Administrative

• HW 2 released!

• Jia-Bin out of town next week

• Chen: Deep Neural Networks II
• Shih-Yang: Diagnosing ML systems
• Midterm review
• Midterm: March 6th (Wed)
**Hard-margin SVM formulation**

\[
\min_{\theta} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2
\]

s. t. \[\theta^T x^{(i)} \geq 1 \quad \text{if} \quad y^{(i)} = 1\]
\[\theta^T x^{(i)} \leq -1 \quad \text{if} \quad y^{(i)} = 0\]

**Soft-margin SVM formulation**

\[
\min_{\theta, \{\xi^{(i)}\}} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2 + C \sum_i \xi^{(i)}
\]

s. t. \[\theta^T x^{(i)} \geq 1 - \xi^{(i)} \quad \text{if} \quad y^{(i)} = 1\]
\[\theta^T x^{(i)} \leq -1 + \xi^{(i)} \quad \text{if} \quad y^{(i)} = 0\]
\[\xi^{(i)} \geq 0 \quad \forall \ i\]
Support Vector Machine

• Cost function

• Large margin classification

• Kernels

• Using an SVM
Non-linear classification

• How do we separate the two classes using a hyperplane?

\[ h_\theta(x) = \begin{cases} 
1 & \text{if } \theta^\top x \geq 0 \\
0 & \text{if } \theta^\top x < 0 
\end{cases} \]
Non-linear classification
Kernel

• $K(\cdot, \cdot)$ a legal definition of inner product:

$$\exists \phi: X \rightarrow \mathbb{R}^N$$

s.t. $K(x, z) = \phi(x)^T \phi(z)$
Why Kernels matter?

• Many algorithms interact with data only via dot-products

• Replace $x^T z$ with $K(x, z) = \phi(x)^T \phi(z)$

• Act \textit{implicitly} as if data was in the higher-dimensional $\phi$-space
Example

\[ K(x, z) = (x^\top z)^2 \] corresponds to

\[ (x_1, x_2) \to \phi(x) = \left( x_1^2, x_2^2, \sqrt{2}x_1x_2 \right) \]

\[ \phi(x)^\top \phi(z) = \left( x_1^2, x_2^2, \sqrt{2}x_1x_2 \right) \left( z_1^2, z_2^2, \sqrt{2}z_1z_2 \right)^\top \]

\[ = x_1^2z_1^2 + x_2^2z_2^2 + 2x_1x_2z_1z_2 = (x_1z_1 + x_2z_2)^2 \]

\[ = (x^\top z)^2 = K(x, z) \]
Example

\[ K(x, z) = (x^\top z)^2 \] corresponds to

\((x_1, x_2) \rightarrow \phi(x) = (x_1^2, x_2^2, \sqrt{2}x_1x_2) \)
Example kernels

- **Linear kernel**
  \[ K(x, z) = x^\top z \]

- **Gaussian (Radial basis function) kernel**
  \[ K(x, z) = \exp \left( -\frac{1}{2} (x - z)^\top \Sigma^{-1} (x - z) \right) \]

- **Sigmoid kernel**
  \[ K(x, z) = \tanh(a \cdot x^\top z + b) \]
Constructing new kernels

• Positive scaling
  \[ K(x, z) = cK_1(x, z), \quad c > 0 \]

• Exponentiation
  \[ K(x, z) = \exp(K_1(x, z)) \]

• Addition
  \[ K(x, z) = K_1(x, z) + K_2(x, z) \]

• Multiplication with function
  \[ K(x, z) = f(x)K_1(x, z)f(z) \]

• Multiplication
  \[ K(x, z) = K_1(x, z)K_2(x, z) \]
Non-linear decision boundary

Predict $y = 1$ if

$$
\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 \\
+ \theta_4 x_1^2 + \theta_5 x_2^2 + \cdots \geq 0
$$

$$
\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 + \cdots
$$

$$
f_1 = x_1, f_2 = x_2, f_3 = x_1 x_2, \cdots
$$

Is there a different/better choice of the features $f_1, f_2, f_3, \cdots$?
Kernel

Give $x$, compute new features depending on proximity to landmarks $l^{(1)}, l^{(2)}, l^{(3)}$

