  - Specify the conditional distribution $P_\theta(y|x)$
  - Use maximum likelihood estimation (MLE) to get a nicer loss
  - Run gradient descent (or related optimization algorithm)

step function



sigmoid function

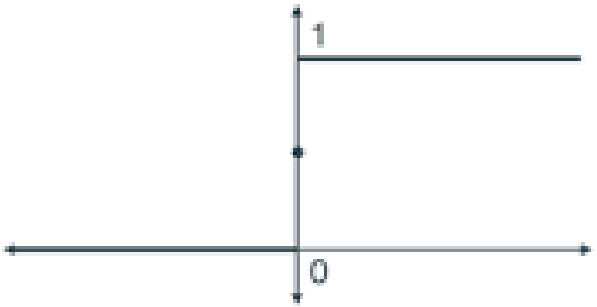


Linear Classification: Attempt 2

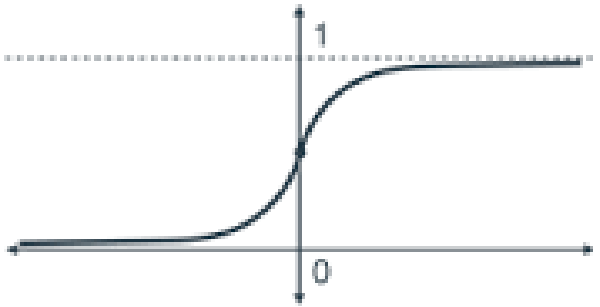
- Let us instead think probabilistically and learn $P_\theta(y|x)$ instead

- How?
  - Specify the conditional distribution $P_\theta(y|x)$
  - **Use maximum likelihood estimation (MLE) to get a nicer loss**
  - Run gradient descent (or related optimization algorithm)

step function



sigmoid function



Likelihood Function

- Captures the probability of seeing some data as a function of model parameters:

$$\mathcal{L}(\theta; X) = P_\theta(X)$$

  - If data is iid, we have $\mathcal{L}(\theta; X) = \prod_j p_\theta(x_j)$

- Often more convenient to work with the log likelihood
  - Both mathematically and for numerical stability
  - Log is a monotonic + strictly increasing function

Maximum Likelihood Estimation

- For some set of data, find the parameters that maximize the likelihood / log-likelihood

$$\hat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta; X)$$

- Example: suppose we have n samples from a Bernoulli distribution

$$P_\theta(X = x) = \begin{cases} \theta & x = 1 \\ 1 - \theta & x = 0 \end{cases}$$

then

$$\mathcal{L}(\theta; X) = \prod_{i=1}^{n} P(X = x_i) = \theta^k (1 - \theta)^{n-k}$$

k: #samples with x=1

Maximum Likelihood: Example

- Want to maximize likelihood w.r.t. Θ

$$\mathcal{L}(\theta; X) = \prod_{i=1}^{n} P(X = x_i) = \theta^k (1-\theta)^{n-k}$$

- Differentiate (use product rule) and set to 0. Get

$$\theta^{k-1}(1-\theta)^{n-k-1}(k - n\theta) = 0$$

- So: ML estimate is

$$\hat{\theta} = \frac{k}{n}$$

Practice: how about maximum log-likelihood?

$$\log \mathcal{L}(\theta; X) = k\log\theta + (n-k)\log(1-\theta)$$

gradient = 0 $\Rightarrow$ $\frac{k}{\theta} - \frac{n-k}{1-\theta} = 0$

$$\hat{\theta} = \frac{\mathrm{k}}{n}$$

ML: Conditional Likelihood

- Similar idea, but now using conditional probabilities:

