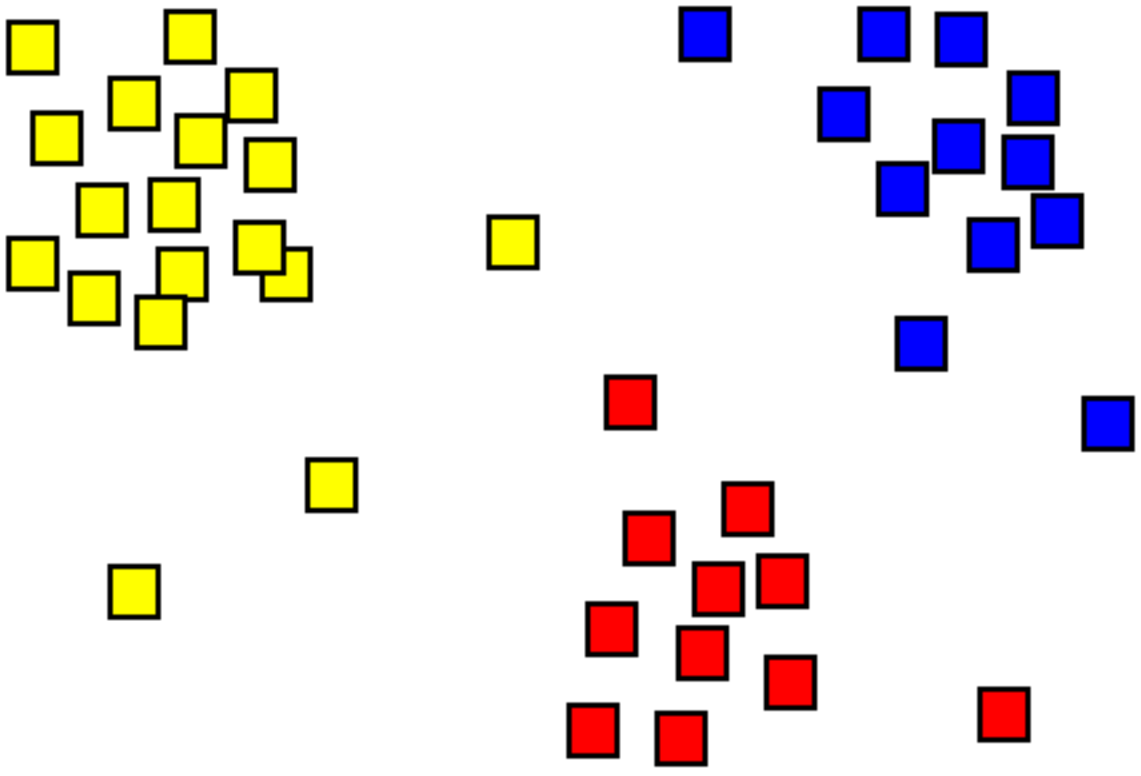
Multiclass classification



Reminders/Comments

- Assignment 2 updates
  - Clarification on step-size selection
  - Clarification on reported l2err
- Appreciate feedback
  - Quick comments about mini quiz

