CSC 411 Lecture 07: Multiclass Classification

Ethan Fetaya, James Lucas and Emad Andrews

University of Toronto

The Midterm

Mid-term Key Info

- Time and Date: **7:10pm - 8:10pm** on **Friday Oct. 20th**
- Location: **MS3153(A-L)/MS3154(M-Z)**

There is an alternate seating on this date from 8-9pm. The **only way** you can write in the alternate seating is if you have a scheduled conflict at the time of the first seating.

Midterm - Alternate Seating

To attend the alternate seating:

1. Send me an email (csc411-20179-instrs@cs.toronto.edu) **by 10:00 PM October 9** containing a screenshot (or pdf) from your ROSI/ACORN where your name and student number are visible and clearly show a regularly scheduled class or lab at University of Toronto which overlaps with the regular midterm. If you send me such an email on October 10, you won't be seated in the alternate seating. Your email must contain the words "CSC411 Test Conflict with regular midterm" in the subject line. **Even if you sent me an email earlier in the term, send another one** to make sure you get your room number and approval.
2. If in fact it is a course conflict, I'll give you the location for the alternate seating. However, having other midterms that day (with times not overlapping ours) does **NOT** count as a conflict.

Midterm - Makeup Test

There will also a makeup test for students who cannot attend **BOTH** the regular time and alternate seating exams for a valid reason(medical documentation or time conflict prove), by doing the following.

Please follow the instructions from the previous slide - providing equivalent proof that you cannot attend **BOTH** the regular midterm and the additional seating.

If you cannot take the scheduled exams on Friday Oct. 20 for medical/emergency reasons, you must contact your instructor immediately to obtain special permission and provide proper documentations.

Missing The Midterm Exam

If you skip a test without prior approval, you will receive a zero for the test (unless the absence is due to an illness or exceptional circumstances and properly documented). If you cannot show up for the test because of illness, you should submit your medical documentation to your instructor no later than one week after the day of the test.

Today

Multi-class classification with:

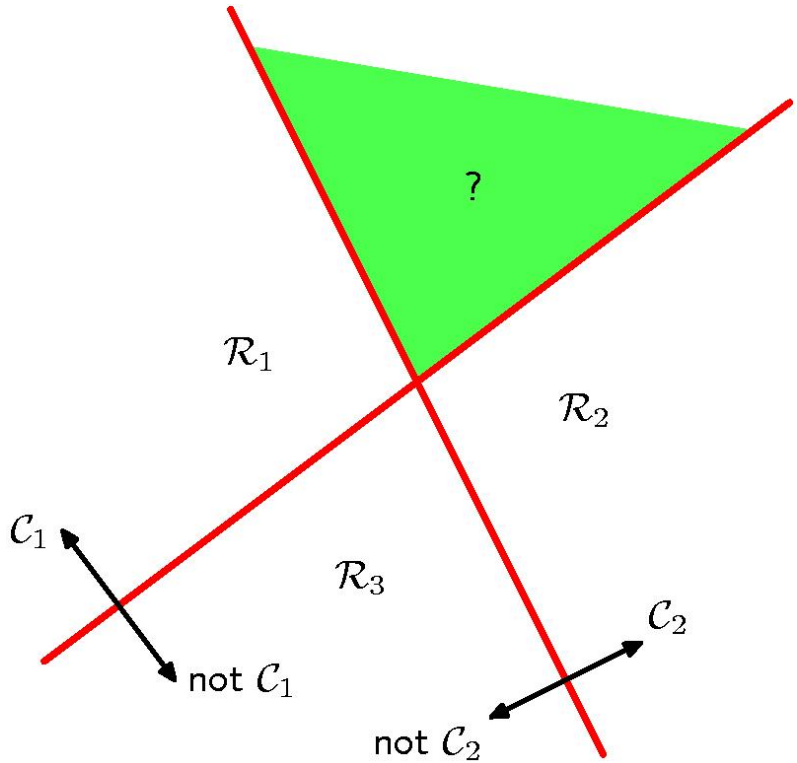
- Logistic Regression
- K-NN
- Decision trees

Reduction to binary case.

