① Adversarial Examples

E.g. Logistic Regression Classifier on 1D Dataset ($x \in \mathbb{R}$)

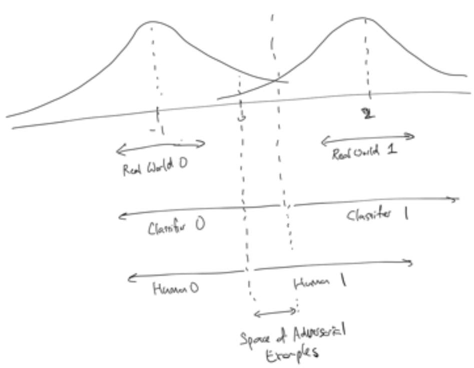
BCE Loss $= -\frac{1}{M}\sum_{i=1}^{M} y^{(i)}\log \hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)})$

where $\hat{y} = \sigma(\omega x + b)$ where $\omega \in \mathbb{R}$, $b \in \mathbb{R}$.

Assume:

- $x \mid y=0 \sim N(-1,1)$
- $x \mid y=1 \sim N(2,1)$
- To humans, if $x<0$, looks like $y=0$
  if $x \geq 0$, looks like $y=1$

(This is a reasonable assumption. Data in the real world might lie in say $(-1.5, -0.5)$ for class 0 and $(1,3)$ for class 1 most of the time and so $\mathrm{sgn}(x)$ might be a good heuristic for humans!)



(A) Train Logistic Regression Classifier

After training, to find boundary, we need $P(y=0|x) = P(y=1|x) = 0.5$

$$0.5 = \sigma(\omega x + b)$$

$$0.5 = \frac{1}{1+e^{-\omega x - b}}$$

$$2 = 1 + e^{-\omega x - b}$$

$$e^{-\omega x - b} = 1$$

$$-\omega x - b = 0$$

$$x = -\frac{b}{\omega}$$

(B) Naive Adversarial Example

Let's start with an example where $y=1$.

e.g. $x=2$.

Now, we train with the same BCE loss but with label $y=0$.

Use BCE loss to backprop to x.

$$x = x - \alpha \frac{\partial L}{\partial x}$$

End Result (see Jupyter): Gets classified as $y=0$ ✓
but no longer "looks like" $y=1$ ✗

(C) Better Adversarial Example

Start with same example ($x=2$)

Train with label $y=0$

Loss = BCE loss + $\lambda(x_{new} - 2)^2$

End Result (see Jupyter): Gets classified as $y=0$ ✓
and still "looks like" $y=1$ ✓

(D) Black Box vs. White Box Attacks

White Box: Access to architecture and weights

↳ Like example above

↳ Can compute $\frac{\partial L}{\partial x}$.

to architecture or weights

Black Box: No access ...

↳ Strategy: Generate a dataset $D = [(x_i, y_i)]_{i=1}^{m}$ where each datapoint $(x^{(i)}, y^{(i)})$ is taken by sampling x from some distribution and getting the corresponding y from the model M that you're trying to attack.

↳ Train new classifier $\hat{M}$ on dataset D. (like part (A))

↳ Do white box attack on $\hat{M}$ (like part (C))

↳ Hope the resulting x is also an adversarial example for M.

(E) Targeted vs. Non-targeted Attacks

Targeted: Want M to misclassify to specific class j

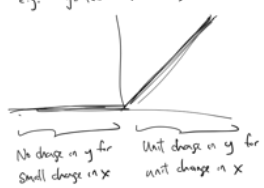
Non-targeted: Want M to misclassify to any class other than correct class

---

(F) Fast Gradient Sign Method

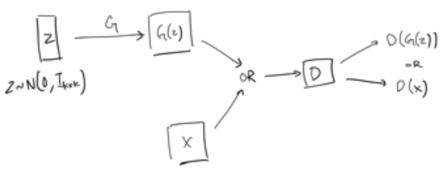
- (c) may be slow! ⇒ Many gradient steps
- Idea: Just do $x = x - \varepsilon \, \mathrm{sign}\left(\frac{\partial L}{\partial x}\right)$ where L is original BCE loss.
  - ↳ Minimize perceptual difference
    - ↳ e.g. if $\left(\frac{\partial L}{\partial x}\right)_j$ is big in magnitude, without taking the sign, x may no longer look like original x.
- Especially effective since most non-linear activations are usually in the linear regime (for fast training).

e.g. $y = \mathrm{ReLU}(wx+b)$



---

② GANs



Notes:

- y is 1 if input is x.
- y is 0 if input is $G(z)$
- $D(G(z))$ and $D(x)$ are probabilities with range $(0,1)$.

D is a classifier. ⇒ Use BCE loss!

$$J^{(D)} = -\frac{1}{m}\sum_{i=1}^{m} y^{(i)}\log \hat{y}^{(i)} + (1-y^{(i)})\log(1-\hat{y}^{(i)})$$

$$= -\frac{1}{M_{real}}\sum_{i=1}^{M_{real}} \log(D(x^{(i)})) - \frac{1}{M_{gen}}\sum_{i=1}^{M_{gen}} \log(1-D(G(z^{(i)})))$$

G is trying to fool D.

$$\text{So } J^{(G)} = -J^{(D)}$$

$$= \frac{1}{M_{real}}\sum_{i=1}^{M_{real}} \log(D(x^{(i)})) + \frac{1}{M_{gen}}\sum_{i=1}^{M_{gen}} \log(1-D(G(z^{(i)})))$$

0 (∵ no G in this term. Constant!)

But this formulation of $J^{(G)}$ is a <u>saturating loss</u>. This is because initially, discriminator is doing very well (∵ much easier to train classifier as compared to generator).

∴ Initially, $D(G(z)) \approx 0.5$. Very quickly, $D(G(z)) \approx 0$ because discriminator trains fast.

$$\frac{\partial J^{(G)}}{\partial h} = \frac{1}{M_{gen}}\sum_{i=1}^{M_{gen}} \frac{-D(1-D)}{1-D} \approx 0$$

∂D ... 1 (same thing) ... ∂G_{out}

Since D is a discriminator between 2 classes, we can write D as

From Chain Rule, $\frac{\partial J^{(G)}}{\partial G_\theta} = \underbrace{\frac{\partial J}{\partial D_{out}}}_{0} \frac{\cdots}{\partial D_{in}} \frac{\cdots}{\cancel{\partial h_{out}}} \frac{\cdots}{\partial G_\theta}$

$\approx 0.$

$\sigma(h)$ where h is last layer before applying sigmoid.

Instead of minimizing the likelihood of discriminator being correct, we Maximize the likelihood of discriminator being wrong.

$$\therefore \min J^{(G)} = \min \left[\frac{1}{M_{gen}} \sum_{i=1}^{M_{gen}} \log\left(1 - D\left(G\left(z^{(i)}\right)\right)\right) \right]$$

$$= \max \left[\frac{1}{M_{gen}} \sum_{i=1}^{M_{gen}} \log\left(D\left(G\left(z^{(i)}\right)\right)\right) \right] = \max J'^{(G)}$$

Once again, initially $D(G(z)) \approx 0$. Then,

$$\frac{\partial J'^{(G)}}{\partial h} = \frac{1}{M_{gen}} \sum_{i=1}^{M_{gen}} \frac{\cancel{D}(1-D)}{\cancel{D}} \approx 1$$

We call $J'^{(G)}$ a non-saturating loss for the generator.