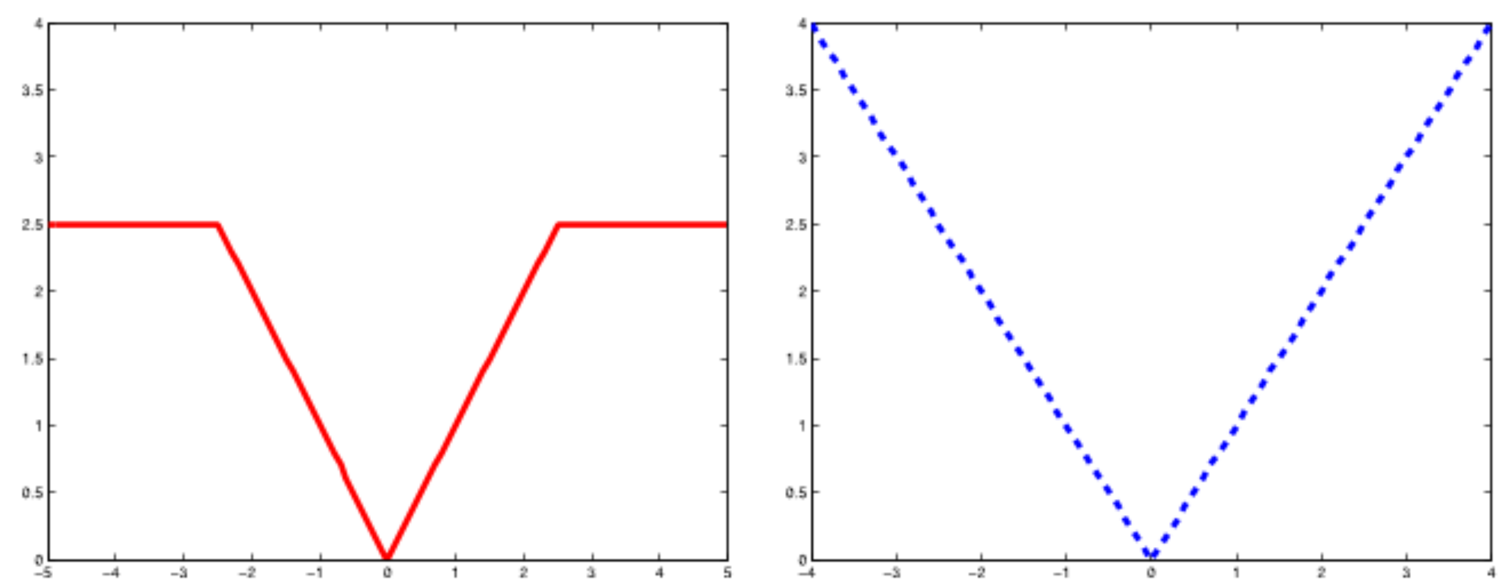
Thought question: regularization

- “In regards to the bias variance trade-off and to figure 5.5, it seems that finding an optimum model complexity could in itself be an ML problem, where we minimize some f(lambda) for our regularization parameter. Therefore, can we not formulate a problem where given a set of function classes and possible parameters, which can find an optimal regularization parameter automatically?”
  - Any suggested objectives to minimize, min_lambda obj(lambda)?

Other regularizers

- Have discussed l2 and l1 regularizers
- Other examples:
  - elastic net regularization is a combination of l1 and l2 (i.e., l1 + l2): ensures a unique solution
  - capped regularizers: do not prevent large weights

Does this regularizer still protect against overfitting?



(a) Capped ℓ_1-norm loss ($\varepsilon = 2.5$)

(b) ℓ_1-norm loss

* Figure from “Robust Dictionary Learning with Capped l1-Norm”, Jiang et al., IJCAI 2015

Adding regularizers to GLMs

- How do we add regularization to logistic regression?
- We had an optimization for logistic regression to get w: minimize negative log-likelihood, i.e. minimize cross-entropy
- Now want to balance negative log-likelihood and regularizer (i.e., the prior for MAP)
- Simply add regularizer to the objective function

Adding a regularizer to logistic regression

- Original objective function for logistic regression

$$\arg\max_{\mathbf{w}} \sum_{i=1}^{n} \left((y_i - 1)\, \mathbf{w}^\top \mathbf{x}_i + \log\left(\frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}} \right) \right)$$

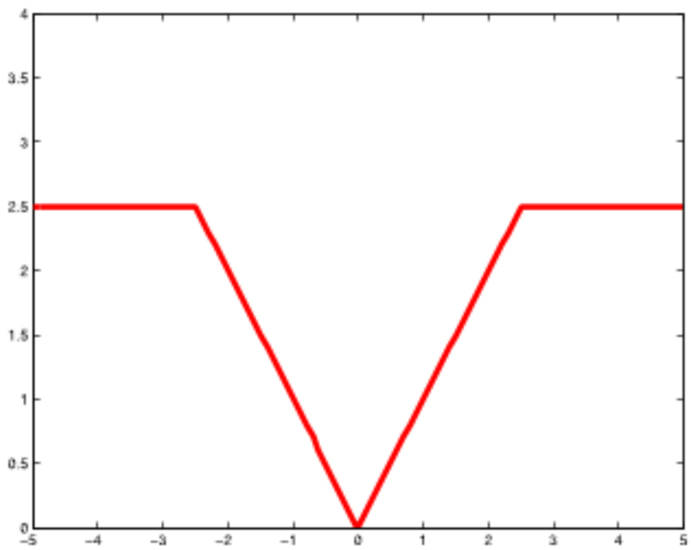
$$\arg\min_{\mathbf{w}} - \sum_{i=1}^{n} \left((y_i - 1)\, \mathbf{w}^\top \mathbf{x}_i + \log\left(\frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}} \right) \right)$$

