Stanford CS230 Notes

1 Introduction to Deep Learning

1.1 What is a Neural Network?

neuron, links.

1.2 Supervised Learning with Neural Networks

Supervised Learning

Examples: Standard NN, Convolutional NN, Recurrent NN

Structured Data: tabular data; Unstructured Data: audio/image/text

1.3 Why is Deep Learning taking off?

Amount of labeled data.

2 Basics of Neural Network Programming

2.1 Binary Classification

2.1.1 Binary Classification

image $\rightarrow$ 1 (cat) vs 0 (non cat)

2.1.2 Notation

m: number of examples

n_x: input size

n_y: output size

x: input, column vector

y: output, 0/1

X: input matrix, shape $= (n_x, m)$

Y: output matrix, shape $= (1, m)$

$x^{(i)}$: superscript (i) will denote the i^{th} example.

2.2 Logistic Regression

Given: $x \in \mathbb{R}^{n_x}$, $0 \leq \hat{y} \leq 1$

Parameters: $w \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}$

Output:

$$z = w^T x + b \tag{1}$$

$$\hat{y} = \sigma(z) \tag{2}$$

$$\sigma(z) \approx \frac{1}{1 + e^{-z}} \tag{3}$$

$$z \approx \infty, \sigma(z) \approx \frac{1}{1 + 0} = 1 \tag{4}$$

$$z \approx -\infty, \sigma(z) \approx \frac{1}{1 + \infty} = 0 \tag{5}$$

Simplified Parameters: $x_0 = 1$, $x \in \mathbb{R}^{n_x + 1}$

$$\theta_0 = b, \theta_1 ... \theta_{n_x} = w \tag{6}$$

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \theta_2 \\ ... \\ \theta_{n_x} \end{bmatrix} \tag{7}$$

$$\hat{y} = \sigma(\theta^T x) \tag{8}$$

2.3 Logistic Regression cost function

Given $\{(x^{(1)}, y^{(1)}, ..., (x^{(m)}, y^{(m)}\}$, want $\hat{y}^{(i)} \approx y^{(i)}$.

Loss (error) function (mean square error/cross entropy):

$$\mathcal{L}(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2 \tag{9}$$

$$\mathcal{L}(\hat{y}, y) = -(y \log \hat{y} + (1 - y) \log(1 - \hat{y})) \tag{10}$$

If y = 1: $\mathcal{L}(\hat{y}, y) = -\log \hat{y}$, want $\mathcal{L}$ small, want $\hat{y}$ large, want $\hat{y}$ equal to 1. If y = 0: $\mathcal{L}(\hat{y}, y) = -\log(1 - \hat{y})$, want $\mathcal{L}$ small, want $\hat{y}$ small, want $\hat{y}$ equal to 0.

Cost function:

$$J(w, b) = \frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(\hat{y}, y) \tag{11}$$

$$J(w, b) = -\frac{1}{m} \sum_{i=1}^{m} [y^{(i)} \log \hat{y}^{(i)} + (1 - y^{(i)}) \log 1 - \hat{y}^{(i)}] \tag{12}$$

Given a random variable X with probability mass function $p_X(x)$, the self information of measuring X as outcome x is defined as:

$$I_X(x) = \log[p_X(x)] = \log(\frac{1}{p_X(x)}) \tag{13}$$

Shannon Entroy of X:

$$H(X) = \sum_x -p_X(x) \log p_X(x) \tag{14}$$
$$= \sum_x p_X(x) I_X(x) \tag{15}$$
$$= E[I_X(x)] \tag{16}$$

Cross Entropy of the the true distributions p and estimated distribution q:

$$H(p,q) = E_p[-\log q] = -\sum_{x\in\mathcal{X}} p(x) \log q(x) \tag{17}$$

2.4 Gradient Descent

Want to find w, b that minimize $J(w,b)$.

Repeat:

$$w := w - \alpha \frac{\mathrm{d}J(w,b)}{\mathrm{d}w} \tag{18}$$
$$b := b - \alpha \frac{\mathrm{d}J(w,b)}{\mathrm{d}b} \tag{19}$$

α: learning rate

2.5 Derivatives

$$f(a) = 3a, \frac{\mathrm{d}f(a)}{\mathrm{d}a} = 3 = \frac{\mathrm{d}}{\mathrm{d}a} f(a) \tag{20}$$
$$f(a) = a^2, \frac{\mathrm{d}}{\mathrm{d}a} f(a) = 2a \tag{21}$$
$$f(a) = a^3, \frac{\mathrm{d}}{\mathrm{d}a} f(a) = 3a^2 \tag{22}$$
$$f(a) = \log_e a = \ln a, \frac{\mathrm{d}}{\mathrm{d}a} f(a) = \frac{1}{a} \tag{23}$$
$$f(x) = \log_a x, \frac{\mathrm{d}}{\mathrm{d}x} f(x) = \frac{1}{x \ln a} \tag{24}$$
$$f(x) = a^x, \frac{\mathrm{d}}{\mathrm{d}x} f(x) = a^x \ln a \tag{25}$$
$$\log_a b = \frac{\log_c b}{\log_c a} = \frac{\ln a}{\ln b} \tag{26}$$

2.6 Computation Graph

$$
\begin{aligned}
J(a,b,c) &= 3(a+bc) && (27)\\
&= 3(a+u) && (28)\\
&= 3v && (29)\\
u &= bc && (30)\\
v &= a+u && (31)\\
J &= 3v && (32)
\end{aligned}
$$

