Machine Learning
4771

Instructor: Tony Jebara
Topic 5

• Review: Back-Propagation
• Statistical Perspective
• Probability Models
• Discrete & Continuous: Gaussian, Bernoulli, Multinomial
• Maximum Likelihood Logistic Regression
• Conditioning, Marginalizing, Bayes Rule, Expectations
• Classification, Regression, Detection
• Dependence/Independence, Graphical Models
• Maximum Likelihood Naïve Bayes
Review: Back-Propagation

- Gradient descent can be done here, layer by layer
- Layers: input, hidden, output. Parameters: $\theta = \{w_{ij}, w_{jk}, w_{kl}\}$

$$
a_j = \sum_i w_{ij} z_i \\
z_j = g(a_j) \\
a_k = \sum_j w_{jk} z_j \\
z_k = g(a_k) \\
a_l = \sum_k w_{kl} z_k \\
z_l = g(a_l)
$$

- Each input $x_n$ for $n=1..N$ generates its own $a$’s and $z$’s
- Back-Propagation: Splits layer into its inputs & outputs
- Get gradient on output...back-track chain rule until input
Review: Back-Propagation

• Cost function: \( R(\theta) = \sum_n \frac{1}{2} (y^n - g(\sum_k w_{kl} g(\sum_j w_{jk} g(\sum_i w_{ij} x_i^n))))^2 \)

\[
\frac{\partial R}{\partial w_{kl}} = \sum_n \left[ \frac{\partial L_n^u}{\partial a_k^n} \right] \left[ \frac{\partial a_k^n}{\partial w_{kl}} \right] = \sum_n \left[ - \delta_k^n \right] \left[ g'(a_k^n) \right] \left[ z_k^n \right] = \sum_n \delta_k^n z_k^n
\]

\[
\frac{\partial R}{\partial w_{jk}} = \sum_n \left[ \frac{\partial L_n^u}{\partial a_k^n} \right] \left[ \frac{\partial a_k^n}{\partial w_{jk}} \right] = \sum_n \left[ \sum_l \delta_l^n w_{kl} g'(a_k^n) \right] \left[ z_j^n \right] = \sum_n \delta_k^n z_j^n
\]

• Any previous (input) layer derivative: repeat the formula!

\[
\frac{\partial R}{\partial w_{ij}} = \sum_n \left[ \frac{\partial L_n^u}{\partial a_j^n} \right] \left[ \frac{\partial a_j^n}{\partial w_{ij}} \right] = \sum_n \left[ \sum_k \frac{\partial L_n^u}{\partial a_k^n} \frac{\partial a_k^n}{\partial a_j^n} \right] \left[ \frac{\partial a_j^n}{\partial w_{ij}} \right] = \sum_n \left[ \sum_k \delta_k^n w_{jk} g'(a_j^n) \right] \left[ z_i^n \right] = \sum_n \delta_k^n z_i^n
\]

• What is this last \( z \)?
Back-Propagation

- Again, take small step in direction opposite to gradient

\[
\begin{align*}
    w_{ij}^{t+1} &= w_{ij}^t - \eta \frac{\partial R}{\partial w_{ij}} \\
    w_{jk}^{t+1} &= w_{jk}^t - \eta \frac{\partial R}{\partial w_{jk}} \\
    w_{kl}^{t+1} &= w_{kl}^t - \eta \frac{\partial R}{\partial w_{kl}}
\end{align*}
\]

- Digits Demo: LeNet... http://yann.lecun.com

- Problems with back-prop is that MLP over-fits...
- Other problems: hard to interpret, black-box
- What are the layers doing? Hiddens=?
- Statistical alternative to Neural Nets Bayesian Networks
  Nodes=random variables Links=cause/dependence
- So, next step: statistics...
Statistical Perspective

• Several problems with framework so far:
  Pulled non-linear squashing functions out of a hat
  Pulled loss functions (squared error, etc.) out of a hat
• Better approach for classification?
• What if we have multi-class classification?
• What if other problems, i.e. unobserved values of x,y,etc...
• Also, what if we don’t have a true function?

