# Multiclass classification

D.A. Forsyth,

University of Illinois at Urbana Champaign

# Simple, two-class linear classifier

- Assume each data item is (feature vector, label)
  - fv is x

**Remember this:** *A two-class linear classifier maps a feature vector* $\mathbf{x}$ *to one of two classes, using*

$$sign(\mathbf{a}^T\mathbf{x} + b)$$

*where* $\mathbf{a}$ *and* $b$ *are parameters of the classifier which are chosen to get strong performance on training data. Linear classifiers are much more powerful than you might expect at first glance. The decision boundary of a linear classifier is a hyperplane in feature space.*

# Generalizing this idea

Imagine you wish to classify something into one of $k$ classes. You have a set of examples where the classes are known for each example. You can encode this information by associating a *one hot vector* with each training example. This vector has $k$ components, one per class. The component corresponding to the class is one, and all others are zero. Write $\mathbf{y}_i$ for that vector for the $i$'th training example.

Your classifier must now produce a $k$ dimensional vector for an example. This is quite commonly called a *score* Write $\mathbf{x}$ for the example and $\mathbf{s}$ for the vector of scores produced by the classifier. Then a linear classifier predicts

$$\mathbf{s} = \mathcal{A}\mathbf{x} + \mathbf{b}.$$

Stack the entries of $\mathcal{A}$ and $\mathbf{b}$ into a vector $\theta$ of all the parameters of the classifier. Notice there is a small disparity between a two class classifier as described in Section 21.2 and this description of a multiclass classifier. This minor nuisance is explored in the exercises **exercises** .

# Interpreting the score

Interpret the score as a probability using

$$P(\text{item is class } u|\mathbf{x}, \theta) = \frac{\exp[s_u]}{\sum_w \exp[s_u]}.$$

The loss could be the negative log-likelihood of the data $\mathbf{y}$ under the model. Write $\mathbf{q}$ for the vector whose $u$'th component is

$$q_u = \frac{\exp[s_u]}{\sum_w \exp[s_w]}.$$

Then the negative log-likelihood for the example is

$$C(\theta; \mathbf{x}, \mathbf{y}) = -\log\left[\mathbf{y}^T \mathbf{q}\right]$$

and averaging over the whole dataset yields

$$\mathcal{L}_{lr} = \frac{1}{N} \sum_{i \in \text{examples}} C(\theta; \mathbf{x}_i, \mathbf{y}_i).$$

# Compare to two classes

- You could have a_1, b_1 and a_2, b_2
- You get:

$$p(\text{class is } 1|\mathbf{x}) = \frac{e^{\mathbf{a}_1^T\mathbf{x}+b_1}}{e^{\mathbf{a}_1^T\mathbf{x}+b_1} + e^{\mathbf{a}_2^T\mathbf{x}+b_2}}$$

$$= \frac{e^{(\mathbf{a}_1-\mathbf{a}_2)^T\mathbf{x}+(b_1-b_2)}}{e^{(\mathbf{a}_1-\mathbf{a}_2)^T\mathbf{x}+(b_1-b_2)} + 1}$$

- This all happens because probabilities sum to 1

# Think about this...

**22.1.** Why does a two-class classifier use only one score, but a $k$-class classifier uses $k$ scores for $k > 2$?

**22.2.** Section 22.1.3 has: "The advantage of the cross-entropy derivation is that it tells you what to do when your training data is uncertain." Explain.