$$\mathcal{L}(\theta; Y, X) = p_\theta(Y|X)$$

  - If data is iid, we have

$$\mathcal{L}(\theta; Y, X) = \prod_j p_\theta(y_j|x_j)$$

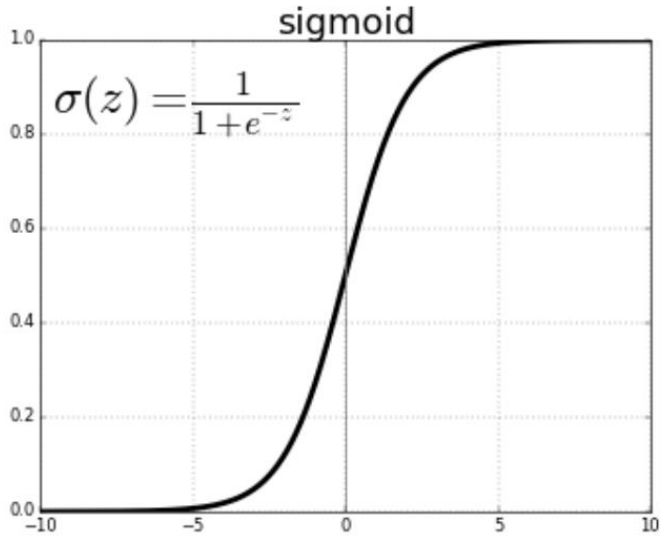
  - Now we can apply this to linear classification to get **logistic regression.**

Logistic Regression: Conditional Distribution

- Notation: $\sigma(z) = \frac{1}{1+\exp(-z)} = \frac{\exp(z)}{1+\exp(z)}$ $z \leftarrow \theta^T x$

sigmoid

"soft" version of step function



- **Conditional distribution model for logistic regression:**

$$P_\theta(y=1|x) = \sigma(\theta^T x) = \frac{1}{1+\exp(-\theta^T x)}$$

Logistic Regression: Loss

- Conditional MLE:

$$\log \text{ likelihood}(\theta|x^{(i)}, y^{(i)}) = \log P_\theta(y^{(i)}|x^{(i)})$$

- So

$$\min_\theta \ell(f_\theta) = \min_\theta -\frac{1}{n}\sum_{i=1}^{n} \log P_\theta(y^{(i)}|x^{(i)})$$

Or

$$\min_\theta -\frac{1}{n}\sum_{y^{(i)}=1} \log \sigma(\theta^T x^{(i)}) - \frac{1}{n}\sum_{y^{(i)}=0} \log(1-\sigma(\theta^T x^{(i)}))$$

Logistic regression: Summary

- **logistic regression = sigmoid conditional distribution + MLE**

- More precisely:
  - Give training data iid from some distribution D,
  - **Train**:

$$\min_{\theta} \ell(f_\theta) = \min_{\theta} -\frac{1}{n} \sum_{i=1}^{n} \log P_\theta(y^{(i)}|x^{(i)})$$

  - **Test**: output label probabilities

$$P_\theta(y = 1|x) = \sigma(\theta^T x) = \frac{1}{1 + \exp(-\theta^T x)}$$

Logistic Regression: Comparisons

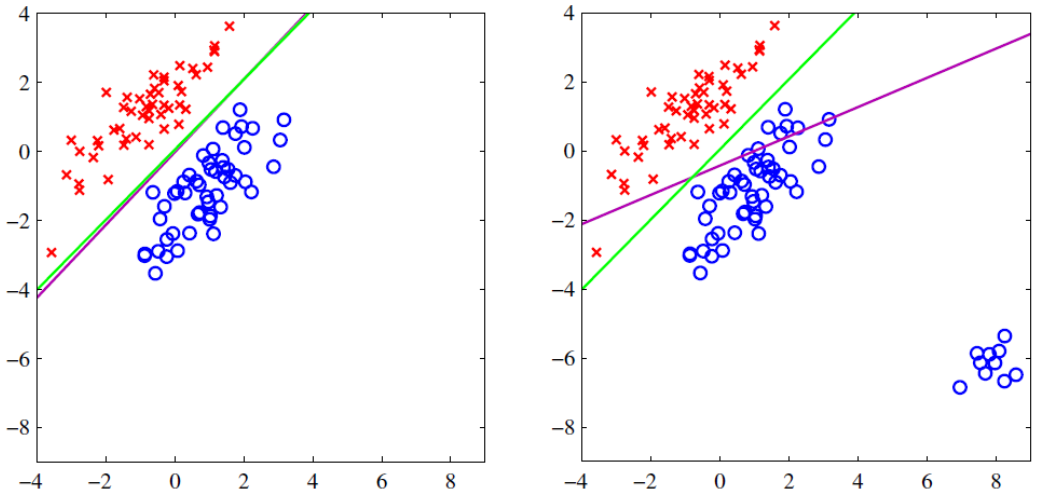
Recall the first attempt: $$\ell(f_\theta) = \frac{1}{m}\sum_{i=1}^{m} \mathbb{1}\left[\text{step}(f_\theta(x^{(i)})) \neq y^{(i)}\right]$$

difficult loss function to optimize!!