• Example of Projectile Cannon (c.f. Distal Learning)

• Would like to train a regression function to control
  a cannon’s angle of fire (y) given target distance (x)
Statistical Perspective

- Example of Projectile Cannon
  (45 degree problem)
  \( x = \text{input target distance} \)
  \( y = \text{output cannon angle} \)

\[ x = \frac{v(0)^2}{g} \sin(2y) + \text{noise} \]

- What does least squares do?
- Conditional statistical models address this problem...
Probability Models

• Instead of deterministic functions, output is a probability
• Previously: our output was a scalar \( \hat{y} = f(x) = \theta^T x + b \)
• Now: our output is a probability \( p(y) \)
  e.g. an probability bump:

• \( p(y) \) subsumes or is a superset of \( \hat{y} \)
• Why is this representation for our answer more general?
Probability Models

• Instead of deterministic functions, output is a probability
• Previously: our output was a scalar \( \hat{y} = f(x) = \theta^T x + b \)
• Now: our output is a probability \( p(y) \)

  e.g. a probability bump:

• \( p(y) \) subsumes or is a superset of \( \hat{y} \)
• Why is this representation for our answer more general? A deterministic answer \( \hat{y} \) with complete confidence is like putting a probability \( p(y) \) where all the mass is at \( \hat{y} \)!

\[
\hat{y} \Leftrightarrow p(y) = \delta(y - \hat{y})
\]
Probability Models

- Now: our output is a probability density function (pdf) \( p(y) \)
- Probability Model: a family of pdf’s with adjustable parameters which lets us select one of many
  \[ p(y) \to p(y | \theta) \]
- E.g.: 1-dim Gaussian distribution ‘given’ ‘mean’ parameter \( \mu \):
  \[ p(y | \mu) = N(y | \mu) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(y-\mu)^2} \]
- Want mean centered on \( f(x) \)’s value
  \[ p(y) = N(y | f(x)) \]
- Now, linear regression is:
  \[ N(y | f(x)) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(y-f(x))^2} = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(y-\theta^T x-b)^2} \]
Probability Models

- To fit to data, we typically “maximize likelihood” of the probability model

- Log-likelihood = objective function (i.e. negative of cost) for probability models which we want to maximize

- Likelihood = \( L(\theta) = \prod_{i=1}^{N} p(y_i | f(x_i)) \)

- Log-Likelihood = \( l(\theta) = \sum_{i=1}^{N} \log p(y_i | f(x_i)) \)

- For Gaussian, get squared error regression! Same solution!

\[
\sum_{i=1}^{N} \log p(y_i | f(x_i)) = \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}(y_i - f(x_i))^2}
\]

\[
= -N \log(\sqrt{2\pi}) - \sum_{i=1}^{N} \frac{1}{2} (y_i - f(x_i))^2
\]
Probability Models

- Can extend probability model to 2 bumps:
  \[ p(y) = \frac{1}{2} N(y | \mu_1) + \frac{1}{2} N(y | \mu_2) \]
- Each mean can be a linear regression fn.
  \[ p(y) = \frac{1}{2} N(y | f_1(x)) + \frac{1}{2} N(y | f_2(x)) \]
  \[ = \frac{1}{2} N(y | \theta_1^T x + b_1) + \frac{1}{2} N(y | \theta_2^T x + b_2) \]
- Log-Likelihood:
  \[ l(\theta_1, b_1, \theta_2, b_2) = \sum_{i=1}^{N} \log p(y_i) \]
  \[ l = \sum_{i=1}^{N} \log \left( \frac{1}{2} N(y_i | \theta_1^T x_i + b_1) + \frac{1}{2} N(y_i | \theta_2^T x_i + b_2) \right) \]
- Hard to solve \quad gradient descent
- Now can handle “cannon fire” example
- Later: a better probabilistic algorithm than gradient descent
Probability Models

- Now classification: can also go beyond deterministic!
- Previously: wanted output to be binary $\hat{y} = \{0,1\}$
- Now: our output is a probability $p(y)$
  
  e.g. a probability table:

<table>
<thead>
<tr>
<th></th>
<th>y=0</th>
<th>y=1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0.73</td>
<td>0.27</td>
</tr>
</tbody>
</table>

- This subsumes or is a superset again...
- Consider probability over binary events (coin flips!):
  
  e.g. Binomial Distribution (i.e. 1x2 probability table) with parameter $\alpha$
  
  $$p(y \mid \alpha) = \alpha^y (1-\alpha)^{1-y} \quad \alpha \in [0,1]$$

- Now linear classification is:
  
  $$p(y \mid f(x)) = f(x)^y (1-f(x))^{1-y} \quad f(x) \in [0,1]$$
Probability Models