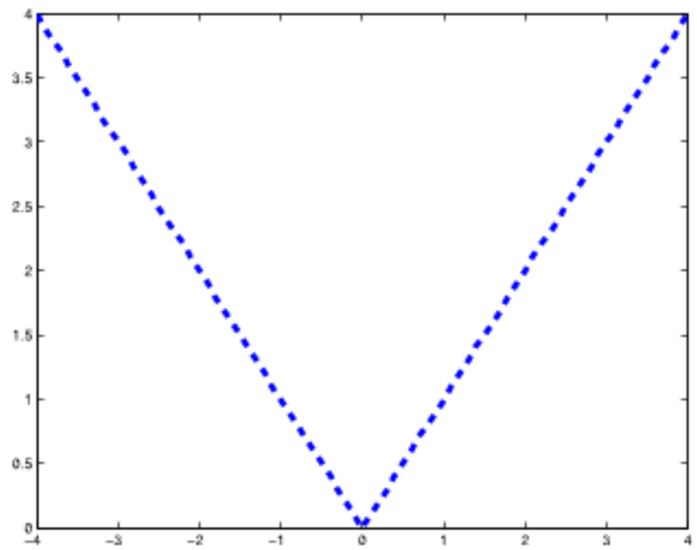
- Adding regularizer

$$\arg\min_{\mathbf{w}} - \sum_{i=1}^{n} \left((y_i - 1)\, \mathbf{w}^\top \mathbf{x}_i + \log\left(\frac{1}{1 + e^{-\mathbf{w}^\top \mathbf{x}_i}} \right) \right) \;\; + \lambda \|\mathbf{w}\|_2^2$$

Practical considerations: outliers

- What happens if one sample is bad?
- Regularization helps a little bit
- Can also change losses
- Robust losses
  - use l1 instead of l2
  - even better: use capped l1





- What are the disadvantages to these losses?

Weighting importance of samples and features

- Last class we added a weighting c_i >= 0 to indicate importance of a sample

- Now imagine you have added l1 regularization, for feature selection:

$$\lambda \|\mathbf{w}\|_1 = \lambda \sum_{j=1}^{d} |w_j|$$

- What if we want to weight features, and say feature j is a useful feature that should not be removed?

$$\sum_{j=1}^{d} \lambda_j |w_j|$$

  - could have different regularization parameter for each feature
  - large regularization parameter more likely to cause that feature to be pushed to zero or not used, with a small parameter saying that a feature should be used