- $f_1 = \text{similarity}(x, l^{(1)})$
- $f_2 = \text{similarity}(x, l^{(2)})$
- $f_3 = \text{similarity}(x, l^{(3)})$

Gaussian kernel

$$\text{similarity}(x, l^{(i)}) = \exp\left(-\frac{\|x - l^{(i)}\|^2}{2\sigma^2}\right)$$

Slide credit: Andrew Ng
Predict $y = 1$ if

$$\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$$

Ex: $\theta_0 = -0.5, \theta_1 = 1, \theta_2 = 1, \theta_3 = 0$

$$f_1 = \text{similarity}(x, l^{(1)})$$
$$f_2 = \text{similarity}(x, l^{(2)})$$
$$f_3 = \text{similarity}(x, l^{(3)})$$

Slide credit: Andrew Ng
Choosing the landmarks

• Given $x$

$$f_i = \text{similarity}(x, l^{(i)}) = \exp\left(-\frac{||x - l^{(i)}||^2}{2\sigma^2}\right)$$

Predict $y = 1$ if $\theta_0 + \theta_1 f_1 + \theta_2 f_2 + \theta_3 f_3 \geq 0$

Where to get $l^{(1)}, l^{(2)}, l^{(3)}, \ldots$?

Slide credit: Andrew Ng
SVM with kernels

• Given \((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(m)}, y^{(m)})\)
• Choose \(l^{(1)} = x^{(1)}, l^{(2)} = x^{(2)}, l^{(3)} = x^{(3)}, \ldots, l^{(m)} = x^{(m)}\)
• Given example \(x\):
  • \(f_1 = \text{similarity}(x, l^{(1)})\)
  • \(f_2 = \text{similarity}(x, l^{(2)})\)
  • \(\ldots\)
• For training example \((x^{(i)}, y^{(i)})\):
  • \(x^{(i)} \rightarrow f^{(i)}\)
SVM with kernels

• Hypothesis: Given $x$, compute features $f \in \mathbb{R}^{m+1}$
  • Predict $y = 1$ if $\theta^T f \geq 0$

• Training (original)
\[
\min_{\theta} C \left[ \sum_{i=1}^{m} y^{(i)} \cos_1(\theta^T x^{(i)}) + (1 - y^{(i)}) \cos_0(\theta^T x^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2
\]

• Training (with kernel)
\[
\min_{\theta} C \left[ \sum_{i=1}^{m} y^{(i)} \cos_1(\theta^T f^{(i)}) + (1 - y^{(i)}) \cos_0(\theta^T f^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{n} \theta_j^2
\]
Support vector machines (Primal/Dual)

• Primal form

\[
\min_{\theta, \xi(i)} \frac{1}{2} \sum_{j=1}^{n} \theta_j^2 + C \sum_i \xi(i)
\]

s.t.
\[
\begin{align*}
\theta^T x(i) &\geq 1 - \xi(i) & \text{if} \ y(i) = 1 \\
\theta^T x(i) &\leq -1 + \xi(i) & \text{if} \ y(i) = 0 \\
\xi(i) &\geq 0 & \forall \ i
\end{align*}
\]

• Lagrangian dual form

\[
\min_{\alpha} \frac{1}{2} \sum_i \sum_j y(i)y(j) \alpha(i) \alpha(j) x(i)^T x(j) - \sum_i \alpha(i)
\]

s.t.
\[
\begin{align*}
0 &\leq \alpha(i) \leq C_i \\
\sum_i y(i) \alpha(i) &= 0
\end{align*}
\]
SVM (Lagrangian dual)

\[
\begin{align*}
\min_{\alpha} & \quad \frac{1}{2} \sum_i \sum_j y^{(i)} y^{(j)} \alpha^{(i)} \alpha^{(j)} x^{(i)\top} x^{(j)} - \sum_i \alpha^{(i)} \\
\text{s. t.} & \quad 0 \leq \alpha^{(i)} \leq C_i \\
& \quad \sum_i y^{(i)} \alpha^{(i)} = 0
\end{align*}
\]

Classifier: \( \theta = \sum_i \alpha^{(i)} y^{(i)} x^{(i)} \)

- The points \( x^{(i)} \) for which \( \alpha^{(i)} \neq 0 \) → Support Vectors

Replace \( x^{(i)\top} x^{(j)} \) with \( K(x^{(i)}, x^{(j)}) \)
SVM parameters

• $C \left( = \frac{1}{\lambda} \right)$
  - Large $C$: Lower bias, high variance.
  - Small $C$: Higher bias, low variance.