- Now linear classification is:
  \[ p(y \mid f(x)) = f(x)^y (1 - f(x))^{1-y} \]
  \[ f(x) \equiv \alpha \in [0,1] \]

- Log-likelihood is (negative of cost function):
  \[
  \sum_{i=1}^{N} \log p(y_i \mid f(x_i)) = \sum_{i=1}^{N} \log f(x_i)^{y_i} (1 - f(x_i))^{1-y_i}
  \]
  \[
  = \sum_{i=1}^{N} y_i \log f(x_i) + (1 - y_i) \log (1 - f(x_i))
  \]
  \[
  = \sum_{i \in \text{class1}} \log f(x_i) + \sum_{i \in \text{class0}} \log (1 - f(x_i))
  \]

- But, need a squashing function since \(f(x)\) in \([0,1]\)
- Use sigmoid or logistic again...
  \[ f(x) = \text{sigmoid}(\theta^T \phi(x)) \equiv \text{sigmoid}(\theta^T x) \in [0,1] \]

- Called logistic regression \textit{new loss function}
- Do gradient descent, similar to logistic output neural net!
- Can also handle multi-layer \(f(x)\) and do backprop again!
Generative Probability Models

- Idea: Extend probability to describe both $X$ and $Y$
- Find probability density function over both: $p(x, y)$

E.g. *describe* data with Multi-Dim. Gaussian (later...)

- Called a ‘Generative Model’ because we can use it to synthesize or re-generate data similar to the training data we learned from

- Regression models & classification boundaries are not as flexible
don’t keep info about $X$
don’t model noise/uncertainty
Properties of PDFs

- Review: basics of probability theory (hang on!)
- First, pdf is a function, multiple inputs, one output:
  \[ p(x_1, \ldots, x_n) \quad p(x_1 = 0.3, \ldots, x_n = 1) = 0.2 \]
- Function’s output is always positive:
  \[ p(x_1, \ldots, x_n) \geq 0 \]
- Can have discrete or continuous or both inputs:
  \[ p(x_1 = 1, x_2 = 0, x_3 = 0, x_4 = 3.1415) \]
- Summing over the domain of all inputs gives unity:
  \[ \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} p(x, y) \, dx \, dy = 1 \quad \sum_{y} \sum_{x} p(x, y) = 1 \]

Continuous integral, Discrete sum
Properties of PDFs

• Marginalizing: integrate/sum out a variable leaves a marginal distribution over the remaining ones...

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

• Conditioning: if a variable ‘y’ is ‘given’ we get a conditional distribution over the remaining ones...

\[ p(x \mid y) = \frac{p(x, y)}{p(y)} \]

• Bayes Rule: mathematically just redo conditioning but has a deeper meaning (1764)... if we have x being data and \( \theta \) being a model

\[ p(\theta \mid x) = \frac{p(x \mid \theta) p(\theta)}{p(x)} \]
Properties of PDFs

- **Expectation:** can use pdf $p(x)$ to compute averages and expected values for quantities, denoted by:

$$E_{p(x)}\{f(x)\} = \int_x p(x) f(x) \, dx \quad \text{or} \quad E_{p(x)}\{f(x)\} = \sum_x p(x) f(x)$$

- **Properties:**

  - $E\{cf(x)\} = cE\{f(x)\}$
  - $E\{f(x) + c\} = E\{f(x)\} + c$
  - $E\{E\{f(x)\}\} = E\{f(x)\}$

- **Mean:** expected value for $x$

$$E_{p(x)}\{x\} = \int_{-\infty}^{\infty} p(x,y) \, x \, dx$$

- **Variance:** expected value of $(x-\text{mean})^2$, how much $x$ varies

$$\text{Var}\{x\} = E\{(x - E\{x\})^2\} = E\{x^2 - 2xE\{x\} + E\{x\}^2\}$$

$$= E\{x^2\} - 2E\{x\} \, E\{x\} + E\{x\}^2 = E\{x^2\} - E\{x\}^2$$

**Example: speeding ticket**

<table>
<thead>
<tr>
<th>Fine</th>
<th>Probability</th>
</tr>
</thead>
<tbody>
<tr>
<td>$0$</td>
<td>0.8</td>
</tr>
<tr>
<td>$20$</td>
<td>0.2</td>
</tr>
</tbody>
</table>