- "if all you have is a hammer, everything looks like a nail"
  - We have binary classifiers, can we reduce to that?
- How can that be done?
  - One vs all
  - One vs one
  - Other ideas: Hierarchical Classification, Error correcting codes.

One vs All

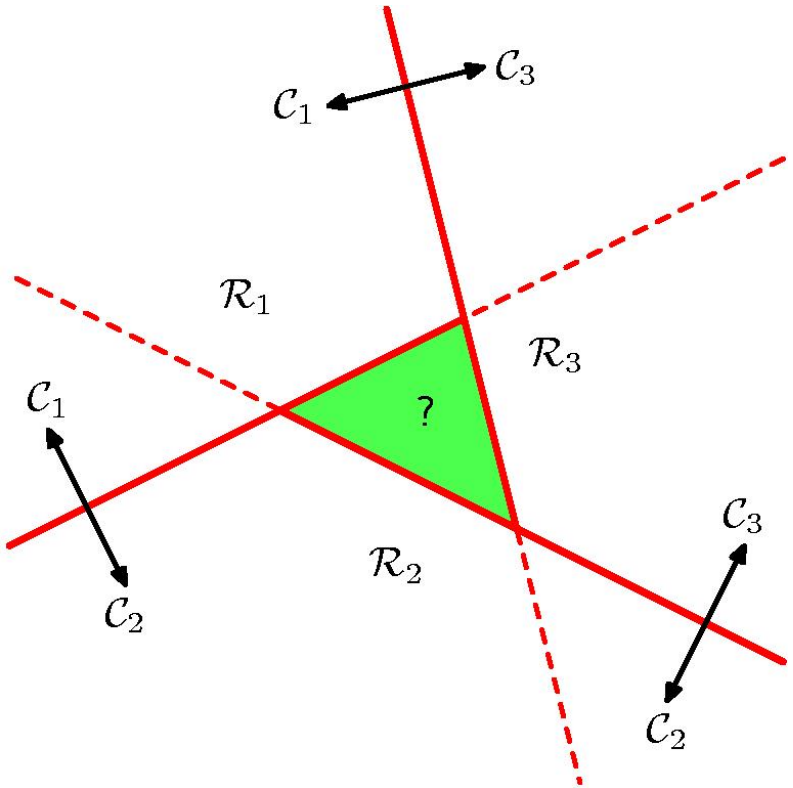
- First idea: Use $K-1$ classifiers, each solving a two class problem of separating point in a class C_k from points not in the class.
- Known as **1 vs all** or **1 vs the rest** classifier



- Each classifier partitions the space with a decision boundary
- PROBLEM: More than one good answer for green region!

One vs One

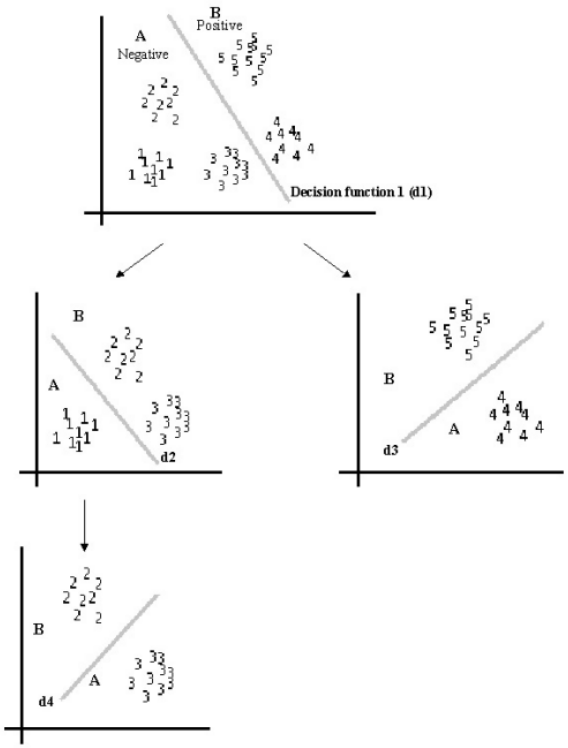
- Another simple idea: Introduce $K(K-1)/2$ two-way classifiers, one for each possible pair of classes
- Each point is classified according to majority vote amongst the disc. func.
- Known as the **1 vs 1 classifier**