Another option: use the squared loss $\left(f_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$

$$\ell(f_\theta) = \frac{1}{n}\sum_{j=1}^{n}(f_\theta(x^{(j)}) - y^{(j)})^2$$

this is just regular linear regression, but it works poorly for classification ☹



Bishop, *Pattern Recognition and Machine Learning*

Logistic Regression: Beyond Binary

- We started with this conditional distribution:

$$P_\theta(y = 1|x) = \sigma(\theta^T x) = \frac{1}{1 + \exp(-\theta^T x)}$$

- Now let us try to extend it to multi-class classification, $y \in \{1, \ldots, k\}$.
  - Can no longer just use one $\theta^T x$
  - But we can try multiple…

Logistic Regression: Beyond Binary

- Let's set, for *y* in *1,2,…,k*

$$P_\theta(y = i|x) = \frac{\exp((\theta^i)^T x)}{\sum_{j=1}^{k} \exp((\theta^j)^T x)}$$

- Note: we have several weight vectors now (one per class).
- To train, same as before (just more weight vectors).

$$\min_\theta -\frac{1}{n} \sum_{i=1}^{n} \log P_\theta(y^{(i)}|x^{(i)})$$

Cross-Entropy Loss

- Define $q^{(i)}$ as the one-hot vector for the ith datapoint's label.

$$q_j^{(i)} = P(y = j \,|x^{(i)})$$

- Next, let's let $p^{(i)} = P_\theta(y|x^{(i)})$ be our prediction distribution (over y)

- Our loss terms can be written

Note: only 1 term non-zero.

$$-\log p(y^{(i)}|x^{(i)}) = -\sum_{j=1}^{k} q_j^{(i)} \log p(y = j|x^{(i)})$$

Looks like the entropy, but …

- This is the "cross-entropy" $H(q^{(i)}, p^{(i)})$

Cross-Entropy Loss

- This is the "cross-entropy"

$$H(q^{(i)}, p^{(i)}) = \mathbb{E}_{q^{(i)}}[\log p^{(i)}]$$

- What are we doing when we minimize the cross-entropy?
- Recall KL divergence,

$$D(q^{(i)}||p^{(i)}) = \underbrace{\mathbb{E}_{q^{(i)}}[\log p^{(i)}]}_{\text{Cross-entropy}} - \underbrace{\mathbb{E}_{q^{(i)}}[\log q^{(i)}]}_{\text{Entropy } H(q^{(i)}) \text{ (fixed)}}$$

- Matching distributions!

Softmax

- We wrote

$$P_\theta(y = i|x) = \frac{\exp((\theta^i)^T x)}{\sum_{j=1}^{k} \exp((\theta^j)^T x)}$$

- This operation is called softmax.
  - Converts a vector into a probability vector (note normalization).
  - If one component in the vector ***a*** is **dominant**, softmax(***a***) is close to the one-hot vector picking out that maximum component

Quiz

Q1: Calculate the softmax of (1, 2, 3, 4, 5):

1. (0, 0.145, 0.229, 0.290, 0.336)
2. (0.012, 0.032, 0.086, 0.234, 0.636) ⟵
3. (0.636, 0.234, 0.086, 0.032, 0.012)

$$softmax(a)_i = \frac{\exp(a_i)}{\sum_j \exp(a_j)}$$

Q2: True or false

Logistic regression is a discriminative model because we obtain $P_\theta(y|x)$.

False! We also obtain this from a generative model. Logistic regression is discriminative because it does not learn the joint distribution over (x,y).

