CSE 250A. Principles of AI

Probabilistic Reasoning and Decision-Making

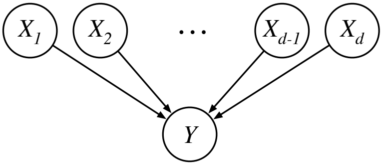
Lecture 10 – Learning from complete and incomplete data

Lawrence Saul
Department of Computer Science and Engineering
University of California, San Diego

Fall 2021

Outline

Linear regression



Suppose $Y \in \mathbb{R}$ is a real-valued random variable.
Then we can use a Gaussian conditional distribution:

$$P\left(y|\vec{x}\right) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left\{-\frac{(y - \vec{w} \cdot \vec{x})^2}{2\sigma^2}\right\}$$

How to learn the weight vector $\vec{w}$ from complete (IID) data?
This is the problem of **linear regression**.

ML estimation for linear regression

- **Log-conditional likelihood**

$$\mathcal{L}(\vec{w}) = -\frac{1}{2}\sum_{t=1}^{T}\left[\log(2\pi\sigma^2) + \frac{(y_t - \vec{w}\cdot\vec{x}_t)^2}{\sigma^2}\right]$$

- **Least-squares solution**

$$\left.\begin{aligned}\mathbf{A} &= \textstyle\sum_t \vec{x}_t\vec{x}_t^\top \\ \vec{b} &= \textstyle\sum_t y_t\vec{x}_t\end{aligned}\right\} \quad \vec{b} = \mathbf{A}\vec{w} \quad \Longrightarrow \quad \boxed{\vec{w}_{\mathrm{ML}} = \mathbf{A}^{-1}\vec{b}}$$

- **Failure modes**

Ill-conditioned problems arise when the inputs $\{\vec{x}_1, \vec{x}_2, \ldots, \vec{x}_T\}$ lie in (or very nearly in) a proper subspace of $\mathbb{R}^d$.

A detour on numerical optimization

How to maximize a multivariable function $f(\vec{\theta})$ over $\vec{\theta} \in \mathbb{R}^d$?

1. **Analytically — when it is possible**

   Compute the gradient and solve for where it vanishes:

$$\nabla f \;=\; \left(\frac{\partial f}{\partial \theta_1}, \frac{\partial f}{\partial \theta_2}, \ldots, \frac{\partial f}{\partial \theta_d},\right) \;=\; (0, 0, \ldots, 0)$$

2. **Numerically — via hillclimbing**

   Perform an iterative local search for a local or global maximum of $f(\vec{\theta})$.

Gradient ascent

- **Iterative update**

$$\boxed{\vec{\theta} \leftarrow \vec{\theta} + \eta \left(\frac{\partial f}{\partial \vec{\theta}} \right)}$$

- **Pros**

  Applies to any once-differentiable function.
  Converges asymptotically for sufficiently small η.

- **Cons**

  Sometimes tricky in practice to tune the learning rate.
  No guarantee of monotonic convergence.
  No guarantee of global optimality.

Newton's method

- **Iterative update**

$$\boxed{\vec{\theta} \leftarrow \vec{\theta} - \mathbf{H}^{-1}\left(\frac{\partial f}{\partial \vec{\theta}}\right)}$$

- **Pros**

  Applies to any twice-differentiable function.
  Converges rapidly (when it converges).
  Avoids the difficulty of tuning of a learning rate.

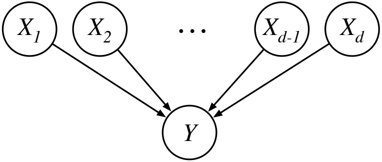
- **Cons**

  Expensive to compute Hessian matrix $O(d^2)$.
  Expensive to solve linear system $O(d^2)$.
  Unpredictable and/or unstable when initial estimate is poor.
  No guarantee of global optimality.

