CSC 311: Introduction to Machine Learning

Lecture 7 - Probabilistic Models

Sayyed Nezhadi

University of Toronto, Summer 2023

Outline

1. Probabilistic Modeling of Data
2. Discriminative and Generative Classifiers
3. Naïve Bayes Models
4. Bayesian Parameter Estimation

Today

- So far in the course we have adopted a modular perspective, in which the model, loss function, optimizer, and regularizer are specified separately.
- Today we begin putting together a probabilistic interpretation of our model and loss, and introduce the concept of maximum likelihood estimation.

1. Probabilistic Modeling of Data
2. Discriminative and Generative Classifiers
3. Naïve Bayes Models
4. Bayesian Parameter Estimation

Example: A Biased Coin

You flip a coin $N = 100$ times and get outcomes $\{x_1, \ldots, x_N\}$ where $x_i \in \{0, 1\}$ and $x_i = 1$ is interpreted as heads H.

Suppose you had $N_H = 55$ heads and $N_T = 45$ tails.

We want to create a model to predict the outcome of the next coin flip. That is, we want to answer this question:

What is the probability it will come up heads if we flip again?

Model

The coin is likely biased. Let's assume that one coin flip outcome x is a Bernoulli random variable for *a currently unknown parameter* $\theta \in [0, 1]$.

$$p(x = 1|\theta) = \theta \quad \text{and} \quad p(x = 0|\theta) = 1 - \theta$$
$$\text{or more succinctly} \quad p(x|\theta) = \theta^x (1 - \theta)^{1-x}$$

Assume that $\{x_1, \ldots, x_N\}$ are independent and identically distributed (i.i.d.). Thus, the joint probability of the outcome $\{x_1, \ldots, x_N\}$ is

$$p(x_1, ..., x_N|\theta) = \prod_{i=1}^{N} \theta^{x_i} (1 - \theta)^{1-x_i}$$

Loss Function

The likelihood function is the probability of observing the data as a function of the parameters θ:

$$L(\theta) = \prod_{i=1}^{N} \theta^{x_i} (1 - \theta)^{1 - x_i}$$

We usually work with log-likelihoods:

$$\ell(\theta) = \sum_{i=1}^{N} x_i \log \theta + (1 - x_i) \log(1 - \theta)$$

Maximum Likelihood Estimation

How can we choose θ? Good values of θ should assign high probability to the observed data.

The maximum likelihood criterion says that we should pick the parameters that maximize the likelihood.

$$\hat{\theta}_{\mathrm{ML}} = \arg\max_{\theta \in [0,1]} \ell(\theta)$$

We can find the optimal solution by setting derivatives to zero.

$$\frac{\mathrm{d}\ell}{\mathrm{d}\theta} = \frac{\mathrm{d}}{\mathrm{d}\theta}\left(\sum_{i=1}^{N} x_i \log\theta + (1 - x_i)\log(1-\theta)\right) = \frac{N_H}{\theta} - \frac{N_T}{1-\theta}$$

where $N_H = \sum_i x_i$ and $N_T = N - \sum_i x_i$.

Setting this to zero gives the maximum likelihood estimate:

$$\hat{\theta}_{\mathrm{ML}} = \frac{N_H}{N_H + N_T}.$$

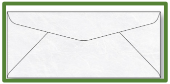
Maximum Likelihood Estimation

- define a model that assigns a probability (or has a probability density at) to a dataset
- maximize the likelihood (or minimize the neg. log-likelihood).

1. Probabilistic Modeling of Data
2. Discriminative and Generative Classifiers
3. Naïve Bayes Models
4. Bayesian Parameter Estimation

Spam Classification

For a large company that runs an email service, one of the important predictive problems is the automated detection of spam email.

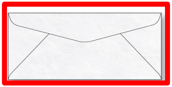


Dear Karim,

I think we should postpone the board meeting to be held after Thanksgiving.

Regards,
Anna

Not spam



Dear Toby,

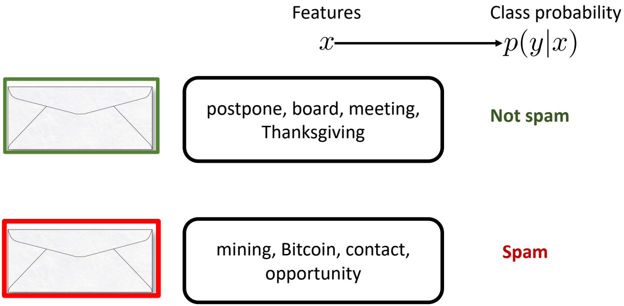
I have an incredible opportunity for mining 2 Bitcoin a day. Please Contact me at the earliest at +1 123 321 1555. You won't want to miss out on this opportunity.

