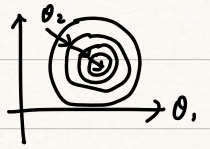
1. What is **gradient descent**? Why is it important?

GD: an iterative optimization algorithm (step-by-step)

→ find the params corresponding to optimal points of a target function (e.g. min loss, max likelihood, ...) step-by-step.

→ Start with initial param values, incrementally modify these values in the way that leads to largest improvement. (take derivatives!)



Important: for optimization problem with no closed from solution.

2. What is **Logistic Regression**? What is "logistic"? What are we "regressing"?

Goal: train classifier that can make binary decision about the class of an input obs.

LR
- $y=1$: positive
- $y=0$: negative

⇒ Given test instance $x = [x_1, \ldots, x_f]$ ⇒ 1/0.

⇒ Model: calculate prob $P(y=1|x)$

Decision boundary as 0.5:

$$\text{classify as } \begin{cases} 1 & \text{if } P(y=1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

We apply logistic function (sigmoid) σ to regression z.

$$\sigma = \frac{1}{1+e^{-z}}$$

$$z = \theta_0 + \theta_1 x_1 + \cdots + \theta_f x_f \quad (\theta_i : \text{params})$$

- Easy to calculate derivative ⇒ for gradient descent
- Has range $[0, 1]$ ⇒ estimate probability.

Regressing: log odds: $\log \frac{p}{1-p} = z$

3. [OPTIONAL] What is the relation between "odds" and "probability"?

prob : $p = p(\text{success})$

odds : ratio of $p(\text{success})$ to $p(\text{failure})$: $\frac{p}{1-p}$

E.g. 8 balls : 5 red $\Rightarrow$ $p(\text{red}) = \frac{5}{8}$

odds of drawing red ball :

$$\text{odds} = \frac{\frac{5}{8}}{1-\frac{5}{8}} = \frac{\frac{5}{8}}{\frac{3}{8}} = \frac{5}{3} = 1.7$$

4. In following dataset each instance represents a news article. The value of the features are counts of selected words in each article. Develop a logistic regression classifier to predict the class of the article (fruit vs. computer). $\hat{y} = 1$ (fruit) and $\hat{y} = 0$ (computer).

ID	*apple*	*ibm*	*lemon*	*sun*	CLASS
TRAINING INSTANCES					
A	1	0	1	5	1 FRUIT
B	1	0	1	2	1 FRUIT
C	2	0	0	1	1 FRUIT
D	2	2	0	0	0 COMPUTER
E	1	2	1	7	0 COMPUTER
TEST INSTANCES					
T	1	2	1	5	?

For the moment, we assume that we already have an estimate of the model parameters, i.e., the weights of the 4 features (and the bias θ_0) is $\widehat{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4] = [0.2, 0.3, -2.2, 3.3, -0.2]$.

(i). Explain the intuition behind the model parameters, and their meaning in relation to the features

Feature engineering :

- Choose terms (as attributes)
- Define word occurence counts as attribute values.

LR :

$$P(y=1 \mid \underset{\sim}{x}) = \frac{1}{1+e^{-z}} = \sigma(z), \quad z = \theta_0 + \theta_1 x_1 + \cdots + \theta_4 x_4$$

Params :

$\theta_1, \theta_2, \theta_3, \theta_4$ → importance of 4 features (terms) for predicting class 1 (fruit).

θ_0 → bias (intercept)

(ii). Predict the test label.

$\hat{z} = \hat{\theta}_0 + \hat{\theta}_1 x_1 + \cdots + \hat{\theta}_4 x_4$

$= 0.2 + 0.3 - 2.2 \times 2 + 3.3 - 0.2 \times 5$

$= -1.6$

$\sigma(-1.6) = \frac{1}{1+e^{1.6}} = 0.17$ (for fruit)

$0.17 < 0.5 \Rightarrow$ Classify as computer

(iii). Recall the conditional likelihood objective

$$\log \mathcal{L}(\theta) = -\sum_{i=1}^{n} y_i \log\big(\sigma(x_i;\theta)\big) + (1-y_i)\log\big(1-\sigma(x_i;\theta)\big)$$

We want to make sure that the Loss (the negative log likelihood) our model, is lower when its prediction the correct label for test instance T, than when it's predicting a wrong label.