Outline

1. **Review**
2. **Logistic regression**
3. **Learning from incomplete data**
4. **Auxiliary functions**

Logistic regression



Suppose $Y \in \{0, 1\}$ is a binary random variable.
Then we can use a **sigmoid conditional distribution**:

$$P(Y\!=\!1|\vec{x}) \;=\; \sigma(\vec{w}\cdot\vec{x}) \;=\; \frac{1}{1+e^{-\vec{w}\cdot\vec{x}}}$$

How to learn the parameter $\vec{w} \in \mathbb{R}^d$ from complete data?
This is the problem of **logistic regression**.

Preliminaries

- **Properties of sigmoid function**

$$\sigma(z) = \frac{1}{1+e^{-z}} \qquad \boxed{\text{bounded between 0 and 1}}$$

$$\sigma(-z) = 1-\sigma(z) \qquad \boxed{\text{reflection symmetry}}$$

$$\tfrac{d}{dz}\sigma(z) = \sigma(z)\,\sigma(-z) \qquad \boxed{\text{derivative}}$$

- **IID data**

  As usual, we assume a complete data set of IID examples $\{(\vec{x}_t, y_t)\}_{t=1}^{T}$ where $\vec{x}_t \in \mathbb{R}^d$ and $y_t \in \{0,1\}$.

Log-conditional likelihood

$$\begin{aligned}
\mathcal{L}(\vec{w}) &= \log P(y_1, y_2, \ldots, y_T | \vec{x}_1, \vec{x}_2, \ldots, \vec{x}_T) \\
&= \log \prod_{t=1}^{T} P(y_t|\vec{x}_t) & \boxed{\text{data is IID}} \\
&= \sum_{t=1}^{T} \log P(y_t|\vec{x}_t) & \boxed{\log ab = \log a + \log b} \\
&= \sum_{t=1}^{T} \log \left[\sigma(\vec{w}\cdot\vec{x}_t)^{y_t} \, (1-\sigma(\vec{w}\cdot\vec{x}_t))^{1-y_t}\right] & \boxed{y_t \in \{0,1\}} \\
&= \sum_{t=1}^{T} \left[y_t \log \sigma(\vec{w}\cdot\vec{x}_t) \; + \; (1-y_t) \log(1-\sigma(\vec{w}\cdot\vec{x}_t))\right] & \boxed{\log a^b = b \log a} \\
&= \sum_{t=1}^{T} \left[y_t \log \sigma(\vec{w}\cdot\vec{x}_t) \; + \; (1-y_t) \log \sigma(-\vec{w}\cdot\vec{x}_t)\right] & \boxed{\sigma(-z) = 1 - \sigma(z)}
\end{aligned}$$

Computing the partial derivatives

$$\mathcal{L}(\vec{w}) = \sum_t \left[y_t \log \sigma(\vec{w}\cdot\vec{x}_t) + (1-y_t) \log \sigma(-\vec{w}\cdot\vec{x}_t) \right]$$

$$\begin{aligned}
\frac{\partial \mathcal{L}}{\partial w_\alpha} &= \sum_t \left[y_t \frac{1}{\sigma(\vec{w}\cdot\vec{x}_t)} \sigma(\vec{w}\cdot\vec{x}_t)\,\sigma(-\vec{w}\cdot\vec{x}_t)\, x_{\alpha t} \right. \\
&\qquad \left. + (1-y_t) \frac{1}{\sigma(-\vec{w}\cdot\vec{x}_t)} \sigma(-\vec{w}\cdot\vec{x}_t)\,\sigma(\vec{w}\cdot\vec{x}_t)\,(-x_{\alpha t}) \right] \\
&= \sum_t x_{\alpha t} \left[y_t\, \sigma(-\vec{w}\cdot\vec{x}_t) - (1-y_t)\, \sigma(\vec{w}\cdot\vec{x}_t) \right] \\
&= \sum_t x_{\alpha t} \left[y_t\, (1 - \sigma(\vec{w}\cdot\vec{x}_t)) - (1-y_t)\, \sigma(\vec{w}\cdot\vec{x}_t) \right] \\
&= \sum_t x_{\alpha t} \left[y_t - \sigma(\vec{w}\cdot\vec{x}_t) \right]
\end{aligned}$$