Regards,
Ark

Spam

Discriminative Classifiers

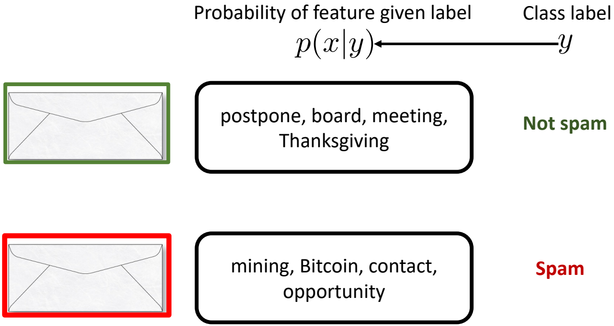
Discriminative classifiers try to learn mappings directly from the space of inputs $\mathcal{X}$ to class labels $\{0, 1, 2, \ldots, K\}$



Generative Classifiers

Generative classifiers try to build a model of "what data for a class looks like", i.e. model $p(\mathbf{x}, y)$. If we know $p(y)$ we can easily compute $p(\mathbf{x}|y)$.

Classification via Bayes rule (thus also called Bayes classifiers)



Generative vs Discriminative

- Discriminative approach: estimate parameters of decision boundary/class separator directly from labeled examples.
  - Model $p(t|\mathbf{x})$ directly (logistic regression models)
  - Learn mappings from inputs to classes (linear/logistic regression, decision trees etc)
  - Tries to solve: How do I separate the classes?
- Generative approach: model the distribution of inputs characteristic of the class (Bayes classifier).
  - Model $p(\mathbf{x}|t)$
  - Apply Bayes Rule to derive $p(t|\mathbf{x})$.
  - Tries to solve: What does each class "look" like?
- Key difference: is there a distributional assumption over inputs?

1 Probabilistic Modeling of Data

2 Discriminative and Generative Classifiers

3 Naïve Bayes Models

4 Bayesian Parameter Estimation

Example: Spam Detection

- Classify email into spam ($c = 1$) or non-spam ($c = 0$).
- Binary features $\mathbf{x} = [x_1, \ldots, x_D], x_i \in \{0, 1\}$ saying whether each of D words appears in the e-mail.

Example email: "You are one of the very few who have been selected as a winner for the free $1000 Gift Card."

Feature vector for this email:

- ...
- "card": 1
- ...
- "winners": 1
- "winter": 0
- ...
- "you": 1

Bayesian Classifier

Given features $\mathbf{x} = [x_1, x_2, \cdots, x_D]^T$
want to compute class probabilities using Bayes Rule:

$$\underbrace{p(c|\mathbf{x})}_{\text{Pr. class given feature}} = \frac{\overbrace{p(\mathbf{x}|c)}^{\text{Pr. feature given class}} \quad p(c)}{p(\mathbf{x})}$$

In words,

$$\text{Posterior for class} = \frac{\text{Pr. of feature given class} \times \text{Prior for class}}{\text{Pr. of feature}}$$

To compute $p(c|\mathbf{x})$ we need: $p(\mathbf{x}|c)$ and $p(c)$.

Motivation for Compact Representation

- Two classes: $c \in \{0, 1\}$.
- Binary features $\mathbf{x} = [x_1, \dots, x_D], x_i \in \{0, 1\}$

- Define a joint distribution $p(c, x_1, \dots, x_D)$.
  How many probabilities do we need to specify this joint dist.?

- Let's impose structure on the distribution so that
  the representation is compact and
  allows for efficient learning and inference

Naïve Bayes Independence Assumption

Naïve assumption:
the features x_i are conditionally independent given the class c.

