Outline

1 Administration
2 Multiclass classification
3 Generative versus discriminative
4 Perceptron
A few announcements

- Homework 1: due 9/24 (see the homework sheets for detailed submission information)
- I’m out of town in Thu class. Dear TA Wenzhe Li will teach the class
Outline

1. Administration

2. Multiclass classification
   - Multinomial logistic regression

3. Generative versus discriminative

4. Perceptron
Suppose we need to predict multiple classes/outcomes:
\( C_1, C_2, \ldots, C_K \)
- Weather prediction: sunny, cloudy, raining, etc
- Optical character recognition: 10 digits + 26 characters (lower and upper cases) + special characters, etc

**Studied methods**
- Nearest neighbor classifier
- Naive Bayes
- Gaussian discriminant analysis
- Logistic regression
Contrast these two approaches

Pros and cons of each approach

- **one versus the rest**: only needs to train $K$ classifiers. Make a *huge* difference if you have a lot of *classes* to go through. Can you think of a good application example where there are a lot of classes?

- **one versus one**: only needs to train a smaller subset of data (only those labeled with those two classes would be involved). Make a *huge* difference if you have a lot of *data* to go through.

Bad about both of them

*Combining classifiers’ outputs seem to be a bit tricky.*

Any other good methods?
Multinomial logistic regression

**Intuition: from the decision rule of our naive Bayes classifier**

\[
y^* = \arg \max_c p(y = c | x) = \arg \max_c \log p(x | y = c) p(y = c) \tag{1}
\]

\[
= \arg \max_c \log \pi_c + \sum_k z_k \log \theta_{ck} = \arg \max_c \mathbf{w}_c^T \mathbf{x} \tag{2}
\]

**Essentially, we are comparing**

\[
\mathbf{w}_1^T \mathbf{x}, \mathbf{w}_2^T \mathbf{x}, \ldots, \mathbf{w}_C^T \mathbf{x} \tag{3}
\]

with *one* for each category.
First try

So, can we define the following conditional model?

$$p(y = c | x) = \sigma [w_c^T x]$$

This would *not* work at least for the reason

$$\sum_c p(y = c | x) = \sum_c \sigma [w_c^T x] \neq 1$$

as each the summand can be any number (independently) between 0 and 1. *But we are close*
Definition of multinomial logistic regression

Model

For each class $C_k$, we have a parameter vector $w_k$ and model the posterior probability as

$$p(C_k|x) = \frac{e^{w_k^T x}}{\sum_{k'} e^{w_{k'}^T x}} \quad \leftarrow \quad \text{This is called softmax function}$$

Decision boundary: assign $x$ with the label that is the maximum of posterior

$$\arg \max_k P(C_k|x) \rightarrow \arg \max_k w_k^T x$$

Note: the notation is changed to denote the classes as $C_k$ instead of just $c$
**Why the name softmax?**

Suppose we have

\[ w_1^T x = 100, \quad w_2^T x = 50, \quad w_3^T x = -20 \]

we could have picked the *winning* class label 1 with certainty according to our classification rule.

Softness comes in when we compute the probability of selecting that

\[ p(y = 1|x) = \frac{e^{100}}{e^{100} + e^{50} + e^{-20}} < 1 \]

despite its begin the largest among the 3 \( p(y = 1|x) > p(y = 2|x) \) and \( p(y = 1|x) > p(y = 2|x) \). Thus the name *softmax*
Sanity check

**Multinomial model reduce to binary logistic regression** when $K = 2$

$$p(C_1|x) = \frac{e^{w_1^T x}}{e^{w_1^T x} + e^{w_2^T x}} = \frac{1}{1 + e^{-(w_1-w_2)^T x}}$$

*Multinomial thus generalizes the (binary) logistic regression to deal with multiple classes.*
Parameter estimation

**Discriminative approach:** maximize conditional likelihood

\[
\log P(D) = \sum_n \log P(y_n | x_n)
\]
Parameter estimation

**Discriminative approach:** maximize conditional likelihood

\[
\log P(D) = \sum_n \log P(y_n | x_n)
\]

We will change \( y_n \) to \( y_n = [y_{n1} \ y_{n2} \ \cdots \ y_{nK}]^T \), a \( K \)-dimensional vector using 1-of-\( K \) encoding.

\[
y_{nk} = \begin{cases} 
1 & \text{if } y_n = k \\
0 & \text{otherwise}
\end{cases}
\]