Interpreting the partial derivatives

- **Partial derivative**

$$\frac{\partial \mathcal{L}}{\partial w_\alpha} = \sum_{t=1}^{T} x_{\alpha t} \underbrace{\left[y_t \; - \; \sigma(\vec{w} \cdot \vec{x}_t) \right]}_{\textbf{error signal}}$$

- **Error signals**

For each example $(\vec{x}_t, y_t)$, the signal compares what the model should predict versus what it does:

$$\textbf{error signal} \quad = \quad \textbf{target label} \quad - \quad \textbf{model prediction}$$

$$y_t \qquad\qquad \sigma(\vec{w} \cdot \vec{x}_t)$$

$$P(Y=1|Y=y_t) \qquad\qquad P(Y=1|X=\vec{x}_t)$$

Maximizing the log-conditional likelihood

- **Where does the gradient vanish?**

$$\frac{\partial \mathcal{L}}{\partial w_\alpha} = 0 \quad \Longrightarrow \quad \sum_{t=1}^{T} x_{\alpha t}\, y_t = \sum_{t=1}^{T} x_{\alpha t}\, \sigma(\vec{w} \cdot \vec{x}_t)$$

- **The good news:**

  We have d equations, one for each $\alpha \in \{1, 2, \ldots, d\}$.
  And we have d unknowns $\vec{w} = (w_1, w_2, \ldots, w_d)$.

- **The bad news:**

  These equations are nonlinear — because $\sigma(z)$ is nonlinear.
  There is no way to solve them in closed form.

Maximizing the log-conditional likelihood

If we can't do it analytically, then we must do it numerically.
What are the simplest hillclimbing methods for this problem?

1. **Gradient ascent**

$$\boxed{\vec{w} \leftarrow \vec{w} + \eta \left(\frac{\partial \mathcal{L}}{\partial \vec{w}} \right)}$$

2. **Newton's method**

$$\boxed{\vec{w} \leftarrow \vec{w} - \mathbf{H}^{-1} \left(\frac{\partial \mathcal{L}}{\partial \vec{w}} \right)}$$

Partial derivatives

- **First partial derivatives**

$$\frac{\partial \mathcal{L}}{\partial w_\alpha} = \sum_{t=1}^{T} x_{\alpha t}\left[y_t - \sigma(\vec{w}\cdot\vec{x}_t)\right]$$

- **Second partial derivatives**

$$\begin{aligned}\frac{\partial^2 \mathcal{L}}{\partial w_\alpha \partial w_\beta} &= \frac{\partial}{\partial w_\beta}\left(\sum_{t=1}^{T} x_{\alpha t}\left[y_t - \sigma(\vec{w}\cdot\vec{x}_t)\right]\right)\\ &= -\sum_{t=1}^{T} x_{\alpha t}\left[\sigma(\vec{w}\cdot\vec{x}_t)\,\sigma(-\vec{w}\cdot\vec{x}_t)\,x_{\beta t}\right]\\ &= -\sum_{t=1}^{T} \sigma(\vec{w}\cdot\vec{x}_t)\,\sigma(-\vec{w}\cdot\vec{x}_t)\,x_{\alpha t}\,x_{\beta t}\end{aligned}$$

Gradient and Hessian in matrix-vector notation

- **Gradient**

$$\boxed{\nabla\mathcal{L} \quad = \quad \frac{\partial\mathcal{L}}{\partial\vec{w}} \quad = \quad \sum_{t=1}^{T}\Big[y_t - \sigma(\vec{w}\cdot\vec{x}_t)\Big]\vec{x}_t}$$

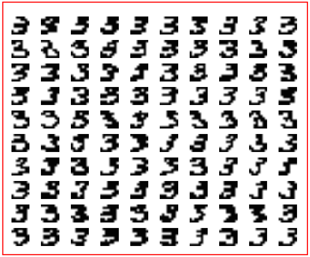
- **Hessian**

$$\boxed{\mathbf{H} \quad = \quad \frac{\partial^2\mathcal{L}}{\partial\vec{w}\,\partial\vec{w}^\top} \quad = \quad -\sum_{t=1}^{T}\sigma(\vec{w}\cdot\vec{x}_t)\,\sigma(-\vec{w}\cdot\vec{x}_t)\,\vec{x}_t\,\vec{x}_t^\top}$$

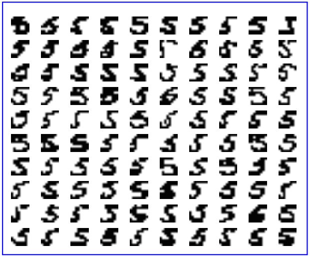
You will need to compute these for HW 5. Questions?