• $\sigma^2$
  - Large $\sigma^2$: features $f_i$ vary more smoothly.
    • Higher bias, lower variance
  - Small $\sigma^2$: features $f_i$ vary less smoothly.
    • Lower bias, higher variance

Slide credit: Andrew Ng
SVM Demo

• [https://cs.stanford.edu/people/karpathy/svmjs/demo/](https://cs.stanford.edu/people/karpathy/svmjs/demo/)
SVM song

• https://www.youtube.com/watch?v=g15bqtyidZs
Support Vector Machine

• Cost function

• Large margin classification

• Kernels

• Using an SVM
Using SVM

• SVM software package (e.g., liblinear, libsvm) to solve for $\theta$

• Need to specify:
  • Choice of parameter $C$.
  • Choice of kernel (similarity function):

• Linear kernel: Predict $y = 1$ if $\theta^\top x \geq 0$

• Gaussian kernel:
  • $f_i = \exp\left( -\frac{\|x-l^{(i)}\|^2}{2\sigma^2} \right)$, where $l^{(i)} = x^{(i)}$
  • Need to choose $\sigma^2$. Need proper feature scaling
Kernel (similarity) functions

• Note: not all similarity functions make valid kernels.

• Many off-the-shelf kernels available:
  • Polynomial kernel
  • String kernel
  • Chi-square kernel
  • Histogram intersection kernel
Multi-class classification

• Use one-vs.-all method. Train $K$ SVMs, one to distinguish $y = i$ from the rest, get $\theta^{(1)}, \theta^{(2)}, \ldots, \theta^{(K)}$

• Pick class $i$ with the largest $\theta^{(i)^T} x$

Slide credit: Andrew Ng
Logistic regression vs. SVMs

• $n =$ number of features ($x \in \mathbb{R}^{n+1}$), $m =$ number of training examples

1. **If $n$ is large (relative to $m$):** ($n = 10,000$, $m = 10 – 1000$)
   → Use logistic regression or SVM without a kernel ("linear kernel")

2. **If $n$ is small, $m$ is intermediate:** ($n = 1 – 1000$, $m = 10 – 10,000$)
   → Use SVM with Gaussian kernel

3. **If $n$ is small, $m$ is large:** ($n = 1 – 1000$, $m = 50,000+$)
   → Create/add more features, then use logistic regression of linear SVM

Neural network likely to work well for most of these case, but slower to train

Slide credit: Andrew Ng
Things to remember

• Cost function

\[
\min_{\theta} C \left[ \sum_{i=1}^{m} y^{(i)} \cos_1(\theta^T f^{(i)}) + (1 - y^{(i)}) \cos_0(\theta^T f^{(i)}) \right] + \frac{1}{2} \sum_{j=1}^{m} \theta_j^2
\]

• Large margin classification

• Kernels

• Using an SVM
Neural Networks

• Why neural networks?
• Model representation
• Examples and intuitions
• Multi-class classification
Neural Networks

• Why neural networks?

• Model representation

• Examples and intuitions

• Multi-class classification
Non-linear classification

Predict $y = 1$ if

$$g(\theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_1 x_2 + \theta_4 x_1^2 + \theta_5 x_2^2 + \ldots) \geq 0$$