Ex: if \( y_n = 2 \), then, \( y_n = [0 \ 1 \ 0 \ 0 \ \cdots \ 0]^T \).
Parameter estimation

**Discriminative approach:** maximize conditional likelihood

\[
\log P(D) = \sum_n \log P(y_n | x_n)
\]

We will change \(y_n\) to \(y_n = [y_{n1} \ y_{n2} \ \cdots \ y_{nK}]^T\), a \(K\)-dimensional vector using 1-of-K encoding.

\[
y_{nk} = \begin{cases} 
1 & \text{if } y_n = k \\
0 & \text{otherwise} 
\end{cases}
\]

Ex: if \(y_n = 2\), then, \(y_n = [0 \ 1 \ 0 \ 0 \ \cdots \ 0]^T\).

\[
\Rightarrow \sum_n \log P(y_n | x_n) = \sum_n \log \prod_{k=1}^{K} P(C_k | x_n)^{y_{nk}} = \sum_n \sum_k y_{nk} \log P(C_k | x_n)
\]
Multiclass classification

Multinomial logistic regression

Cross-entropy error function

**Definition**: negated likelihood

\[ E(w_1, w_2, \ldots, w_K) = - \sum_n \sum_k y_{nk} \log P(C_k|x_n) \]
Cross-entropy error function

**Definition:** negated likelihood

\[ \mathcal{E}(\mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_K) = - \sum_{n} \sum_{k} y_{nk} \log P(C_k | \mathbf{x}_n) \]

**Properties**

- Convex, therefore unique global optimum
- Optimization requires numerical procedures, analogous to those used for binary logistic regression
- Large-scale implementation, in both the number of classes and the training examples, is non-trivial.
Outline

1. Administration

2. Multiclass classification

3. Generative versus discriminative
   - Contrast Naive Bayes and logistic regression
   - Another example: Gaussian discriminant analysis

4. Perceptron
Setup of the learning problem
Suppose the training data is from an unknown joint probabilistic model \( p(x, y) \)

Differences in assuming models for the data
- the generative approach requires we specify the model for the joint distribution (such as Naive Bayes), and thus, maximize the joint likelihood \( \sum_n \log p(x_n, y_n) \)
- the discriminative approach (discriminative) requires only specifying a model for the conditional distribution (such as logistic regression), and thus, maximize the conditional likelihood \( \sum_n \log p(y_n|x_n) \)
Naive Bayes and logistic regression: two different modeling paradigms

- Setup of the learning problem
  Suppose the training data is from an unknown joint probabilistic model $p(x, y)$
- Differences in assuming models for the data
  - the generative approach requires we specify the model for the joint distribution (such as Naive Bayes), and thus, maximize the joint likelihood $\sum_n \log p(x_n, y_n)$
  - the discriminative approach (discriminative) requires only specifying a model for the conditional distribution (such as logistic regression), and thus, maximize the conditional likelihood $\sum_n \log p(y_n | x_n)$
- Differences in computation
  - Sometimes, modeling by discriminative approach is easier
  - Sometimes, parameter estimation by generative approach is easier
Determining sex (man or woman) based on measurements
Generative approach

Propose a model of the joint distribution of \((x = \text{height}, \ y = \text{sex})\)

<table>
<thead>
<tr>
<th>Sex</th>
<th>Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>6’</td>
</tr>
<tr>
<td>2</td>
<td>5’2”</td>
</tr>
<tr>
<td>1</td>
<td>5’6”</td>
</tr>
<tr>
<td>1</td>
<td>6’2”</td>
</tr>
<tr>
<td>2</td>
<td>5.7”</td>
</tr>
</tbody>
</table>

Intuition: we will model how heights vary (according to a Gaussian) in each sub-population (male and female).

*Note:* This is similar to Naive Bayes for detecting spam emails, cf. Homework 1 too.
Model of the joint distribution

\[ p(x, y) = p(y)p(x|y) \]  
\[ = \begin{cases} 
  p_1 \frac{1}{\sqrt{2\pi\sigma_1}} e^{-\frac{(x-\mu_1)^2}{2\sigma_1^2}} & \text{if } y = 1 \\
  p_2 \frac{1}{\sqrt{2\pi\sigma_2}} e^{-\frac{(x-\mu_2)^2}{2\sigma_2^2}} & \text{if } y = 2 
\end{cases} \]

where \( p_1 + p_2 = 1 \) represents two prior probabilities that \( x \) is given the label 1 or 2 respectively. \( p(x|y) \) is called class distributions, which we have assumed to be Gaussians.
Parameter estimation