Logistic regression in HW 5

How well can this model distinguish images of 3s versus 5s?



$y = 0$



$y = 1$

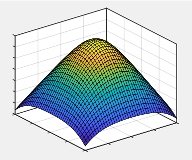
Recommendations:

- Set $\eta = \frac{0.2}{T}$ for gradient ascent.
- Initialize $\vec{w} = (0, 0, \ldots, 0)$ for Newton's method.

Global optimality

- **Theorem**

  The log-conditional likelihood $\mathcal{L}(\vec{w})$ for logistic regression is a **concave** *function of $\vec{w}$.*



- **Corollary**

  $\mathcal{L}(\vec{w})$ has no spurious local maxima.
  You should all converge to the same solution for HW 5!

- **Proof sketch**

  The Hessian is negative semidefinite: $\vec{v}^\top \mathbf{H} \vec{v} \leq 0$ for all $\vec{v} \in \mathbb{R}^d$.
  This is a sufficient condition for concavity.

Outline

1. **Review**
2. **Logistic regression**
3. **Learning from incomplete data**
4. **Auxiliary functions**

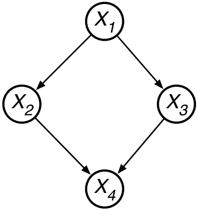
Learning from incomplete data with tabular CPTs

ASSUMPTIONS

1. The DAG is fixed (and known) over a finite set of discrete random variables $\{X_1, X_2, \ldots, X_n\}$.
2. CPTs enumerate $P(X_i = x | \text{pa}(X_i) = \pi)$ as lookup tables; each must be estimated for all values of x and π.
3. The data is IID, but only consists of T **partially** complete instantiations of the nodes in the BN.

Toy example

- **Fixed DAG over binary random variables**



$$\begin{aligned} X_1 &\in \{0,1\} \\ X_2 &\in \{0,1\} \\ X_3 &\in \{0,1\} \\ X_4 &\in \{0,1\} \end{aligned}$$

- **Incomplete data set**

example	X_1	X_2	X_3	X_4
1	1	**?**	0	1
2	0	1	**?**	0
3	**?**	**?**	**?**	1
⋮	⋮	⋮	⋮	⋮
T	**?**	1	1	0

How to choose the CPTs so that the BN maximizes the probability of this data set?

A more interesting example ...


How to build a movie recommendation system?

- Collect a data set of movie ratings:

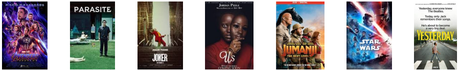
$$\begin{bmatrix} + & - & + & - & ? & ? & + \\ - & ? & ? & + & + & ? & ? \\ + & + & + & + & + & + & + \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ - & - & - & - & - & ? & - \\ ? & ? & + & ? & ? & ? & - \end{bmatrix}$$

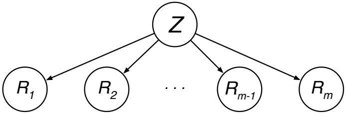
$+$	liked
$-$	disliked
$?$	not seen

(user-item matrix)

- Build a model of user profiles and fill in the missing ratings. But what model to build? (HW 8)

Naive Bayes model with incomplete data





- **Movie recommender system**

$$
\begin{array}{lcll}
Z & \in & \{1,2,\ldots,k\} & \text{type of movie-goer} \\
R_i & \in & \{0,1\} & \text{rating for } i^{\text{th}} \text{ movie}
\end{array}
$$

- **Incomplete data set**

student	Z	R_1	R_2	R_3	R_4	$\cdots$
1	**?**	0	1	1	**?**	$\cdots$
2	**?**	1	**?**	0	1	$\cdots$
3	**?**	0	0	**?**	1	$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
T	**?**	**?**	1	0	**?**	$\cdots$

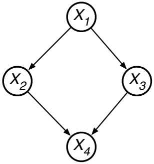
Note that the variable Z is **never observed**.