- PROBLEM: Two-way preferences need not be transitive

Hierarchical Classification *

- Hierarchical Classification - classify by a sequence of binary decisions
  - Similar to decision tree - but on the labels



- How to decide the hierarchy?
- Problems: Sensitive to single mistake, decision can be harder then 1-vs-1 or 1-vs-all

Error correcting codes *

- Each binary classifier h_i gives some classes label 1 and some zero.
- Binary classifiers $h_1, .., h_L$ give each class a binary code, e.g. $[0, 0, 1, 0, 1]$
- Idea - use error correcting codes. Two separate class codes should be very different.
- Should be robust to several classifier errors.
- Problem: The binary classifiers, e.g. even vs odd numbers, can be hard to train.

K-Class Discriminant

- We can avoid these problems by considering a single K-class discriminant comprising K functions of the form

$$y_k(\mathbf{x}) = \mathbf{w}_k^T \mathbf{x} + w_{k,0}$$

and then assigning a point $\mathbf{x}$ to class C_k if

$$\forall j \neq k \qquad y_k(\mathbf{x}) > y_j(\mathbf{x})$$

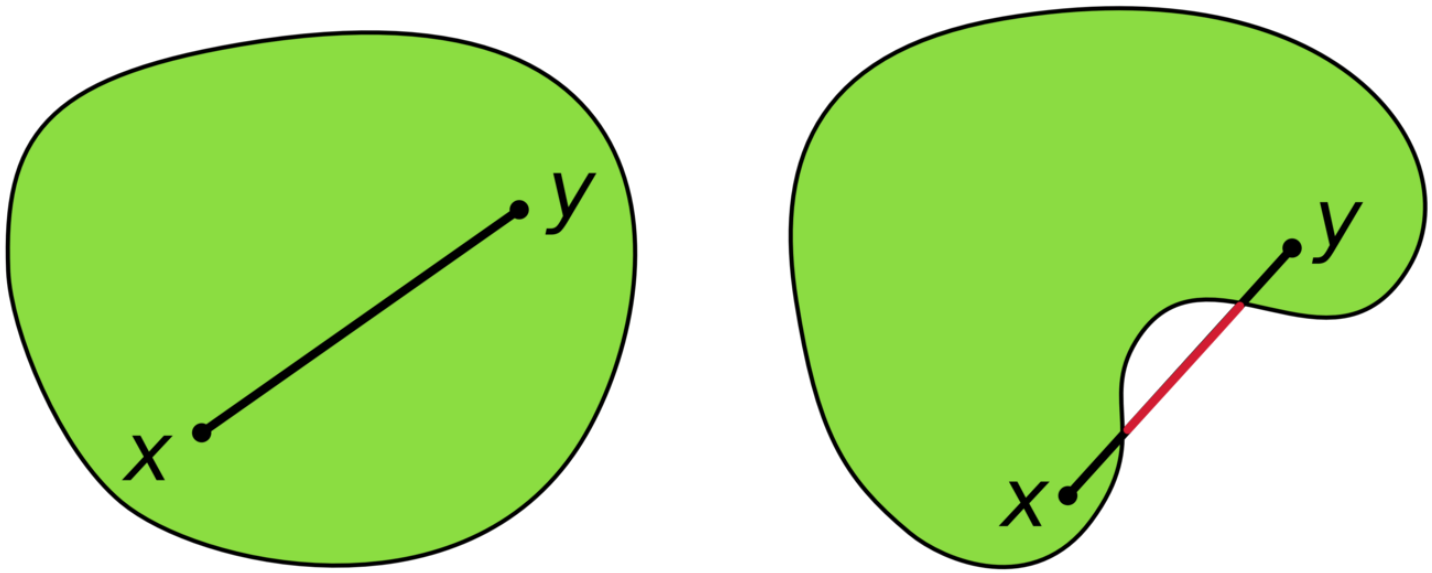
- Note that $\mathbf{w}_k^T$ is now a vector, not the k-th coordinate
- The decision boundary between class C_j and class C_k is given by $y_j(\mathbf{x}) = y_k(\mathbf{x})$, and thus it's a $(D-1)$ dimensional hyperplane defined as