**Likelihood of the training data** $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^{N}$ with $y_n \in \{1, 2\}$

$$\log P(\mathcal{D}) = \sum_n \log p(x_n, y_n)$$

$$= \sum_{n:y_n=1} \log \left( p_1 \frac{1}{\sqrt{2\pi\sigma_1}} e^{-\frac{(x_n-\mu_1)^2}{2\sigma_1^2}} \right)$$

$$+ \sum_{n:y_n=2} \log \left( p_2 \frac{1}{\sqrt{2\pi\sigma_2}} e^{-\frac{(x_n-\mu_2)^2}{2\sigma_2^2}} \right)$$
Parameter estimation

Likelihood of the training data $\mathcal{D} = \{(x_n, y_n)\}_{n=1}^N$ with $y_n \in \{1, 2\}$

$$\log P(\mathcal{D}) = \sum_n \log p(x_n, y_n)$$

$$= \sum_{n: y_n = 1} \log \left( p_1 \frac{1}{\sqrt{2\pi\sigma_1}} e^{-\frac{(x_n - \mu_1)^2}{2\sigma_1^2}} \right)$$

$$+ \sum_{n: y_n = 2} \log \left( p_2 \frac{1}{\sqrt{2\pi\sigma_2}} e^{-\frac{(x_n - \mu_2)^2}{2\sigma_2^2}} \right)$$

Maximize the likelihood function

$$(p_1^*, p_2^*, \mu_1^*, \mu_2^*, \sigma_1^*, \sigma_2^*) = \arg \max \log P(\mathcal{D})$$
Decision boundary

As before, the Bayes optimal one under the assumed joint distribution depends on

\[ p(y = 1|x) \geq p(y = 2|x) \]

which is equivalent to

\[ p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2) \]
Generative versus discriminative

Another example: Gaussian discriminant analysis

Decision boundary

As before, the Bayes optimal one under the assumed joint distribution depends on

\[ p(y = 1|x) \geq p(y = 2|x) \]

which is equivalent to

\[ p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2) \]

Namely,

\[ -\frac{(x - \mu_1)^2}{2\sigma_1^2} - \log \sqrt{2\pi\sigma_1} + \log p_1 \geq -\frac{(x - \mu_2)^2}{2\sigma_2^2} - \log \sqrt{2\pi\sigma_2} + \log p_2 \]
Generative versus discriminative

Another example: Gaussian discriminant analysis

Decision boundary

As before, the Bayes optimal one under the assumed joint distribution depends on

\[ p(y = 1|x) \geq p(y = 2|x) \]

which is equivalent to

\[ p(x|y = 1)p(y = 1) \geq p(x|y = 2)p(y = 2) \]

Namely,

\[ -\frac{(x - \mu_1)^2}{2\sigma_1^2} - \log \sqrt{2\pi\sigma_1} + \log p_1 \geq -\frac{(x - \mu_2)^2}{2\sigma_2^2} - \log \sqrt{2\pi\sigma_2} + \log p_2 \]

\[ \Rightarrow ax^2 + bx + c \geq 0 \quad \Leftarrow \text{the decision boundary not linear!} \]
Example of nonlinear decision boundary

Note: the boundary is characterized by a quadratic function, giving rise to the shape of parabolic curve.
Generative versus discriminative

Another example: Gaussian discriminant analysis

A special case: what if we assume the two Gaussians have the same variance?

We will get a linear decision boundary

\[-\frac{(x - \mu_1)^2}{2\sigma_1^2} - \log \sqrt{2\pi\sigma_1} + \log p_1 \geq -\frac{(x - \mu_2)^2}{2\sigma_2^2} - \log \sqrt{2\pi\sigma_2} + \log p_2\]

with \(\sigma_1 = \sigma_2\), we have

\[bx + c \geq 0\]
A special case: what if we assume the two Gaussians have the same variance?

We will get a linear decision boundary

\[-\frac{(x - \mu_1)^2}{2\sigma_1^2} - \log \sqrt{2\pi\sigma_1} + \log p_1 \geq -\frac{(x - \mu_2)^2}{2\sigma_2^2} - \log \sqrt{2\pi\sigma_2} + \log p_2\]

with $\sigma_1 = \sigma_2$, we have

$$bx + c \geq 0$$

Note: equal variances across two different categories could be a very strong assumption.