Learning from incomplete data

- **Notation**

$$H_t = \text{set of hidden (latent) variables for } t^{\text{th}} \text{ example}$$
$$V_t = \text{set of visible (observed) variables for } t^{\text{th}} \text{ example}$$

- **Illustration**



example	X_1	X_2	X_3	X_4
1	1	?	0	1
2	0	1	?	0
3	?	?	?	1
⋮	⋮	⋮	⋮	⋮

$$\begin{aligned} H_1 &= \{X_2\} & V_1 &= \{X_1, X_3, X_4\} \\ H_2 &= \{X_3\} & V_2 &= \{X_1, X_2, X_4\} \\ H_3 &= \{X_1, X_2, X_3\} & V_3 &= \{X_4\} \end{aligned}$$

Computing the log-likelihood with incomplete data

$$\mathcal{L} = \log P(\textbf{data})$$

$$= \log \prod_{t=1}^{T} P\left(V_t = v_t\right) \qquad \boxed{\textbf{data is IID}}$$

$$= \sum_{t=1}^{T} \log P\left(V_t = v_t\right) \qquad \boxed{\log ab = \log a + \log b}$$

$$= \sum_{t=1}^{T} \log \sum_{h} P\left(H_t = h, V_t = v_t\right) \qquad \boxed{\textbf{marginalization}}$$

$$= \sum_{t=1}^{T} \log \sum_{h} P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n)\Big|_{\{H_t = h, V_t = v_t\}} \qquad \boxed{\textbf{joint}}$$

$$= \sum_{t=1}^{T} \log \sum_{h} \prod_{i=1}^{n} P(X_i = x_i | \text{pa}_i = \pi_i)\Big|_{\{H_t = h, V_t = v_t\}} \qquad \boxed{\textbf{product rule}}$$

Complete versus incomplete data

- **Complete data**

$$\mathcal{L} \;=\; \sum_{i,\pi,x} \mathrm{count}(X_i\!=\!x, \mathrm{pa}_i\!=\!\pi) \log P(X_i\!=\!x|\mathrm{pa}_i\!=\!\pi)$$

The CPTs at different nodes are decoupled!
We can compute ML estimates in closed form.

- **Incomplete data**

$$\mathcal{L} \;=\; \sum_{t=1}^{T} \log \sum_{h} \prod_{i=1}^{n} P(X_i\!=\!x_i|\mathrm{pa}_i\!=\!\pi_i)\Big|_{\{H_t=h,V_t=v_t\}}$$

The CPTs are potentially all coupled.
How to proceed?

Outline

1. **Review**
2. **Logistic regression**
3. **Learning from incomplete data**
4. **Auxiliary functions**

How to maximize $f(\vec{\theta})$?

1. **Gradient ascent**

   $$\vec{\theta} \leftarrow \vec{\theta} + \eta \left(\frac{\partial f}{\partial \vec{\theta}} \right)$$

   × Tedious to tune η?
   × Not monotonically convergent.

2. **Newton's method**

   $$\vec{\theta} \leftarrow \vec{\theta} - \mathbf{H}^{-1} \left(\frac{\partial f}{\partial \vec{\theta}} \right)$$

   × Expensive for large problems.
   × Fast but unstable.

3. **Auxiliary function**

   $$\vec{\theta}_{\text{new}} = \operatorname*{argmax}_{\vec{\theta}} Q(\vec{\theta}, \vec{\theta}_{\text{old}})$$

   ✓ No learning rate.
   ✓ Monotonically convergent.