$x_1 =$ size
$x_1 =$ #bedrooms
$x_1 =$ #floors
$x_1 =$ age
...
$x_{100}$

# Quadratic features?
# Cubic features?

Slide credit: Andrew Ng
What humans see

Slide credit: Larry Zitnick
### What computers see

<table>
<thead>
<tr>
<th>243</th>
<th>239</th>
<th>240</th>
<th>225</th>
<th>206</th>
<th>185</th>
<th>188</th>
<th>218</th>
<th>211</th>
<th>206</th>
<th>216</th>
<th>225</th>
</tr>
</thead>
<tbody>
<tr>
<td>242</td>
<td>239</td>
<td>218</td>
<td>110</td>
<td>67</td>
<td>36</td>
<td>34</td>
<td>152</td>
<td>213</td>
<td>208</td>
<td>208</td>
<td>221</td>
</tr>
<tr>
<td>243</td>
<td>242</td>
<td>123</td>
<td>58</td>
<td>94</td>
<td>82</td>
<td>132</td>
<td>77</td>
<td>108</td>
<td>208</td>
<td>208</td>
<td>215</td>
</tr>
<tr>
<td>235</td>
<td>217</td>
<td>115</td>
<td>212</td>
<td>243</td>
<td>236</td>
<td>247</td>
<td>139</td>
<td>91</td>
<td>209</td>
<td>208</td>
<td>213</td>
</tr>
<tr>
<td>233</td>
<td>208</td>
<td>131</td>
<td>222</td>
<td>219</td>
<td>226</td>
<td>196</td>
<td>114</td>
<td>74</td>
<td>208</td>
<td>213</td>
<td>214</td>
</tr>
<tr>
<td>232</td>
<td>217</td>
<td>131</td>
<td>116</td>
<td>77</td>
<td>150</td>
<td>69</td>
<td>56</td>
<td>52</td>
<td>201</td>
<td>228</td>
<td>223</td>
</tr>
<tr>
<td>232</td>
<td>232</td>
<td>182</td>
<td>186</td>
<td>184</td>
<td>179</td>
<td>159</td>
<td>123</td>
<td>93</td>
<td>232</td>
<td>235</td>
<td>235</td>
</tr>
<tr>
<td>232</td>
<td>236</td>
<td>201</td>
<td>154</td>
<td>216</td>
<td>133</td>
<td>129</td>
<td>81</td>
<td>175</td>
<td>252</td>
<td>241</td>
<td>240</td>
</tr>
<tr>
<td>235</td>
<td>238</td>
<td>230</td>
<td>128</td>
<td>172</td>
<td>138</td>
<td>65</td>
<td>63</td>
<td>234</td>
<td>249</td>
<td>241</td>
<td>245</td>
</tr>
<tr>
<td>237</td>
<td>236</td>
<td>247</td>
<td>143</td>
<td>59</td>
<td>78</td>
<td>94</td>
<td>255</td>
<td>248</td>
<td>247</td>
<td>251</td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>237</td>
<td>245</td>
<td>193</td>
<td>55</td>
<td>33</td>
<td>115</td>
<td>144</td>
<td>213</td>
<td>255</td>
<td>253</td>
<td>251</td>
</tr>
<tr>
<td>248</td>
<td>245</td>
<td>161</td>
<td>128</td>
<td>149</td>
<td>109</td>
<td>138</td>
<td>65</td>
<td>47</td>
<td>156</td>
<td>239</td>
<td>255</td>
</tr>
<tr>
<td>190</td>
<td>107</td>
<td>39</td>
<td>102</td>
<td>94</td>
<td>73</td>
<td>114</td>
<td>58</td>
<td>211</td>
<td>251</td>
<td>51</td>
<td>137</td>
</tr>
<tr>
<td>191</td>
<td>19</td>
<td>31</td>
<td>248</td>
<td>168</td>
<td>203</td>
<td>179</td>
<td>43</td>
<td>27</td>
<td>12</td>
<td>33</td>
<td>44</td>
</tr>
<tr>
<td>197</td>
<td>26</td>
<td>195</td>
<td>160</td>
<td>255</td>
<td>255</td>
<td>255</td>
<td>109</td>
<td>26</td>
<td>19</td>
<td>35</td>
<td>44</td>
</tr>
</tbody>
</table>
Computer Vision: Car detection

Testing:

Cars

Not a car

Slide credit: Andrew Ng
50x50 pixel images -> 2500 pixels

Quadratic features ($x_i x_j$)

3 million features

Slide credit: Andrew Ng
Neural Networks

• Origins: Algorithms that try to mimic the brain.

• Was very widely used in 80s and early 90s; popularity diminished in late 90s.