$$(\mathbf{w}_k - \mathbf{w}_j)^T \mathbf{x} + (w_{k0} - w_{j0}) = 0$$

- What about the binary case? Is this different?
- What is the shape of the overall decision boundary?

K-Class Discriminant

- The decision regions of such a discriminant are always **convex**
- In Euclidean space, an object is **convex** if for every pair of points within the object, every point on the straight line segment that joins the pair of points is also within the object

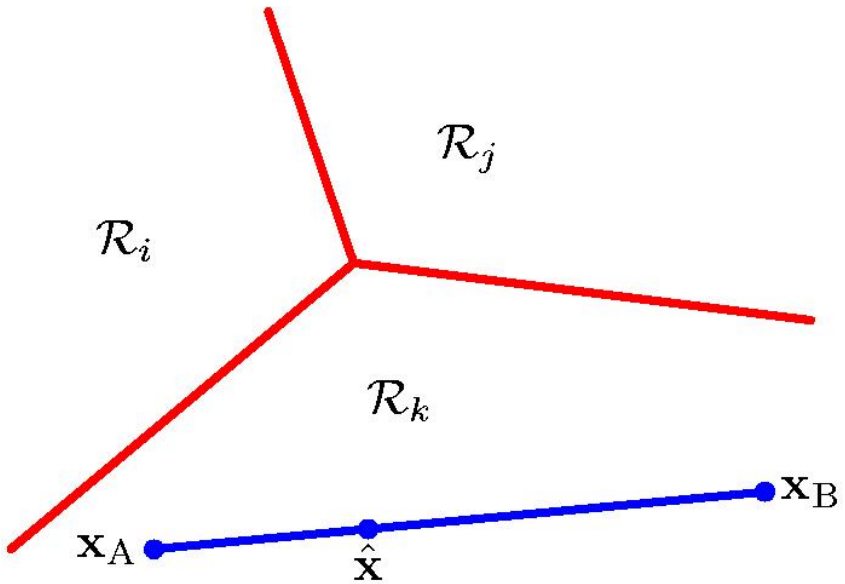


- Which object is convex?

K-Class Discriminant

- The decision regions of such a discriminant are always **convex**
- Consider 2 points $\mathbf{x}_A$ and $\mathbf{x}_B$ that lie inside decision region R_k
- Any convex combination $\hat{\mathbf{x}}$ of those points also will be in R_k