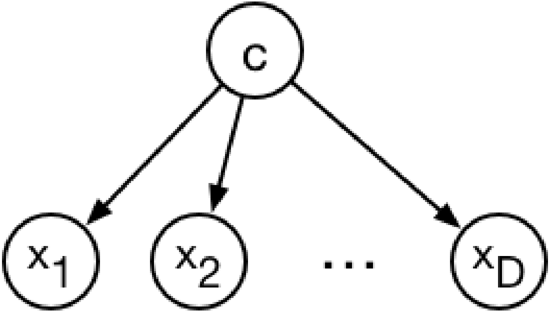
- Allows us to decompose the joint distribution:

$$p(c, x_1, \ldots, x_D) = p(c)\, p(x_1|c) \cdots p(x_D|c).$$

Compact representation of the joint distribution

- Prior probability of class:
  $p(c = 1) = \pi$ (e.g. prob of spam)
- Conditional probability of feature given class:
  $p(x_j = 1|c) = \theta_{jc}$ (e.g. prob of word appearing in spam)

Bayesian Network for a Naive Bayes Model



- Which probabilities do we need to specify this dist.?
- How many probabilities do we need to specify this dist.?

Decomposing the Log-Likelihood

Decompose the log-likelihood into independent terms.
Optimize each term independently.

$$\begin{aligned}
\ell(\boldsymbol{\theta}) &= \sum_{i=1}^{N} \log p(c^{(i)}, \mathbf{x}^{(i)}) = \sum_{i=1}^{N} \log \left\{ p(\mathbf{x}^{(i)} | c^{(i)}) p(c^{(i)}) \right\} \\
&= \sum_{i=1}^{N} \log \left\{ p(c^{(i)}) \prod_{j=1}^{D} p(x_j^{(i)} \mid c^{(i)}) \right\} \\
&= \sum_{i=1}^{N} \left[\log p(c^{(i)}) + \sum_{j=1}^{D} \log p(x_j^{(i)} \mid c^{(i)}) \right] \\
&= \underbrace{\sum_{i=1}^{N} \log p(c^{(i)})}_{\substack{\text{Log-likelihood} \\ \text{of labels}}} + \sum_{j=1}^{D} \underbrace{\sum_{i=1}^{N} \log p(x_j^{(i)} \mid c^{(i)})}_{\substack{\text{Log-likelihood} \\ \text{for feature } x_j}}
\end{aligned}$$

Learning the Prior over Class

- To learn the prior, we maximize $\sum_{i=1}^{N} \log p(c^{(i)})$
- Define $\pi = p(c^{(i)} = 1)$
- Pr. i-th email: $p(c^{(i)}) = \pi^{c^{(i)}}(1-\pi)^{1-c^{(i)}}$.
- Log-likelihood of the dataset:

$$\sum_{i=1}^{N} \log p(c^{(i)}) = \sum_{i=1}^{N} c^{(i)} \log \pi + \sum_{i=1}^{N} (1 - c^{(i)}) \log(1-\pi)$$

- Maximum likelihood estimate of the prior π is the fraction of spams in dataset.

$$\hat{\pi} = \frac{\sum_i \mathbb{1}[c^{(i)} = 1]}{N} = \frac{\text{\# spams in dataset}}{\text{total \# samples}}$$

Learning Pr. Feature Given Class

- To learn $p(x_j^{(i)} = 1 \,|\, c)$, we maximize $\sum_{i=1}^{N} \log p(x_j^{(i)} \,|\, c^{(i)})$
- Define $\theta_{jc} = p(x_j^{(i)} = 1 \,|\, c)$.
- Pr. of i-th email: $p(x_j^{(i)} \,|\, c) = \theta_{jc}^{x_j^{(i)}} (1 - \theta_{jc})^{1 - x_j^{(i)}}$.
- Log-likelihood of the dataset:

$$\sum_{i=1}^{N} \log p(x_j^{(i)} \,|\, c^{(i)}) = \sum_{i=1}^{N} c^{(i)} \left\{ x_j^{(i)} \log \theta_{j1} + (1 - x_j^{(i)}) \log(1 - \theta_{j1}) \right\}$$
$$+ \sum_{i=1}^{N} (1 - c^{(i)}) \left\{ x_j^{(i)} \log \theta_{j0} + (1 - x_j^{(i)}) \log(1 - \theta_{j0}) \right\}$$

- Maximum likelihood estimate of θ_{jc}
  is the fraction of word j occurrances in each class in the dataset.

$$\hat{\theta}_{jc} = \frac{\sum_i \mathbb{1}[x_j^{(i)} = 1 \ \& \ c^{(i)} = c]}{\sum_i \mathbb{1}[c^{(i)} = c]} \overset{\text{for } c = 1}{=} \frac{\#\text{word } j \text{ appears in class } c}{\# \text{ class } c \text{ in dataset}}$$