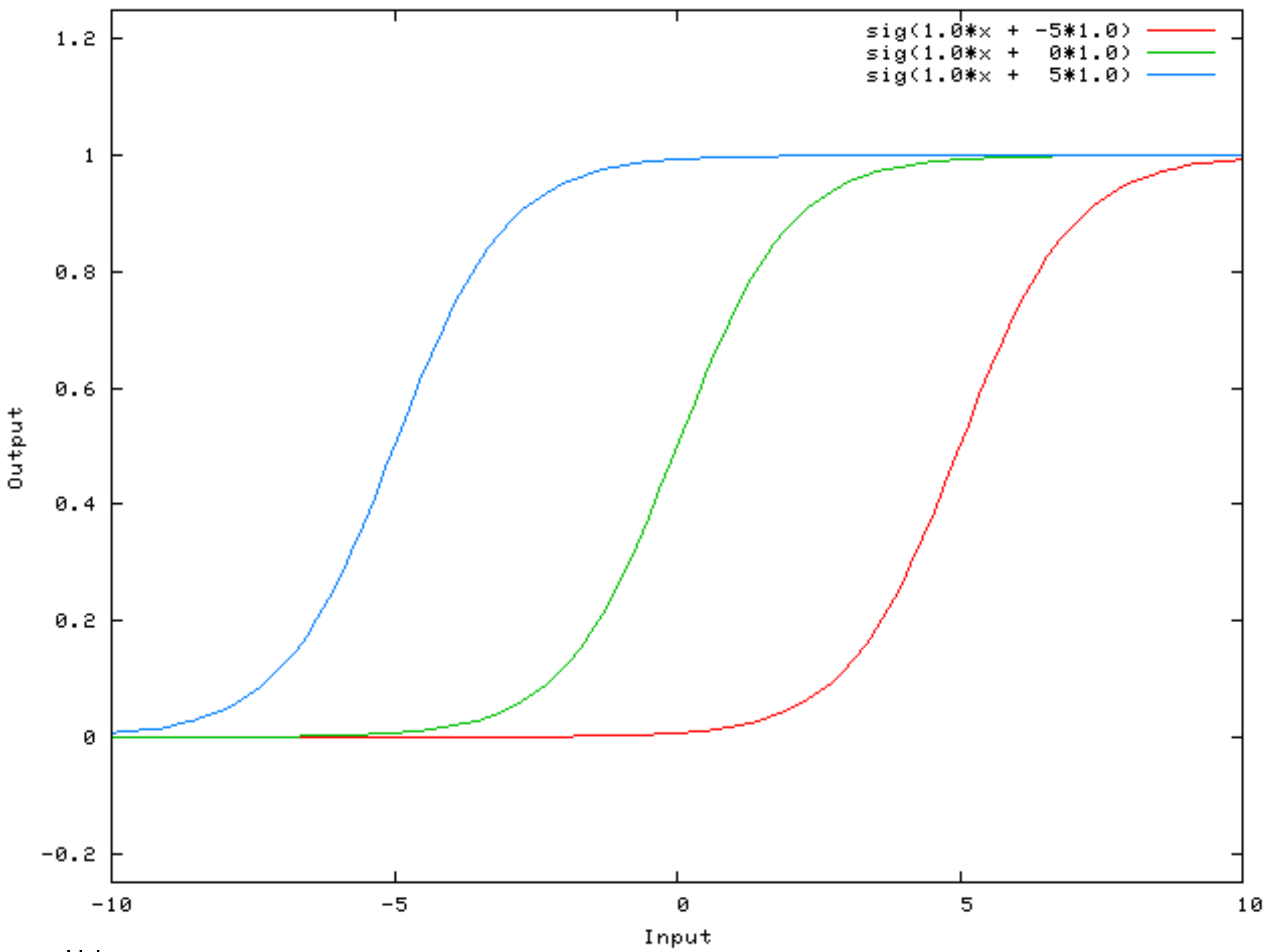
Exercise: intercept unit

- In linear regression, we added an intercept unit (bias unit) to the features
  - i.e., added a feature that is always 1 to the feature vector
- Does it make sense to do this for GLMs?
  - e.g., sigmoid(<x,w> + w_0)

Adding a column of ones to GLMs

- This provides the same outcome as for linear regression

- g(E[y l x]) = x w —> bias unit in x with coefficient w0 shifts the function left or right