$$\hat{\mathbf{x}} = \lambda \mathbf{x}_A + (1 - \lambda)\mathbf{x}_B$$



Proof

- A convex combination point, i.e., $\lambda \in [0, 1]$

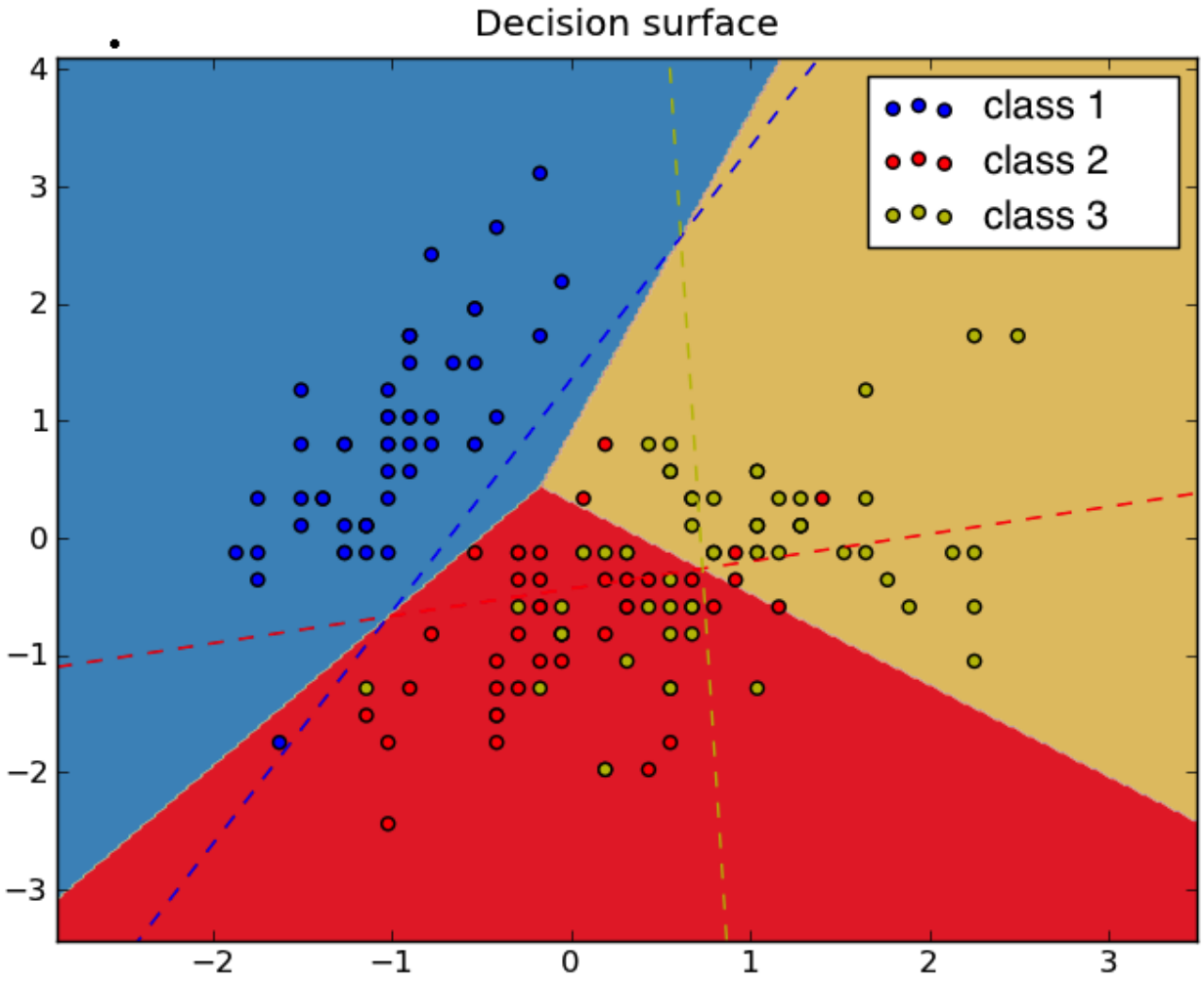
$$\hat{\mathbf{x}} = \lambda \mathbf{x}_A + (1 - \lambda)\mathbf{x}_B$$

- From the linearity of the classifier $y(\mathbf{x})$

$$y_k(\hat{\mathbf{x}}) = \lambda y_k(\mathbf{x}_A) + (1 - \lambda) y_k(\mathbf{x}_B)$$

- Since $\mathbf{x}_A$ and $\mathbf{x}_B$ are in R_k, it follows that $y_k(\mathbf{x}_A) > y_j(\mathbf{x}_A)$, $y_k(\mathbf{x}_B) > y_j(\mathbf{x}_B)$, $\forall j \neq k$
- Since λ and $1 - \lambda$ are positive, then $\hat{\mathbf{x}}$ is inside R_k
- Thus R_k is convex