Outline

- **Parametric modeling**
  - Comparison with non-parametric models
  - Generative vs. discriminative modeling
- **Logistic regression**
  - Maximum likelihood estimation
  - Setup
- **Naïve Bayes**
  - Bayesian modeling, MAP vs. MLE
  - Motivation / training / inference / smoothing
  - Examples (Bernoulli, Multiclass, Gaussian)

Review: **Maximum Likelihood**

- For some set of data, find the parameters that maximize the likelihood / log-likelihood

$$\hat{\theta} = \arg\max_{\theta} \mathcal{L}(\theta; X)$$

- Example: suppose we have n samples from a Bernoulli distribution

$$P_\theta(X = x) = \begin{cases} \theta & x = 1 \\ 1 - \theta & x = 0 \end{cases}$$

Then,

$$\mathcal{L}(\theta; X) = \prod_{i=1}^{n} P(X = x_i) = \theta^k (1 - \theta)^{n-k}$$

Review: **Maximum Likelihood**

- For some set of data, find the parameters that maximize the likelihood / log-likelihood
- Example: exponential distribution

  - pdf of Exponential(λ): $f(x) = \lambda e^{-\lambda x}$
  - Suppose $X_i \sim$ Exponential(λ) for $1 \leq i \leq N$.
  - Find MLE for data $\mathcal{D} = \{x^{(i)}\}_{i=1}^{N}$
  - First write down log-likelihood of sample.
  - Compute first derivative, set to zero, solve for λ.
  - Compute second derivative and check that it is concave down at λ^{MLE}.

Review: **Maximum Likelihood**

- Example: exponential distribution
  - First write down log-likelihood of sample.

$$\ell(\lambda) = \sum_{i=1}^{N} \log f(x^{(i)}) \tag{1}$$

$$= \sum_{i=1}^{N} \log(\lambda \exp(-\lambda x^{(i)})) \tag{2}$$

$$= \sum_{i=1}^{N} \log(\lambda) + -\lambda x^{(i)} \tag{3}$$

$$= N \log(\lambda) - \lambda \sum_{i=1}^{N} x^{(i)} \tag{4}$$

Review: **Maximum Likelihood**

- Example: exponential distribution

  - Compute first derivative, set to zero, solve for λ.

$$\frac{d\ell(\lambda)}{d\lambda} = \frac{d}{d\lambda} N \log(\lambda) - \lambda \sum_{i=1}^{N} x^{(i)} \quad (1)$$

$$= \frac{N}{\lambda} - \sum_{i=1}^{N} x^{(i)} = 0 \quad (2)$$

$$\Rightarrow \lambda^{\text{MLE}} = \frac{N}{\sum_{i=1}^{N} x^{(i)}} \quad (3)$$

Another Approach: **Bayesian Inference**

- Let's consider a different approach
- Need a little bit of terminology

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

- *H* is the hypothesis
- *E* is the evidence

Bayesian Inference Definitions

- Terminology:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)} \longleftarrow \textbf{Prior}$$

- Prior: estimate of the probability **without** evidence

Bayesian Inference Definitions

- Terminology:

Likelihood

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

- Likelihood: probability of evidence **given a hypothesis**.
  - Compare to the way we defined the likelihood earlier

Bayesian Inference Definitions

- Terminology:

$$P(H|E) = \frac{P(E|H)P(H)}{P(E)}$$

↑

Posterior

- Posterior: probability of hypothesis **given evidence**.

MAP Definition

- Suppose we think of the parameters as random variables
  - There is a prior $P(\theta)$

- Then, can do learning as Bayesian inference
  - "Evidence" is the data

$$P(\theta|X) = \frac{P(X|\theta)P(\theta)}{P(X)}$$

  - **Maximum a posteriori probability** (**MAP**) estimation

$$\theta^{\mathrm{MAP}} = \arg\max_{\theta} \prod_{i=1}^{n} p(x^{(i)}|\theta)p(\theta)$$

MAP vs ML

- What's the difference between ML and MAP?

$$\theta^{\mathrm{MLE}} = \arg\max_{\theta} \prod_{i=1}^{n} p(x^{(i)}|\theta)$$

$$\theta^{\mathrm{MAP}} = \arg\max_{\theta} \prod_{i=1}^{n} p(x^{(i)}|\theta)p(\theta)$$

- **the prior!**

Outline

- **Parametric modeling**
  - Comparison with non-parametric models
  - Generative vs. discriminative modeling
- **Logistic regression**
  - Maximum likelihood estimation
  - Setup
- **Naïve Bayes**
  - Bayesian modeling, MAP vs. MLE
  - **Motivation / training / inference / smoothing**
  - **Examples (Bernoulli, Multiclass, Gaussian)**