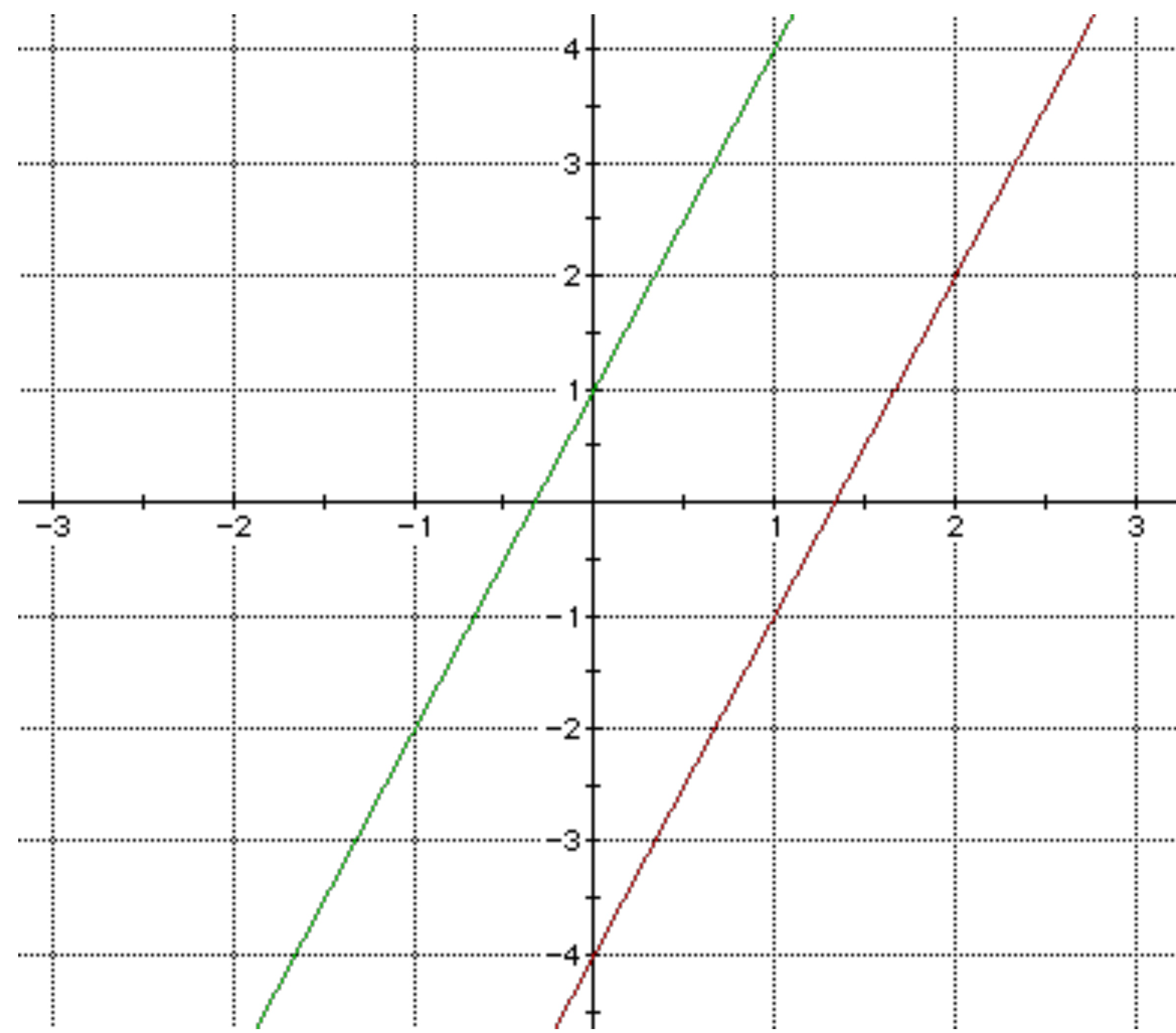


*Figure from http://stackoverflow.com/questions/2480650/role-of-bias-in-neural-networks