Predicting the Most Likely Class

- We predict the class by performing inference in the model.
- Apply Bayes' Rule:

$$p(c \,|\, \mathbf{x}) = \frac{p(c)p(\mathbf{x} \,|\, c)}{\sum_{c'} p(c')p(\mathbf{x} \,|\, c')} = \frac{p(c) \prod_{j=1}^{D} p(x_j \,|\, c)}{\sum_{c'} p(c') \prod_{j=1}^{D} p(x_j \,|\, c')}$$

- For input $\mathbf{x}$, predict c with the largest $p(c) \displaystyle\prod_{j=1}^{D} p(x_j \,|\, c)$

  (the most likely class).

$$p(c \,|\, \mathbf{x}) \propto p(c) \prod_{j=1}^{D} p(x_j \,|\, c)$$

Naïve Bayes Properties

- An amazingly cheap learning algorithm!
- Training time: estimate parameters using maximum likelihood
  - Compute co-occurrence counts of each feature with the labels.
  - Requires only one pass through the data!
- Test time: apply Bayes' Rule
  - Cheap because of the model structure. (For more general models, Bayesian inference can be very expensive and/or complicated.)
- Analysis easily extends to prob. distributions other than Bernoulli.
- Less accurate in practice compared to discriminative models due to its "naïve" independence assumption.

1 Probabilistic Modeling of Data

2 Discriminative and Generative Classifiers

3 Naïve Bayes Models

4 Bayesian Parameter Estimation

Data Sparsity

Maximum likelihood can overfit if there is too little data.

Example: what if you flip the coin twice and get H both times?

$$\theta_{\text{ML}} = \frac{N_H}{N_H + N_T} = \frac{2}{2+0} = 1$$

The model assigned probability 0 to T.
This problem is known as data sparsity.

Defining a Bayesian Model

We need to specify two distributions:

- The prior distribution $p(\boldsymbol{\theta})$
  encodes our beliefs about the parameters
  before we observe the data.

- The likelihood $p(\mathcal{D} \mid \boldsymbol{\theta})$
  encodes the likelihood of observing the data
  given the parameters.

The Posterior Distribution

- When we update our beliefs based on the observations, we compute the posterior distribution using Bayes' Rule:

$$p(\boldsymbol{\theta} \mid \mathcal{D}) = \frac{p(\boldsymbol{\theta})p(\mathcal{D} \mid \boldsymbol{\theta})}{\int p(\boldsymbol{\theta}')p(\mathcal{D} \mid \boldsymbol{\theta}') \, \mathrm{d}\boldsymbol{\theta}'}.$$

- Rarely ever compute the denominator explicitly.
- In general, computing the denominator is intractable.

Revisiting Coin Flip Example

We already know the likelihood:

$$L(\theta) = p(\mathcal{D}|\theta) = \theta^{N_H}(1-\theta)^{N_T}$$

It remains to specify the prior $p(\theta)$.

- An uninformative prior, which assumes as little as possible.
  A reasonable choice is the uniform prior.
- But, experience tells us 0.5 is more likely than 0.99.
  One particularly useful prior is the beta distribution:

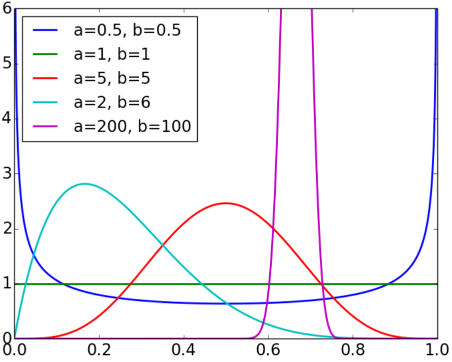
$$p(\theta; a, b) = \frac{\Gamma(a+b)}{\Gamma(a)\Gamma(b)}\,\theta^{a-1}(1-\theta)^{b-1}.$$

- We can ignore the normalization constant.

$$p(\theta; a, b) \propto \theta^{a-1}(1-\theta)^{b-1}.$$

Beta Distribution Properties

- The expectation is $\mathbb{E}[\theta] = a/(a+b)$.
- The distribution gets more peaked when a and b are large.
- When $a = b = 1$, it becomes the uniform distribution.