For example, from the plot, it does seem that the male population has slightly bigger variance (i.e., bigger eclipse) than the female population. So the assumption might not be applicable.
Mini-summary

Gaussian discriminant analysis

- A generative approach, assuming the data modeled by

\[ p(x, y) = p(y)p(x|y) \]

where \( p(x|y) \) is a Gaussian distribution.

- Parameters (of those Gaussian distributions) are estimated by maximizing the likelihood
  
  - Computationally, estimating those parameters are very easy — it amounts to computing sample mean vectors and covariance matrices

- Decision boundary
  
  - In general, nonlinear functions of \( x \) — in this case, we call the approach *quadratic discriminant analysis*
  
  - In the special case we assume equal variance of the Gaussian distributions, we get a linear decision boundary — we call the approach *linear discriminant analysis*
So what is the discriminative counterpart?

**Intuition**
The decision boundary in Gaussian discriminant analysis is

\[ ax^2 + bx + c = 0 \]

Let us model the conditional distribution analogously

\[ p(y|x) = \sigma[ax^2 + bx + c] = \frac{1}{1 + e^{-(ax^2+bx+c)}} \]

Or, even simpler, going after the decision boundary of linear discriminant analysis

\[ p(y|x) = \sigma[bx + c] \]

Both look very similar to logistic regression — i.e. we focus on writing down the *conditional* probability, *not* the joint probability.
Does this change how we estimate the parameters?

**First change: a smaller number of parameters to estimate**

Our models are only parameterized by $a$, $b$ and $c$. There is no prior probabilities $(p_1, p_2)$ or Gaussian distribution parameters $(\mu_1, \mu_2, \sigma_1$ and $\sigma_2)$.

**Second change: we need to maximize the conditional likelihood $p(y|x)$**

$$(a^*, b^*, c^*) = \arg \min \sum_n \{ y_n \log \sigma(ax_n^2 + bx_n + c) \}$$

$$(1 - y_n) \log[1 - \sigma(ax_n^2 + bx_n + c)] \}$$

**Computationally, much harder!**
How easy for our Gaussian discriminant analysis?

Example

\[ p_1 = \frac{\text{# of training samples in class 1}}{\text{# of training samples}} \quad (8) \]

\[ \mu_1 = \frac{\sum_{n:y_n=1} x_n}{\text{# of training samples in class 1}} \quad (9) \]

\[ \sigma_1^2 = \frac{\sum_{n:y_n=1} (x_n - \mu_1)^2}{\text{# of training samples in class 1}} \quad (10) \]

Note: detailed derivation is in the books. They can be generalized rather easily to multi-variate distributions as well as multiple classes.
There is no fixed rule

- Selecting which type of method to use is dataset/task specific
- It depends on how well your modeling assumption fits the data
- Recent trend: big data is always useful for both!
  - Apply very complex discriminative models, such as deep learning methods, for building classifiers
  - Apply very complex generative models, such as nonparametric Bayesian methods, for modeling data
Outline

1. Administration
2. Multiclass classification
3. Generative versus discriminative
4. Perceptron
Classification

Given linear discriminant function

$$w^T x$$

is used to distinguish two classes $$\{-1, +1\}$$.

Our goal

$$\varepsilon = \sum_n \mathbb{I}[y_n \neq \text{sign}(w^T x_n)]$$

i.e., at least the errors on the training dataset are reduced.
Hard, but easy if we have only one training example

How to change \( \mathbf{w} \) such that

\[ y_n = \text{sign}(\mathbf{w}^T \mathbf{x}_n) \]

Two cases

- If \( y_n = \text{sign}(\mathbf{w}^T \mathbf{x}_n) \), do nothing.
- If \( y_n \neq \text{sign}(\mathbf{w}^T \mathbf{x}_n) \),

\[ \mathbf{w} \leftarrow \mathbf{w} + y_n \mathbf{x}_n \]
Why would it work?

Derivation on the blackboard
Iteratively solving one case at a time

- REPEAT
- Pick a data point $x_n$ (can be a fixed order of the training instances)
- Make a prediction $y = \text{sign}(w^T x_n)$
- If $y = y_n$, do nothing. Else,

$$w \leftarrow w + y_n x_n$$

- UNTIL converged.

Properties

- If the training data is linearly separable, the algorithm stops in a finite number of steps.
- The parameter vector is always a linear combination of training instances.