Multi-class and Multi-label

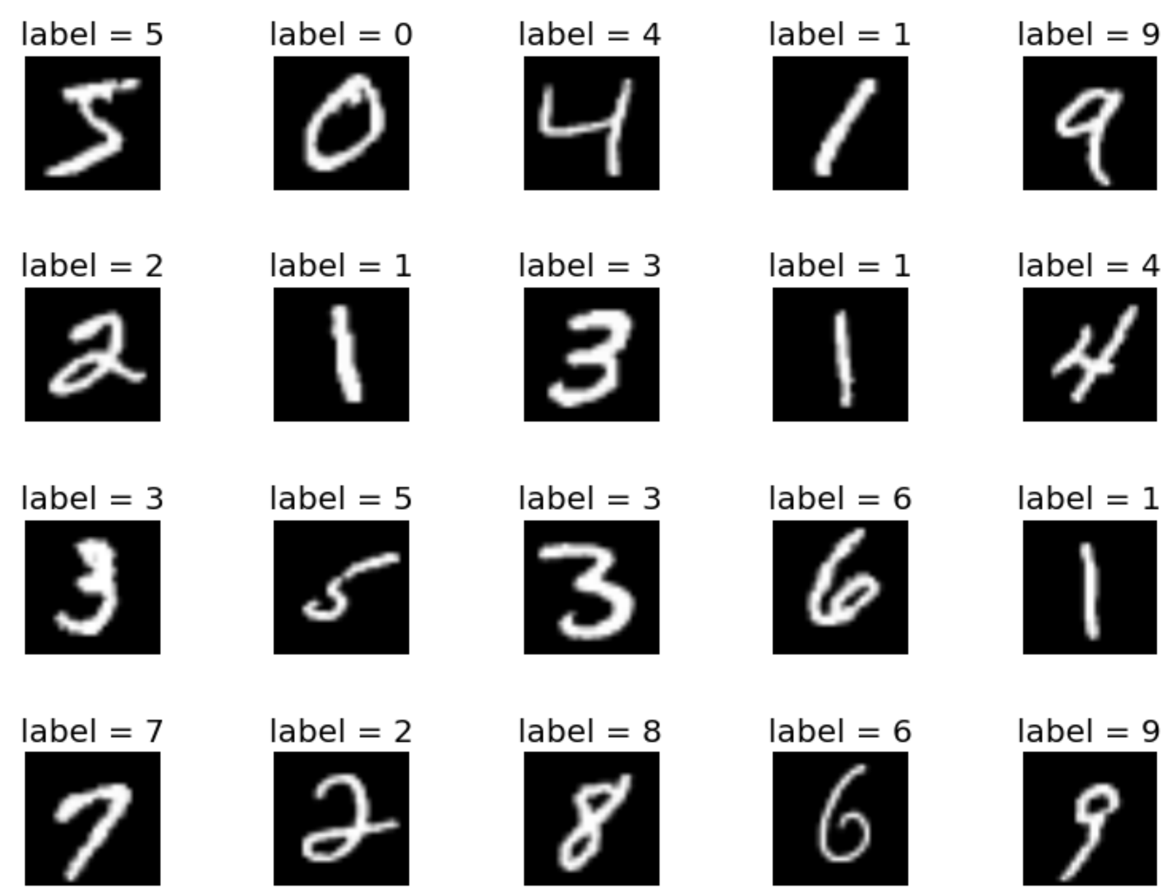
- Multi-class: have multiple classes, with each instance only in one class
  - e.g., a person can only have one blood type
- Multi-label: have multiple classes, where each instance can have multiple class labels
  - e.g., a newspaper article can be a sports article and medicine article

Exercise: problem representation for classification

- What if have many classes (e.g., image classification)?
    - Example: classify written digit (e.g., 7 or 3)
    - Example: classify images based on presence of an object (e.g., cat)
    - Is image classification multi-class or multi-label?

- What other settings can you imagine with many classes?

How can we learn this?



Exercise: problem representation for classification

- What if have many many classes (e.g., image classification)?
  - Example: classify written digit (e.g., 7 or 3)
  - Example: classify images based on presence of an object (e.g., cat)
  - Is image classification multi-class or multi-label?
- One approach for either multi-class or multi-label: learn one logistic regression model for each class
  - For each sample,
  - What are the issues here?
- What other techniques can we use for multi-class and multi-label?

One-vs-all

- Learn binary classifier for each class, w1, …, wk
  - i.e., weights w2 predict if the sample is either class 2 or its not
  - If training sample x is class 2, then weights w2 get label y = 1 and the weights wi for the other classes get a label of y = 0
- Once have wi, how do we predict on a new sample?
  - e.g., w1, w2 w3,
  - p(y = 1 | x, w1) = 0.9. p(y = 1 | x, w2) = 0.6. p(y = 1 | x, w3) = 0.1
- For multi-class, pick class such that p(y = 1 | x, wi) is largest
  - In this example that is class 1
- For multi-label, pick classes such that p(y = 1 | x, wi) > 0.5
  - In this example that is class 1 and 2

One-vs-all for multi-class

- What are the issues with this approach for multi-class?
    - see many more negative samples
    - have to compare confidence p(y=1 | x) between different classes, but scale could be different
- Learn binary classifier for each class, w1, …, wk
    - i.e., weights w2 predict if the sample is either class 2 or its not
    - If training sample x is class 2, then weights w2 get label y = 1 and the weights wi for the other classes get a label of y = 0
- Once have wi, how do we predict on a new sample?
    - p(y = 1 | x, w1) = 0.9. p(y = 1 | x, w2) = 0.6. p(y = 1 | x, w3) = 0.1

