Multinomial Logistic Regression

CSC 461: Machine Learning

Fall 2022

Prof. Marco Alvarez
University of Rhode Island

MNIST

- The MNIST database is a large database of handwritten digits
  - ✓ contains 60,000 training images and 10,000 testing images
  - ✓ convolutional neural networks, manages to get an error rate of 0.23%
  - ✓ original paper reports an error rate of 0.8% with SVMs

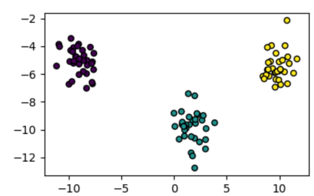
http://yann.lecun.com/exdb/mnist/

https://en.wikipedia.org/wiki/MNIST_database

Logistic regression

- Binary classification
  - ✓ uses a **logistic function** (type of sigmoid function)
  - ✓ models **probability** of output in terms of input
- What if we want k > 2 classes?
  - ✓ can try **one-vs-all**
    - ✓ learn a binary classifier per class; relabel training data with samples of that class as positive and all other as negatives; predict using the highest score from all classifiers
  - ✓ can try **one-vs-one**
    - ✓ learn k (k-1) / 2 binary classifiers; each learns to distinguish between two classes; predict using a voting scheme

OvA and OvO



Issues with OvA or OvO

- Class imbalance
- Scale of scores may differ from classifier to classifier
- Computational cost (both train and predict)

Basics of multiclass classification

- Data instance
  - ✓ in general, $\mathbf{x} \in \mathbb{R}^d$
  - ✓ $y \in \{1,2,\ldots,C\}$
- Hypothesis
  - ✓ $h : \mathcal{X} \mapsto \mathcal{Y}, h \in \mathcal{H}$

From binary to k classes

- Binary logistic regression:

$$P(y=+1 \mid \mathbf{x};\mathbf{w}) = \frac{1}{1+e^{-\mathbf{w}^T\mathbf{x}}} = \frac{e^{\mathbf{w}^T\mathbf{x}}}{e^{\mathbf{w}^T\mathbf{x}}+1}$$

- Can be extended to:

softmax function

$$P(y=c \mid \mathbf{x};\mathbf{W}) = \boxed{\frac{e^{\mathbf{w}_c^T\mathbf{x}}}{\sum_{k=1}^{C} e^{\mathbf{w}_k^T\mathbf{x}}}}$$

$\mathbf{w}_c$ is the row vector c from $\mathbf{W}$

$W_{C\times d+1}$ is a matrix where every row is a "class" weight vector

Softmax function

$$\sigma(\mathbf{z}) = \frac{e^{z_i}}{\sum_{k=1}^{K} e^{z_k}}$$

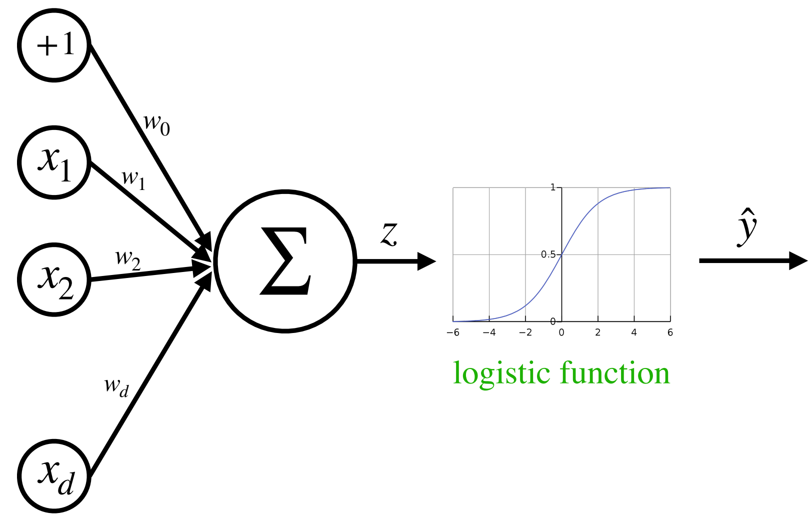
for $i = 1,\ldots,K$ and $\mathbf{z} \in \mathbb{R}^K$

converts a vector of K real numbers into a **probability distribution** of K possible outcomes