2.7 Derivatives with a Computation Graph

a = 5, b = 3, c = 2

$$
\begin{aligned}
\frac{\mathrm{d}J}{\mathrm{d}v} &= 3 && (33)\\
\frac{\mathrm{d}J}{\mathrm{d}a} &= \frac{\mathrm{d}J}{\mathrm{d}v}\frac{\mathrm{d}v}{\mathrm{d}a} && (34)\\
&= 3 * 1 && (35)\\
&= 3 && (36)\\
\frac{\mathrm{d}J}{\mathrm{d}u} &= \frac{\mathrm{d}J}{\mathrm{d}v}\frac{\mathrm{d}v}{\mathrm{d}u} && (37)\\
&= 3 && (38)\\
\frac{\mathrm{d}J}{\mathrm{d}b} &= \frac{\mathrm{d}J}{\mathrm{d}u}\frac{\mathrm{d}u}{\mathrm{d}b} && (39)\\
&= 3 * c && (40)\\
&= 3 * 2 && (41)\\
&= 6 && (42)\\
\frac{\mathrm{d}J}{\mathrm{d}c} &= \frac{\mathrm{d}J}{\mathrm{d}u}\frac{\mathrm{d}u}{\mathrm{d}c} && (43)\\
&= 3 * b && (44)\\
&= 3 * 3 && (45)\\
&= 9 && (46)
\end{aligned}
$$

2.8 Logistic Regression Gradient Descent

2.8.1 Logicstic regression recap

$$z = w^T x + b \tag{47}$$

$$\hat{y} = a = \sigma(z) \tag{48}$$

$$\mathcal{L}(a, y) = -(y \log(a) + (1 - y) \log(1 - a)) \tag{49}$$

2.8.2 Logistic regression derivatives

$z = w_1 x_1 + w_2 x_2 + b \rightarrow a = \sigma(z) \rightarrow \mathcal{L}(a, y)$

$$\mathrm{d}a = \frac{\mathrm{d}\mathcal{L}(a, y)}{\mathrm{d}a} \tag{50}$$

$$= -\frac{y}{a} + \frac{1 - y}{1 - a} \tag{51}$$

$$\mathrm{d}z = \frac{\mathrm{d}\mathcal{L}(a, y)}{\mathrm{d}z} \tag{52}$$

$$= \frac{\mathrm{d}\mathcal{L}}{\mathrm{d}a} * \frac{\mathrm{d}a}{\mathrm{d}z} \tag{53}$$

$$= (-\frac{y}{a} + \frac{1 - y}{1 - a}) * a * (1 - a) \tag{54}$$

$$= -(1 - a)y + a(1 - y) \tag{55}$$

$$= a - y \tag{56}$$

$$\frac{\mathrm{d}a}{\mathrm{d}z} = a * (1 - a) \tag{57}$$

$$= \frac{1}{1 + e^{-z}} * (1 - \frac{1}{1 + e^{-z}}) \tag{58}$$

$$\mathrm{d}w_1 = x_1 * \mathrm{d}z \tag{59}$$

$$\mathrm{d}w_2 = x_2 * \mathrm{d}z \tag{60}$$

$$\mathrm{d}b = \mathrm{d}z \tag{61}$$

Repeat:

$$w_1 := w_1 - \alpha \mathrm{d}w_1 \tag{62}$$

$$w_2 := w_2 - \alpha \mathrm{d}w_2 \tag{63}$$

$$b := b - \alpha \mathrm{d}b \tag{64}$$

2.9 Gradient Descent on m examples

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(a^{(i)}, y(i)) \tag{65}$$

$$\frac{\mathrm{d}}{\mathrm{d}w_1}J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\frac{\mathrm{d}}{\mathrm{d}w_1}\mathcal{L}(a^{(i)}, y(i)) \tag{66}$$

$$= \frac{1}{m}\sum_{i=1}^{m}\mathrm{d}w_1^{(i)} \tag{67}$$

2.10 Vectorization

Vectorized, for GPU.

```
z = np.dot(w, x) + b
```

2.11 More Vectorization Examples

2.11.1 Neural network programming guideline

Whenever possible, avoid explicit for-loops.

```
u = np.dot(A, x)
```

2.11.2 Vectors and matrix valued functions

```
import numpy as np

u = np.exp(v)
np.log(v)
np.abs(v)
np.maximum(v, 0)
np.pow(v, 2)
np.divide(1, v)
```

2.12 Vectorizing Logistic Regression

2.13 Vectorizing Logistic Regression's Gradient Computation

2.14 Broadcasting in Python

```
np.add((1.0, 2.0), 2.0)    # (3.0, 4.0)
```

2.15 Explanation of logistic regression cost function (Optional)

2.15.1 Logistic regression cost function

$\hat{y} = \sigma(w^T x + b)$ where $\sigma(z) = \frac{1}{1+e^{-z}}$

$$\hat{y} = p(y = 1|x) \tag{68}$$

If $y = 1$: $p(y|x) = \hat{y}$
If $y = 0$: $p(y|x) = 1 - \hat{y}$

$$p(y|x) = \hat{y}^y (1 - \hat{y})^{(1-y)} \tag{69}$$
$$y = 1 : p(y|x) = \hat{y} \tag{70}$$
$$y = 0 : p(y|x) = 1 - \hat{y} \tag{71}$$
$$\log p(y|x) = y \log \hat{y} + (1 - y) \log 1 - \hat{y} \tag{72}$$
$$= -\mathcal{L}(\hat{y}, y) \tag{73}$$

2.15.2 Cost on m examples

Maximum likehood estimation:

$$\log p(...) = \sum_{i=1}^{m} -\mathcal{L}(\hat{y}, y) \tag{74}$$
$$= -\sum_{i=1}^{m} \mathcal{L}(\hat{y}, y) \tag{75}$$

Minimize cost:

$$J(w,b) = \frac{1}{m}\sum_{i=1}^{m}\mathcal{L}(\hat{y}, y) \tag{76}$$