• Recent resurgence: State-of-the-art technique for many applications
An AI Timeline

Birth of AI
- Information Theory – digital signals
- Cybernetics – thinking machines
- The Turing Test
- Symbolic reasoning

Focus on Specific ‘Intelligence’
- Expert Systems (knowledge)
- Neural networks make a comeback
- Optical character recognition
- Speech recognition

Focus on Specific Problems
- Machine learning
- Deep learning – pattern analysis / classification
- Big data: large databases
- Fast processors to crunch data
- High-speed networks


- Limited computer processing power
- Limited database capacity
- Limited networking capabilities
- Real-world problems are complicated
  - Image processing / face recognition
  - Combinatorial explosion

- Disappointing results
- Collapse of dedicated hardware vendors

AI Winter AI Winter II

https://www.slideshare.net/dlavenda/ai-and-productivity
The “one learning algorithm” hypothesis

[Roe et al. 1992]
The “one learning algorithm” hypothesis

[Metin and Frost 1989]
Sensor representations in the brain

- Seeing with your tongue
- Human echolocation (sonar)
- Haptic belt: Direction sense
- Implanting a 3rd eye

[BrainPort; Welsh & Blasch, 1997; Nagel et al., 2005; Constantine-Paton & Law, 2009]
Neural Networks

• Why neural networks?

• Model representation

• Examples and intuitions

• Multi-class classification
A single neuron in the brain

Dendrite Input

Cell body

Axon Terminal

Node of Ranvier

Schwann cell

Myelin sheath

Nucleus

Output

Slide credit: Andrew Ng
An artificial neuron: Logistic unit

\[ x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ \theta_3 \end{bmatrix} \]

\[ h_\theta(x) = \frac{1}{1 + e^{-\theta^T x}} \]

- Sigmoid (logistic) activation function

Slide credit: Andrew Ng
Neural network

\[ a_i^{(j)} = \text{“activation” of unit } i \text{ in layer } j \]

\[ \Theta^{(j)} = \text{matrix of weights controlling function mapping from layer } j \text{ to layer } j + 1 \]

\[ s_j \text{ unit in layer } j \]

\[ s_{j+1} \text{ units in layer } j + 1 \]

\[ a_1^{(2)} = g \left( \Theta^{(1)}_{10} x_0 + \Theta^{(1)}_{11} x_1 + \Theta^{(1)}_{12} x_2 + \Theta^{(1)}_{13} x_3 \right) \]

\[ a_2^{(2)} = g \left( \Theta^{(1)}_{20} x_0 + \Theta^{(1)}_{21} x_1 + \Theta^{(1)}_{22} x_2 + \Theta^{(1)}_{23} x_3 \right) \]

\[ a_3^{(2)} = g \left( \Theta^{(1)}_{30} x_0 + \Theta^{(1)}_{31} x_1 + \Theta^{(1)}_{32} x_2 + \Theta^{(1)}_{33} x_3 \right) \]

\[ h_\Theta(x) = g \left( \Theta^{(2)}_{10} a_0^{(2)} + \Theta^{(1)}_{11} a_1^{(2)} + \Theta^{(1)}_{12} a_2^{(2)} + \Theta^{(1)}_{13} a_3^{(2)} \right) \]

Size of \( \Theta^{(j)} \)?

\[ s_{j+1} \times (s_j + 1) \]

Slide credit: Andrew Ng
Neural network

\[ a_1^{(2)} = g \left( \Theta_1^{(1)} x_0 + \Theta_2^{(1)} x_1 + \Theta_3^{(1)} x_2 + \Theta_4^{(1)} x_3 \right) = g(z_1^{(2)}) \]

\[ a_2^{(2)} = g \left( \Theta_5^{(1)} x_0 + \Theta_6^{(1)} x_1 + \Theta_7^{(1)} x_2 + \Theta_8^{(1)} x_3 \right) = g(z_2^{(2)}) \]

\[ a_3^{(2)} = g \left( \Theta_9^{(1)} x_0 + \Theta_{10}^{(1)} x_1 + \Theta_{11}^{(1)} x_2 + \Theta_{12}^{(1)} x_3 \right) = g(z_3^{(2)}) \]

\[ h_\Theta(x) = g \left( \Theta_{10}^{(2)} a_0^{(2)} + \Theta_{11}^{(2)} a_1^{(2)} + \Theta_{12}^{(2)} a_2^{(2)} + \Theta_{13}^{(2)} a_3^{(2)} \right) = g(z^{(3)}) \]