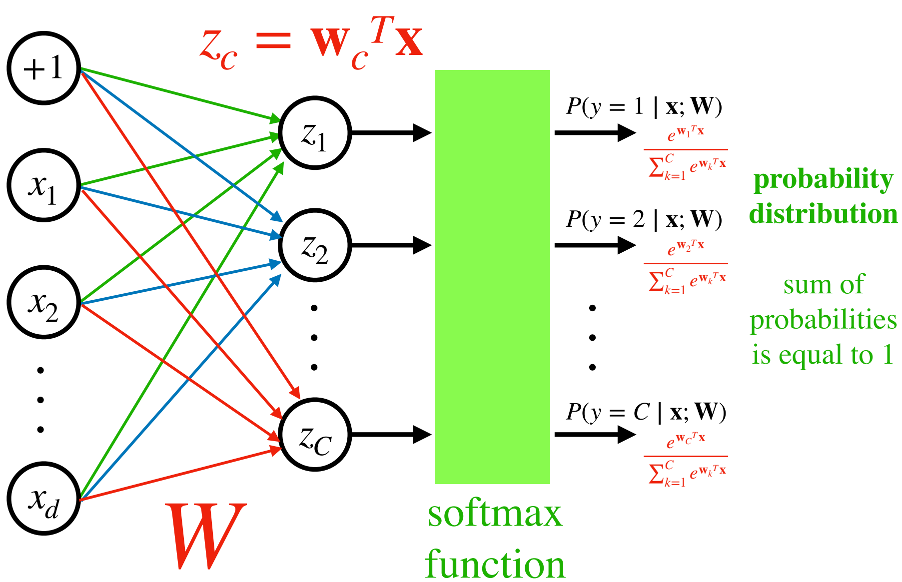
Example

- What is the value of softmax($\mathbf{z}$), given that $\mathbf{z}^T = [-10,10,5,4.3,7]$?

From … logistic regression



To … multinomial logistic regression



Multinomial logistic regression

- Use the **softmax function** for activation
- Predict the label with the highest probability score

$$\hat{y} = \arg\max_c P(y = c \mid \mathbf{x}; \mathbf{W})$$

- How to learn the weights?
  - ✓ need to define a **Loss Function** … then apply **gradient descent**
  - ✓ loss function can be derived using **MLE** (similar to binary logistic regression)

Maximum likelihood estimation

$$\mathbf{W}^* = \arg\max_{\mathbf{W}} \frac{1}{n}\prod_{i=1}^{n} P(y^{(i)} \mid \mathbf{x}^{(i)}; \mathbf{W})$$

$$= \arg\max_{\mathbf{W}} \frac{1}{n}\prod_{i=1}^{n}\prod_{c=1}^{C} P\left(y^{(i)} = c \mid \mathbf{x}^{(i)}; \mathbf{W}\right)^{t_{i,c}}$$

$$= \arg\max_{\mathbf{W}} \frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C} t_{i,c}\log\left(P(y^{(i)} = c \mid \mathbf{x}^{(i)}; \mathbf{W})\right)$$

$$= \arg\min_{\mathbf{W}} -\frac{1}{n}\sum_{i=1}^{n}\sum_{c=1}^{C} t_{i,c}\log\left(P(y^{(i)} = c \mid \mathbf{x}^{(i)}; \mathbf{W})\right)$$

$$= \arg\min_{\mathbf{W}} -\frac{1}{n}\sum_{i=1}^{n}\underbrace{\sum_{c=1}^{C} t_{i,c}\log\left(\frac{e^{\mathbf{w}_c\mathbf{x}^{(i)}}}{\sum_{k=1}^{C} e^{\mathbf{w}_k\mathbf{x}^{(i)}}}\right)}_{\text{error measure } e(h_{\mathbf{w}}(\mathbf{x}), y)}$$