Posterior for the Coin Flip Example

- Computing the posterior distribution:

$$\begin{aligned} p(\boldsymbol{\theta} \mid \mathcal{D}) &\propto p(\boldsymbol{\theta}) p(\mathcal{D} \mid \boldsymbol{\theta}) \\ &\propto \left[\theta^{a-1}(1-\theta)^{b-1}\right] \left[\theta^{N_H}(1-\theta)^{N_T}\right] \\ &= \theta^{a-1+N_H}(1-\theta)^{b-1+N_T}. \end{aligned}$$

  A beta distribution with parameters $N_H + a$ and $N_T + b$.

- The posterior expectation of θ is:

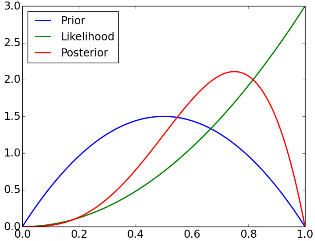
$$\mathbb{E}[\theta \mid \mathcal{D}] = \frac{N_H + a}{N_H + N_T + a + b}$$

- Think of a and b as pseudo-counts.
  beta(a, b) = beta$(1, 1)$ + $a - 1$ heads + $b - 1$ tails.
- The prior and likelihood have the same functional form (conjugate priors).

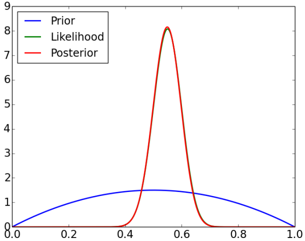
Bayesian Inference for the Coin Flip Example

When you have enough observations, the data overwhelm the prior.

Small data setting
$N_H = 2,\ N_T = 0$

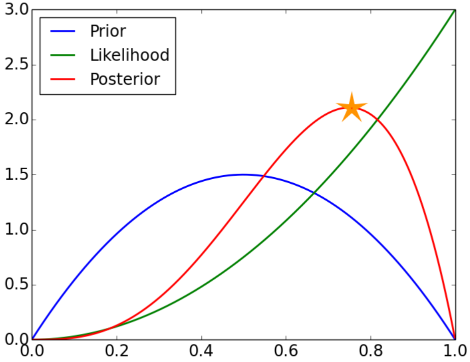


Large data setting
$N_H = 55,\ N_T = 45$



Maximum A-Posteriori (MAP) Estimation

Finds the most likely parameters under the posterior (i.e. the mode).



Maximum A-Posteriori Estimation

Converts the Bayesian parameter estimation problem into a maximization problem

$$\begin{aligned}
\hat{\boldsymbol{\theta}}_{\text{MAP}} &= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta} \mid \mathcal{D}) \\
&= \arg\max_{\boldsymbol{\theta}} \; p(\boldsymbol{\theta}) \, p(\mathcal{D} \mid \boldsymbol{\theta}) \\
&= \arg\max_{\boldsymbol{\theta}} \; \log p(\boldsymbol{\theta}) + \log p(\mathcal{D} \mid \boldsymbol{\theta})
\end{aligned}$$

Maximum A-Posteriori Estimation

Joint probability of parameters and data:

$$\begin{aligned}\log p(\theta, \mathcal{D}) &= \log p(\theta) + \log p(\mathcal{D} \,|\, \theta) \\ &= \text{Const} + (N_H + a - 1)\log\theta + (N_T + b - 1)\log(1-\theta)\end{aligned}$$

Maximize by finding a critical point

$$\frac{\mathrm{d}}{\mathrm{d}\theta}\log p(\theta, \mathcal{D}) = \frac{N_H + a - 1}{\theta} - \frac{N_T + b - 1}{1-\theta} = 0$$

Solving for θ,

$$\hat{\theta}_{\text{MAP}} = \frac{N_H + a - 1}{N_H + N_T + a + b - 2}$$

Estimate Comparison for Coin Flip Example

	Formula	$N_H = 2, N_T = 0$	$N_H = 55, N_T = 45$
$\hat{\theta}_{\text{ML}}$	$\frac{N_H}{N_H+N_T}$	1	$\frac{55}{100} = 0.55$
$\mathbb{E}[\theta \vert \mathcal{D}]$	$\frac{N_H+a}{N_H+N_T+a+b}$	$\frac{4}{6} \approx 0.67$	$\frac{57}{104} \approx 0.548$
$\hat{\theta}_{\text{MAP}}$	$\frac{N_H+a-1}{N_H+N_T+a+b-2}$	$\frac{3}{4} = 0.75$	$\frac{56}{102} \approx 0.549$

$\hat{\theta}_{\text{MAP}}$ assigns nonzero probabilities as long as $a, b > 1$.