**Expected cost of speeding?**
Properties of PDFs

- **Covariance:** how strongly $x$ and $y$ vary together

$$\text{Cov} \{x, y\} = E \{(x - E \{x\})(y - E \{y\})\} = E \{xy\} - E \{x\} E \{y\}$$

- **Conditional Expectation:**

$$E \{y \mid x\} = \int_y p(y \mid x) \, y \, dx$$

$$E \left\{ E \{y \mid x\} \right\} = \int_x p(x) \int_y p(y \mid x) \, y \, dy \, dx = E \{y\}$$

- **Sample Expectation:** If we don’t have pdf $p(x, y)$ can approximate expectations using samples of data

$$E_{p(x)} \{f(x)\} \simeq \frac{1}{N} \sum_{i=1}^{N} f(x_i)$$

- **Sample Mean:**

$$E \{x\} \simeq \bar{x} = \frac{1}{N} \sum_{i=1}^{N} x_i$$

- **Sample Var:**

$$E \{(x - E(x))^2\} \simeq \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})^2$$

- **Sample Cov:**

$$E \{(x - E(x))(y - E(y))\} \simeq \frac{1}{N} \sum_{i=1}^{N} (x_i - \bar{x})(y_i - \bar{y})$$
More Properties of PDFs

• **Independence**: probabilities of independent variables multiply. Denote with the following notation:

\[
x \indep y \quad \rightarrow \quad p(x, y) = p(x)p(y)
\]

\[
x \indep y \quad \rightarrow \quad p(x \mid y) = p(x)
\]

Also note in this case:

\[
E_{p(x,y)} \{xy\} = \int_x \int_y p(x)p(y)xy \, dx \, dy
\]

\[
= \int_x p(x) \, x \, dx \int_y p(y) \, y \, dy = E_{p(x)} \{x\} \, E_{p(y)} \{y\}
\]

• **Conditional independence**: when two variables become independent only if another is observed

\[
x \indep y \mid z \quad \rightarrow \quad p(x \mid y, z) = p(x \mid z)
\]

\[
x \indep y \mid z \quad \rightarrow \quad p(x \mid y) \neq p(x)
\]
Uses of PDFs

- **Classification:** have $p(x,y)$ and given $x$. Asked for discrete $y$ output, give most probable one
  \[ p(x,y) \rightarrow p(y \mid x) \rightarrow \hat{y} = \arg\max_m p(y = m \mid x) \]

- **Regression:** have $p(x,y)$ and given $x$. Asked for a scalar $y$ output, give most probable or expected one
  \[ \hat{y} = \begin{cases} \arg\max_y p(y \mid x) \\ E_{p(y|y)} \{y\} \end{cases} \]

- **Anomaly Detection:** if have $p(x,y)$ and given both $x,y$. Asked if it is similar threshold
  \[ p(x,y) \geq \text{threshold} \rightarrow \{\text{normal, anomaly}\} \]
Discrete Probability Models

• **Binomial:** recall binary (coin flip) probability, just 1x2 table

\[ p(x) = \alpha^x (1 - \alpha)^{1-x} \quad \alpha \in [0,1] \quad x \in \{0,1\} \]

<table>
<thead>
<tr>
<th>x=0</th>
<th>x=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.73</td>
<td>0.27</td>
</tr>
</tbody>
</table>

• **Multidimensional Binomials:** multiple binary events

\[
\begin{array}{c|cc}
  & x_2=0 & x_2=1 \\
\hline
  x_1=0 & 0.4 & 0.1 \\
  x_1=1 & 0.3 & 0.2
\end{array}
\]

• **Multinomial:** beyond binary, multi-category event
  i.e. roll of the dice...

\[
p(x) = \prod_{m=1}^{M} \alpha_m^{\delta(x=m)} \quad \sum_m \alpha_m = 1 \quad x \in \{0,1,...,M\}
\]

<table>
<thead>
<tr>
<th>x=1</th>
<th>x=2</th>
<th>x=3</th>
<th>x=4</th>
<th>x=5</th>
<th>x=6</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.5</td>
</tr>
</tbody>
</table>

• Why do we write these as an equations instead of tables?
Structuring Probability Models

• Quickly explode if we have many variables...

• If they were independent: much more efficient

• But real events in the world not completely independent!
• Unrealistic...
• Can use graphical structure to identify more subtle dependencies and independencies

• Directed Graphical Model, also called Bayesian Network
• Neural Network = Graphical Function Representation
• Bayesian Network = Graphical Probability Representation