One-vs-all for multi-label

- Called binary relevance for multi-label

- What are the issues with one-vs-rest for multi-label?
  - class independence assumption
  - Do not take advantage of relationships between classes

- Learn binary classifier for each class, w1, …, wk
  - i.e., weights w2 predict if the sample is either class 2 or its not
  - If training sample x is class 2, then weights w2 get label y = 1 and the weights wi for the other classes get a label of y = 0

- Once have wi, how do we predict on a new sample?
  - $p(y = 1 \mid x, w1) = 0.9$. $p(y = 1 \mid x, w2) = 0.6$. $p(y = 1 \mid x, w3) = 0.1$

One-vs-one for multi-class

- Learn k (k-1)/2 binary classifiers, with voting scheme: class with most positive predictions is outputted

- For k = 3 (three classes), train class 1 vs class 2 (p_12), class 1 vs 3 (p_13), class 2 vs 3 (p_23)

- Predict with

$$f(\mathbf{x}) = \arg \max_{i \in \{1,2,3\}} \sum_j p_{ij}(y = 1(\text{i.e., } y = i \text{ not } j | \mathbf{x})$$

- Notice this uses p_{31}, but I didn't train that. Where does it come from?

Advantages to vs-all or vs-one

- Imagine you have a dataset with n samples, d features, k classes
- When might vs-all or vs-one be better?
  - Vs-one has to train about k^2 models, can be expensive!
  - But, gets to train k^2 models on a subset of the data (about 2 n / k if the data is balanced, i.e., equal number of each class)
- If n = 1 million and k = 10, which one might be better?
- Which learning methods might prefer one or the other?
  - If method scales poorly with sample, vs-all might be better
  - If method can share solutions across classes, vs-all might be better

Other approaches for multi-class

- naive Bayes (generative model)
- Instance-based approaches (e.g. k-NN)
  - keep a representative set of samples, compare to these points and what labels they had
- Hierarchical classification
- Multinomial logistic regression

Multinomial distributions

- Extend binary GLM (logistic regression) to multi-class, by moving from Bernoulli to Multinomial
- What does target y look like?
- y = [0 1 0 0] means that instance is in class 2 out of 4 classes
- What distribution matches such a target?
  - Clearly not Bernoulli, where its only zero or 1
  - Clearly not Gaussian…

Multinomial distributions

- Extend binary GLM (logistic regression) to multi-class, by moving from Bernoulli to Multinomial

- Multinomial distribution is probability of n successes in k Bernoulli trials

$$f(x_1, \ldots, x_k; n, p_1, \ldots, p_k) = \Pr(X_1 = x_1 \text{ and } \ldots \text{ and } X_k = x_k)$$

$$= \begin{cases} \dfrac{n!}{x_1! \cdots x_k!} p_1^{x_1} \cdots p_k^{x_k}, & \text{when } \sum_{i=1}^k x_i = n \\ \\ 0 & \text{otherwise,} \end{cases}$$

We have n = 1, e.g. y = [1 0 0 0]

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{y_1! \ldots y_k!} p(y_1 = 1|\mathbf{x})^{y_1} \ldots, p(y_k = 1|\mathbf{x})^{y_k}$$

Predictions

- Targets look like y = [0 1 0 0], meaning that instance is in class 2 out of 4 classes
- For a new sample, we predict [p(y=1 | x), p(y = 2| x), …, p(y = k | x)]
- Example: [0.1 0.2 0.6 0.1] suggests we should pick class y = 3, since it has the highest probability
- How do we generate such a vector of probabilities?