Application: Parody Detection

- The Economist
- The Onion

La paralización

Spain may be heading for its third election in a year

All latest updates

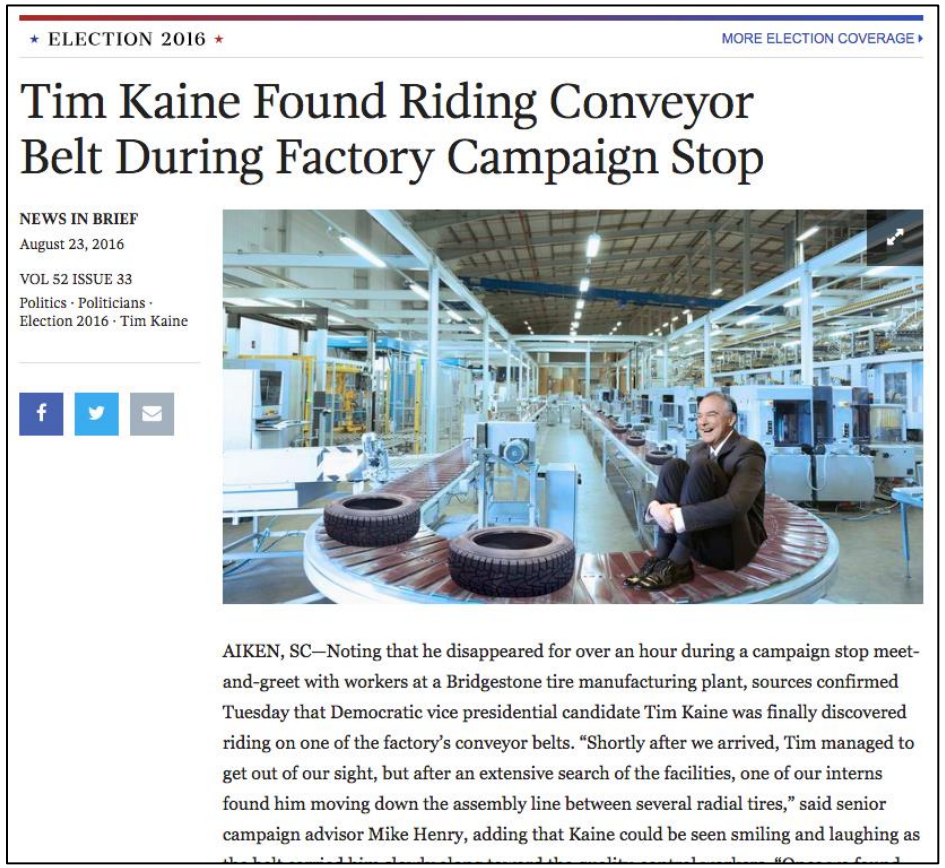
Stubborn Socialists are blocking Mariano Rajoy from forming a centre-right government

Sep 5th 2016 | MADRID | Europe

Timekeeper Like 80 Tweet

EPA

BACK in June, after Spain's second indecisive election in six months, the general expectation was that Mariano Rajoy, the prime minister, would swiftly form a new government. Although his conservative People's Party (PP) did not win back the absolute majority it had lost in December, it remained easily the largest party, with 137 of the 350



★ ELECTION 2016 ★

MORE ELECTION COVERAGE ▸

Tim Kaine Found Riding Conveyor Belt During Factory Campaign Stop

NEWS IN BRIEF

August 23, 2016

VOL 52 ISSUE 33

Politics · Politicians · Election 2016 · Tim Kaine

AIKEN, SC—Noting that he disappeared for over an hour during a campaign stop meet-and-greet with workers at a Bridgestone tire manufacturing plant, sources confirmed Tuesday that Democratic vice presidential candidate Tim Kaine was finally discovered riding on one of the factory's conveyor belts. "Shortly after we arrived, Tim managed to get out of our sight, but after an extensive search of the facilities, one of our interns found him moving down the assembly line between several radial tires," said senior campaign advisor Mike Henry, adding that Kaine could be seen smiling and laughing as

Model 0: Not-Naïve Model

Generative story:

1. Flip a weighted coin (Y)
2. If heads, sample a document ID (X) from the parody distribution
3. If tails, sample a document ID (X) from the regular distribution

$$P(X, Y) = P(X|Y)P(Y)$$

Model 0: Not-Naïve Model

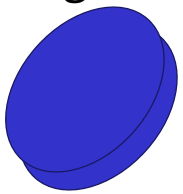
Generative story:

1. Flip a weighted coin (Y)
2. If heads, roll the **yellow** many sided die to sample a document vector ($\boldsymbol{X}$) from the parody distribution
3. If tails, roll the **blue** many sided die to sample a document vector ($\boldsymbol{X}$) from the regular distribution

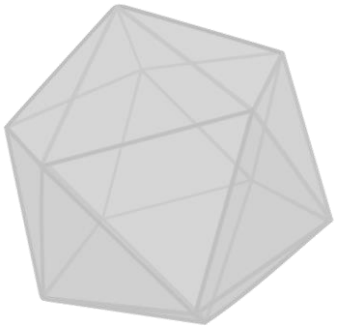
$$P(X_1, \ldots, X_K, Y) = P(X_1, \ldots, X_K|Y)P(Y)$$

Model 0: Not-Naïve Model

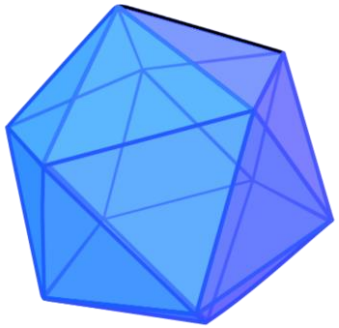
Flip weighted coin



If HEADS, roll yellow die



If TAILS, roll blue die



y	x_1	x_2	x_3	…	x_K
0	1	0	1	…	1
1	0	1	0	…	1
1	1	1	1	…	1
0	0	0	1	…	1
0	1	0	1	…	0
1	1	0	1	…	0

Each side of the die is labeled with a document vector (e.g. [1,0,1,…,1])

Model 0: Main Problem

How many terms are we modeling?

- Say features are binary: $X_i \in \{0, 1\}$ e.g. is word i in the document?

$$P(X_1, \ldots, X_K | Y)$$

- 2^k choices of feature vector, each gets its own probability…
  - Exponentially big table (e.g. in vocabulary size)

Naïve Bayes: Core Assumption

How do we fix this problem?

- Conditional **independence** of features:

$$P(X_1, \ldots, X_K, Y) = P(X_1, \ldots, X_K | Y) P(Y)$$

$$= \left(\prod_{k=1}^{K} P(X_k | Y) \right) P(Y)$$

- What do we gain? With binary features, get 2 entries per feature
- So, number of probabilities

$$2^k \rightarrow 2k$$

Naïve Bayes: Overall Model

Support of $P(X_k|Y)$ depends on the application

Model: Product of **class prior** and the event model

$$P(\mathbf{X}, Y) = P(Y) \prod_{k=1}^{K} P(X_k|Y)$$

Training: Find the **class-conditional** MLE parameters:

- For prior $P(Y)$, we find the MLE using the data.
- For each $P(X_k|Y)$ we condition on the data with the corresponding class.

Prediction: Find the class that maximizes the posterior

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, p(y|\mathbf{x})$$

Naïve Bayes: Smoothing

- **Training**: empirically estimate the probabilities
  - Could we just use counts to obtain the probabilities?

$$P(X_k = i|Y) = \frac{\#\text{times } X_k = i \text{ in class } Y}{\#\text{examples of class } Y} = \frac{c_i}{N}$$

  - But what if $c_i = 0$? Then $\mathrm{P}(\mathrm{X_k} = \mathrm{i}|\mathrm{Y}) = 0$, which will make predictions using the event model zero $(= P(Y)\prod_k \mathrm{P}(\mathrm{X_k} = \mathrm{i}|\mathrm{Y}))$.