"Pre-activation"
Neural network

\[ a_1^{(2)} = g(z_1^{(2)}) \]
\[ a_2^{(2)} = g(z_2^{(2)}) \]
\[ a_3^{(2)} = g(z_3^{(2)}) \]

\[ h_\Theta(x) = g(z^{(3)}) \]

\[ x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \]
\[ z^{(2)} = \begin{bmatrix} z_1^{(2)} \\ z_2^{(2)} \\ z_3^{(2)} \end{bmatrix} \]

\[ z^{(2)} = \Theta^{(1)} x = \Theta^{(1)} a^{(1)} \]
\[ a^{(2)} = g(z^{(2)}) \]

Add \[ a_0^{(2)} = 1 \]
\[ z^{(3)} = \Theta^{(2)} a^{(2)} \]
\[ h_\Theta(x) = a^{(3)} = g(z^{(3)}) \]

"Pre-activation"
Neural network learning its own features

\[ h_\Theta(x) \]

Slide credit: Andrew Ng
Other network architectures

Slide credit: Andrew Ng
Neural Networks

• Why neural networks?

• Model representation

• Examples and intuitions

• Multi-class classification
Non-linear classification example: XOR/XNOR

• $x_1, x_2$ are binary (0 or 1)
• $y = XOR(x_1, x_2)$

Slide credit: Andrew Ng
Simple example: AND

- $x_1, x_2 \in \{0, 1\}$
- $y = x_1 \text{ AND } x_2$

$h_\Theta(x) = g(-30 + 20x_1 + 20x_2)$

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$h_\Theta(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$g(-30) \approx 0$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$g(-10) \approx 0$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$g(-10) \approx 0$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$g(10) \approx 1$</td>
</tr>
</tbody>
</table>

$h_\Theta(x) \approx x_1 \text{ AND } x_2$

Slide credit: Andrew Ng
Simple example: OR

- $x_1, x_2 \in \{0, 1\}$
- $y = x_1 \text{ AND } x_2$

$h_\Theta(x) = g(-10 + 20x_1 + 20x_2)$

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$h_\Theta(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>$g(-10) \approx 0$</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>$g(10) \approx 1$</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>$g(10) \approx 1$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$g(30) \approx 1$</td>
</tr>
</tbody>
</table>

$h_\Theta(x) \approx x_1 \text{ OR } x_2$

Slide credit: Andrew Ng
Simple example: NOT

• $x_1, x_2 \in \{0, 1\}$
• $y = x_1 \text{ AND } x_2$

\[
h_\Theta(x) = g(10 - 20x_1)
\]

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$h_\Theta(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>$g(10) \approx 1$</td>
</tr>
<tr>
<td>1</td>
<td>$g(-10) \approx 0$</td>
</tr>
</tbody>
</table>

$h_\Theta(x) \approx \text{NOT } x_1$

Slide credit: Andrew Ng
Putting it together: $x_1$ XNOR $x_2$

- $x_1$ AND $x_2$
- (NOT $x_1$) AND (NOT $x_2$)
- $x_1$ OR $x_2$

<table>
<thead>
<tr>
<th>$x_1$</th>
<th>$x_2$</th>
<th>$a_1^{(2)}$</th>
<th>$a_2^{(2)}$</th>
<th>$h_\theta(x)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Slide credit: Andrew Ng
Layer 1

Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]
Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]
Layer 3

Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]
Layer 4 and 5

Visualizing and Understanding Convolutional Networks [Zeiler and Fergus, ECCV 2014]
Neural Networks

• Why neural networks?
• Model representation
• Examples and intuitions

• Multi-class classification
Multiple output units: One-vs-all

Pedestrian? Car? Motorcycle? Truck?

\[ a_1^{(2)} \]
\[ a_2^{(2)} \]
\[ a_3^{(2)} \]
\[ a_1^{(3)} \]
\[ a_2^{(3)} \]
\[ a_3^{(3)} \]
Multiple output units: One-vs-all

\[ h_{\Theta}(x) \in \mathbb{R}^4 \]

Training set: \((x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots (x^{(m)}, y^{(m)}),\)

\[ y^{(i)} \text{ one of } \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \]
Things to remember

• Why neural networks?

• Model representation

• Examples and intuitions

• Multi-class classification