Multinomial logistic regression

- y = [0 1 0 0] means that instance is in class 2 out of 4 classes
- Let k be the number of classes, n = 1 for 1 success

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{y_1! \dots y_k!} p(y_1 = 1|\mathbf{x})^{y_1} \dots, p(y_k = 1|\mathbf{x})^{y_k}$$

- The transfer (inverse of link) is the softmax transfer

$$\begin{aligned}\mathrm{softmax}(\mathbf{x}^\top \mathbf{W}) &= \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \mathbf{w}_j)}, \dots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \mathbf{w}_j)}\right] \\ &= \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})}, \dots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})}\right]\end{aligned}$$

Softmax transfer

$$\mathrm{softmax}(\mathbf{x}^\top \mathbf{W}) = \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \mathbf{w}_j)}, \ldots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \mathbf{w}_j)}\right]$$
$$= \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})}, \ldots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})}\right]$$

Normalization to ensure that get valid probabilities

Must set vector of weights w_k = 0 to make softmax an invertible transfer

Relation to logistic regression

- For k= 2, y = [0 1] or y = [1 0]

$$\mathrm{softmax}(\mathbf{x}^\top \mathbf{W}) = \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \mathbf{w}_j)}, \ldots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\sum_{j=1}^k \exp(\mathbf{x}^\top \mathbf{w}_j)} \right]$$
$$= \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})}, \ldots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})} \right]$$

$$\mathbf{W} = [\mathbf{w},\ \mathbf{0}]$$

$$\begin{aligned} p(y = 0 | \mathbf{x}) &= \frac{\exp(\mathbf{x}^\top \mathbf{w})}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})} \\ &= \frac{\exp(\mathbf{x}^\top \mathbf{w})}{\exp(\mathbf{x}^\top \mathbf{w}) + \exp(\mathbf{x}^\top \mathbf{0})} \\ &= \frac{\exp(\mathbf{x}^\top \mathbf{w})}{\exp(\mathbf{x}^\top \mathbf{w}) + 1} \\ &= \sigma(\mathbf{x}^\top \mathbf{w}). \end{aligned}$$

Relation to logistic regression…

- In general, setting w_k to zero is a convention
  - could choose any of the classes, e.g., could learn p(y = 1 | x)

- Normalization enforces constraint on w_k (or on one of the classes), so can set w_k = 0
  - **Exercise**: show that setting w_k = 0 is also required to ensure that the softmax transfer is invertible

$$
\begin{aligned}
\mathrm{softmax}(\mathbf{x}^\top \mathbf{W}) &= \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\sum_{j=1}^{k} \exp(\mathbf{x}^\top \mathbf{w}_j)}, \ldots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\sum_{j=1}^{k} \exp(\mathbf{x}^\top \mathbf{w}_j)} \right] \\
&= \left[\frac{\exp(\mathbf{x}^\top \mathbf{w}_1)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})}, \ldots, \frac{\exp(\mathbf{x}^\top \mathbf{w}_k)}{\mathbf{1}^\top \exp(\mathbf{x}^\top \mathbf{W})} \right]
\end{aligned}
$$

Summary of multinomial logistic regression

- p(y | x) is a multinomial distribution
- Corresponding transfer is the softmax transfer with w_k = 0
- Prediction on new x is
- softmax(xW) = [p(y=1 | x), p(y = 2| x), …, p(y = k | x)]

Learning strategy

- Using the minimization obtained for our generalized linear models, we can plug in the transfer to get

$$\min_{\mathbf{W} \in \mathbb{R}^{d \times k} : \mathbf{W}_{:k} = \mathbf{0}} \sum_{i=1}^{n} \log\left(\mathbf{1}^\top \exp(\mathbf{x}_i^\top \mathbf{W})\right) - \mathbf{x}_i^\top \mathbf{W} \mathbf{y}_i$$

with gradient

$$\nabla \sum_{i=1}^{n} \left(\log\left(\mathbf{1}^\top \exp(\mathbf{x}_i^\top \mathbf{W})\right) - \mathbf{x}_i^\top \mathbf{W} \mathbf{y}_i \right) = \sum_{i=1}^{n} \frac{\exp(\mathbf{x}_i^\top \mathbf{W})^\top \mathbf{x}_i^\top}{\mathbf{1}^\top \exp(\mathbf{x}_i^\top \mathbf{W})} - \mathbf{x}_i \mathbf{y}_i^\top .$$

How do we constrain w_k = 0?

Exercise: nonlinear GLMs

- In linear regression, we used nonlinear expansions on the features to get nonlinear learning
    - e.g., convert x to polynomials
- Can we do the same for logistic regression or multinomial logistic regression?
- Why would we want to? Aren't GLMs already nonlinear?