Consider a matrix $T_{n\times C}$ where every row is a **one-hot encoding** of the target variable

Cross-entropy loss

Softmax and the loss function

Want to interpret raw classifier scores as **probabilities**

$s = f(x_i; W)$ $\quad P(Y = k | X = x_i) = \frac{e^{s_k}}{\sum_j e^{s_j}}$ Softmax Function

$L_i = -\log P(Y = y_i | X = x_i)$

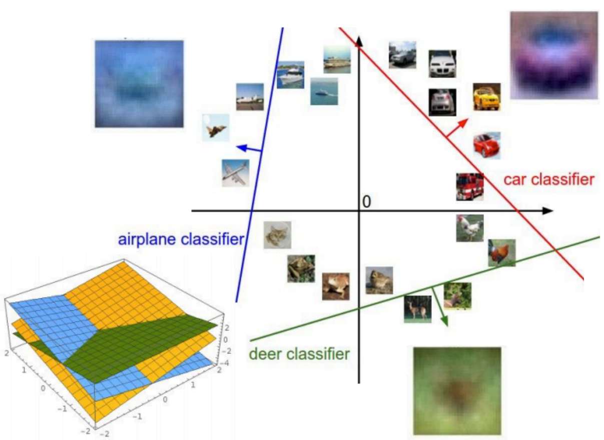
	Unnormalized log-probabilities / logits	$\xrightarrow{\text{exp}}$ unnormalized probabilities (Probabilities must be >= 0)	$\xrightarrow{\text{normalize}}$ probabilities (Probabilities must sum to 1)
cat	**3.2**	**24.5**	**0.13**
car	5.1	164.0	0.87
frog	-1.7	0.18	0.00

→ L_i = -log(0.13) = **2.04**

$$L_i = -\log\left(\frac{e^{s_{y_i}}}{\sum_j e^{s_j}}\right)$$

http://vision.stanford.edu/teaching/cs231n/slides/2019/cs231n_2019_lecture03.pdf

Geometric interpretation

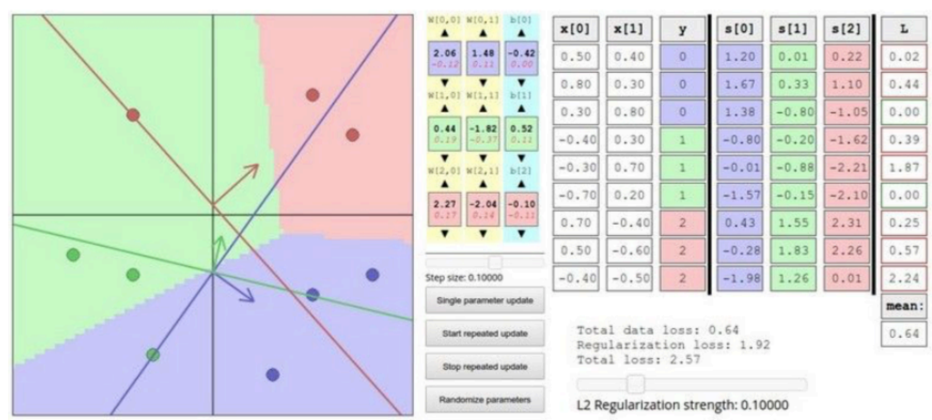


$$Wx$$

Array of **32x32x3** numbers

http://vision.stanford.edu/teaching/cs231n/slides/2019/cs231n_2019_lecture02.pdf

Interactive web demo



http://vision.stanford.edu/teaching/cs231n-demos/linear-classify/