Auxiliary functions

- **Definition**

  A function $Q(\vec{\theta}', \vec{\theta})$ is called an *auxiliary function* for the *objective function* $f(\vec{\theta})$ if it satisfies two properties:

  (i) $Q(\vec{\theta}, \vec{\theta}) = f(\vec{\theta})$ for all $\vec{\theta}$ **equality**

  (ii) $Q(\vec{\theta}', \vec{\theta}) \leq f(\vec{\theta}')$ for all $\vec{\theta}, \vec{\theta}'$ **lower bound**

- **Theorem**

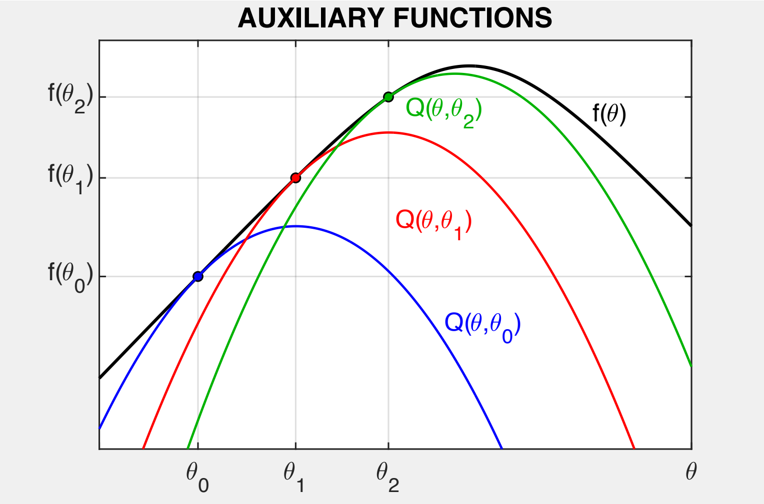
  Let $Q(\vec{\theta}', \vec{\theta})$ be an auxiliary function for the objective function $f(\vec{\theta})$. Then the update rule

$$\vec{\theta}_{\mathrm{new}} = \operatorname*{argmax}_{\vec{\theta}} Q(\vec{\theta}, \vec{\theta}_{\mathrm{old}})$$

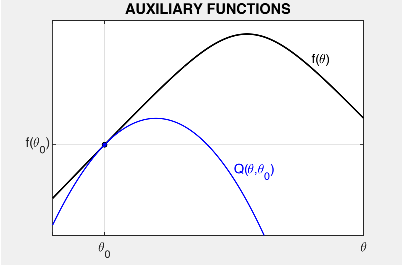
  converges monotonically to a stationary point with

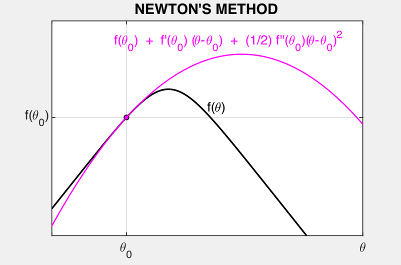
$$f(\vec{\theta}_{\mathrm{new}}) \geq f(\vec{\theta}_{\mathrm{old}}).$$

Visualization



Versus Newton's method





The quadratic approximation in Newton's method is **not** guaranteed to provide a lower bound on the objective function.

Proof of monotonic convergence

- **Proof**

Consider the update rule $\vec{\theta}_{\text{new}} = \text{argmax}_{\vec{\theta}}\, Q(\vec{\theta}, \vec{\theta}_{\text{old}})$.
Then we have

$$
\begin{aligned}
f(\vec{\theta}_{\text{new}}) &\geq Q(\vec{\theta}_{\text{new}}, \vec{\theta}_{\text{old}}) && \boxed{\text{property (ii)}} \\
&\geq Q(\vec{\theta}_{\text{old}}, \vec{\theta}_{\text{old}}) && \boxed{\text{argmax update}} \\
&= f(\vec{\theta}_{\text{old}}) && \boxed{\text{property (i)}}
\end{aligned}
$$

Iterating this process, we have:

$$f(\vec{\theta}_0) \leq f(\vec{\theta}_1) \leq f(\vec{\theta}_2) \leq \cdots \leq f(\vec{\theta}_n).$$

Next lecture: ML estimation for incomplete data!