  - Solution: smooth! $\hat{P}(X_k|Y) = \frac{c_i + \alpha}{N + \alpha m}$ ← Smoothing parameter

Naïve Bayes: Predicting

- With conditional probabilities, how to predict?

$$\hat{y} = \underset{y}{\operatorname{argmax}}\, p(y|\mathbf{x}) \quad \text{(posterior)}$$

$$= \underset{y}{\operatorname{argmax}}\, \frac{p(\mathbf{x}|y)p(y)}{p(x)} \quad \text{(by Bayes' rule)}$$

$$= \underset{y}{\operatorname{argmax}}\, p(\mathbf{x}|y)p(y)$$

Naïve Bayes Example 1: Bernoulli

Support: Binary vectors of length K

$$\mathbf{x} \in \{0, 1\}^K$$

Generative Story:

$$Y \sim \text{Bernoulli}(\phi)$$

$$X_k \sim \text{Bernoulli}(\theta_{k,Y}) \ \forall k \in \{1, \ldots, K\}$$

Model:

$$\begin{aligned} p_{\phi,\boldsymbol{\theta}}(\boldsymbol{x}, y) &= p_{\phi,\boldsymbol{\theta}}(x_1, \ldots, x_K, y) \\ &= p_\phi(y) \prod_{k=1}^{K} p_{\boldsymbol{\theta}_k}(x_k|y) \\ &= (\phi)^y (1-\phi)^{(1-y)} \prod_{k=1}^{K} (\theta_{k,y})^{x_k} (1-\theta_{k,y})^{(1-x_k)} \end{aligned}$$

Training Bernoulli Naïve Bayes

Recall: train (by MLE) is to find **class-conditional** parameters

- To find P(Y): use all the data
- For $P(X_i|Y=y)$: use the data for that class

$$\phi = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}{N}$$

$$\theta_{k,0} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0 \wedge x_k^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 0)}$$

$$\theta_{k,1} = \frac{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1 \wedge x_k^{(i)} = 1)}{\sum_{i=1}^{N} \mathbb{I}(y^{(i)} = 1)}$$

$$\forall k \in \{1, \ldots, K\}$$

Naïve Bayes Example 2: Multinomial

Integer vector (word IDs)

$\mathbf{x} = [x_1, x_2, \ldots, x_M]$ where $x_m \in \{1, \ldots, K\}$ a word id.

Generative Story:

for $i \in \{1, \ldots, N\}$**:**

$y^{(i)} \sim \text{Bernoulli}(\phi)$

for $j \in \{1, \ldots, M_i\}$**:**

$x_j^{(i)} \sim \text{Multinomial}(\boldsymbol{\theta}_{y^{(i)}}, 1)$

Model:

$$p_{\phi,\boldsymbol{\theta}}(\boldsymbol{x}, y) = p_\phi(y) \prod_{k=1}^{K} p_{\boldsymbol{\theta}_k}(x_k|y)$$

$$= (\phi)^y (1-\phi)^{(1-y)} \prod_{j=1}^{M_i} \theta_{y,x_j}$$

Naïve Bayes Example 3: Gaussian

Support: $\mathbf{x} \in \mathbb{R}^K$

Model: Product of **prior** and the event model

$$p(\boldsymbol{x}, y) = p(x_1, \ldots, x_K, y)$$

$$= p(y) \prod_{k=1}^{K} p(x_k|y)$$

Gaussian Naive Bayes assumes that $p(x_k|y)$ is given by a Normal distribution.

Q2-1: Are these statements true or false?
(A) Naïve Bayes assumes conditional independence of features to decompose the joint probability into the conditional probabilities.
(B) We use the Bayes' rule to calculate the posterior probability.

1. True, True
2. True, False
3. False, True
4. False, False

Q2-1: Are these statements true or false?
(A) Naïve Bayes assumes conditional independence of features to decompose the joint probability into the conditional probabilities.
(B) We use the Bayes' rule to calculate the posterior probability.

1. True, True ⟵
2. True, False
3. False, True
4. False, False

(A) Just as we learnt in the lecture.
(B) We use Bayes rule to decompose posterior probability into prior probability and conditional probability given each class, so that we can compute it using the estimated parameters.

Thanks Everyone!

Some of the slides in these lectures have been adapted/borrowed from materials developed by Mark Craven, David Page, Jude Shavlik, Tom Mitchell, Nina Balcan, Elad Hazan, Tom Dietterich, Pedro Domingos, Jerry Zhu, Yingyu Liang, Volodymyr Kuleshov, Fred Sala, Tengyang Xie