HINT: Compute the negative log-likelihood of the test instance (1) assuming the true label as y = 1 (fruit), i.e., our classifier made a mistake; and (2) assuming that the true label y = 0 (computer), i.e., our classifier predicted correctly.

Predict $\hat{y} = 0$ (computer): (from (ii))

(1) If $y = 1$:

$\log \mathcal{L}(\theta) = -\{1 \cdot \log(\sigma(x_i;\theta)) + 0 \cdot \log(1-\sigma(x_i;\theta))\}$

$= -\log(\sigma(x_i;\theta))$

$= -\log(0.17)$

$= 1.77$

(2) If $y = 0$:

$\log \mathcal{L}(\theta) = -\{0 \cdot \log(\sigma(x_i;\theta)) + 1 \cdot \log(1-\sigma(x_i;\theta))\}$

$= -\log(1-\sigma(x_i;\theta))$

$= -\log(1-0.17)$

$= 0.19$ (lower loss)

5.

A. For the model created in question 4, compute a single gradient descent update for parameter θ_1 given the training instances given above. Recall that for each feature j, we compute its weight update as

$$\theta_j \leftarrow \theta_j - \eta \sum_i (\sigma(x_i;\theta) - y_i)x_{ij}$$

$-\frac{\partial}{\partial \theta} \log L(\theta)$

Summing over all training instances i. We will compute the update for θ_j assuming the current parameters as specified above, and a learning rate $\eta = 0.1$.

$\hat{\theta} = [0.2, 0.3, -2.2, 3.3, -0.2]$

① Compute $\sigma(x_i;\theta)$ for all i (train instances) (pred)

x_{1A} to x_{1E}

$$\sigma(x_A;\theta) = \sigma(0.2 + (0.3 \times 1 + (-2.2) \times 0 + 3.3 \times 1 + (-0.2) \times 5)) = 0.94$$

$$\sigma(x_B;\theta) = \sigma(0.2 + (0.3 \times 1 + (-2.2) \times 0 + 3.3 \times 1 + (-0.2) \times 2)) = 0.97$$

$$\sigma(x_C;\theta) = \sigma(0.2 + (0.3 \times 2 + (-2.2) \times 0 + 3.3 \times 0 + (-0.2) \times 1)) = 0.65$$

$$\sigma(x_D;\theta) = \sigma(0.2 + (0.3 \times 2 + (-2.2) \times 2 + 3.3 \times 0 + (-0.2) \times 0)) = 0.03$$

$$\sigma(x_E;\theta) = \sigma(0.2 + (0.3 \times 1 + (-2.2) \times 2 + 3.3 \times 1 + (-0.2) \times 7)) = 0.12$$

② Update params (e.g. θ_1)

$$\theta_1 = \theta_1 - \eta \sum_{i\in\{A,B,C,D,E\}} (\sigma(x_i;\theta) - y_i)\, x_{1i}$$

$$\theta_1 = 0.3 - 0.1 \sum_{i\in\{A,B,C,D,E\}} (\sigma(x_i;\theta) - y_i)\, x_{1i}$$

η (learning rate)

$$\theta_1 = 0.3 - 0.1\,[((\sigma(x_A;\theta) - y_A).x_{1A}) + ((\sigma(x_B;\theta) - y_B).x_{1B}) + ((\sigma(x_C;\theta) - y_C).x_{1C})$$

$$+((\sigma(x_D;\theta) - y_D).x_{1D}) + ((\sigma(x_E;\theta) - y_E).x_{1E})] \quad \Sigma$$

$$= 0.3 - 0.1\,[((0.94 - 1) \times 1) + ((0.97 - 1) \times 1) + ((0.65 - 1) \times 2) + ((0.03 - 0) \times 2)$$

$$+((0.12 - 0) \times 1)]$$

$$= 0.3 - 0.1(-0.06) + (-0.03) + (-0.70) + 0.06 + 0.12) = 0.3 - 0.1\,(-0.61)$$

$$= 0.3 + 0.061 = 3.061 \quad \text{(new } \theta_1\text{)}$$

⇒ Do same thing for other θ's.

✱ Note: update all params at once.