Example



Multi-class Logistic Regression

- Associate a set of weights with each class, then use a normalized exponential output

$$p(C_k|\mathbf{x}) = y_k(\mathbf{x}) = \frac{\exp(z_k)}{\sum_j \exp(z_j)}$$

where the activations are given by

$$z_k = \mathbf{w}_k^T \mathbf{x}$$

- The function $\frac{\exp(z_k)}{\sum_j \exp(z_j)}$ is called a softmax function
- Useful notation: One-hot encoding.
  - instead of using $t = k$ (target has label k) we use a vector of K target values containing a single 1 for the correct class and zeros elsewhere
  - *Example:* For a 4-class problem, we would write a target with class label 2 as:

$$\mathbf{t} = [0, 1, 0, 0]^T$$

Multi-class Logistic Regression

- The likelihood

$$p(\mathbf{T}|\mathbf{X}, \mathbf{w}_1, \cdots, \mathbf{w}_k) = \prod_{n=1}^{N}\prod_{k=1}^{K} p(C_k|\mathbf{x}^{(n)})^{t_k^{(n)}} = \prod_{n=1}^{N}\prod_{k=1}^{K} y_k^{(n)}(\mathbf{x}^{(n)})^{t_k^{(n)}}$$

with

$$p(C_k|\mathbf{x}) = y_k(\mathbf{x}) = \frac{\exp(z_k)}{\sum_j \exp(z_j)}$$

where n-th row of $\mathbf{T}$ is 1-of-K encoding of example n and

$$z_k = \mathbf{w}_k^T \mathbf{x} + w_{k0}$$

- What assumptions have I used to derive the likelihood?
- Derive the loss by computing the negative log-likelihood:

$$L(\mathbf{w}_1, \cdots, \mathbf{w}_K) = -\log p(\mathbf{T}|\mathbf{X}, \mathbf{w}_1, \cdots, \mathbf{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_k^{(n)} \log[y_k^{(n)}(\mathbf{x}^{(n)})]$$

This is known as the **cross-entropy** error for multiclass classification

- How do we obtain the weights?

Training Multi-class Logistic Regression

- How do we obtain the weights?

$$L(\mathbf{w}_1, \cdots, \mathbf{w}_K) = -\log p(\mathbf{T}|\mathbf{X}, \mathbf{w}_1, \cdots, \mathbf{w}_K) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_k^{(n)} \log[y_k^{(n)}(\mathbf{x}^{(n)})]$$

- Do gradient descent, where the derivatives are

$$\frac{\partial y_j^{(n)}}{\partial z_k^{(n)}} = \frac{\partial}{\partial z_k^{(n)}}\left(\frac{\exp(z_j^{(n)})}{\sum_l \exp(z_l^{(n)})}\right) = \delta(k,j)y_j^{(n)} - y_j^{(n)}y_k^{(n)}$$

and

$$\frac{\partial L}{\partial z_k^{(n)}} = \sum_{j=1}^{K} \frac{\partial L}{\partial y_j^{(n)}} \cdot \frac{\partial y_j^{(n)}}{\partial z_k^{(n)}} = y_k^{(n)} - t_k^{(n)}$$

$$\frac{\partial L}{\partial w_{k,i}} = \sum_{n=1}^{N} \frac{\partial L}{\partial z_k^{(n)}} \cdot \frac{\partial z_k^{(n)}}{\partial w_{k,i}} = \sum_{n=1}^{N} (y_k^{(n)} - t_k^{(n)}) \cdot x_i^{(n)}$$

- The derivative is the error times the input

Softmax for 2 Classes

- Let's write the probability of one of the classes

$$p(C_1|\mathbf{x}) = y_1(\mathbf{x}) = \frac{\exp(z_1)}{\sum_j \exp(z_j)} = \frac{\exp(z_1)}{\exp(z_1) + \exp(z_2)}$$

- I can equivalently write this as

$$p(C_1|\mathbf{x}) = y_1(\mathbf{x}) = \frac{\exp(z_1)}{\exp(z_1) + \exp(z_2)} = \frac{1}{1 + \exp\left(-(z_1 - z_2)\right)}$$

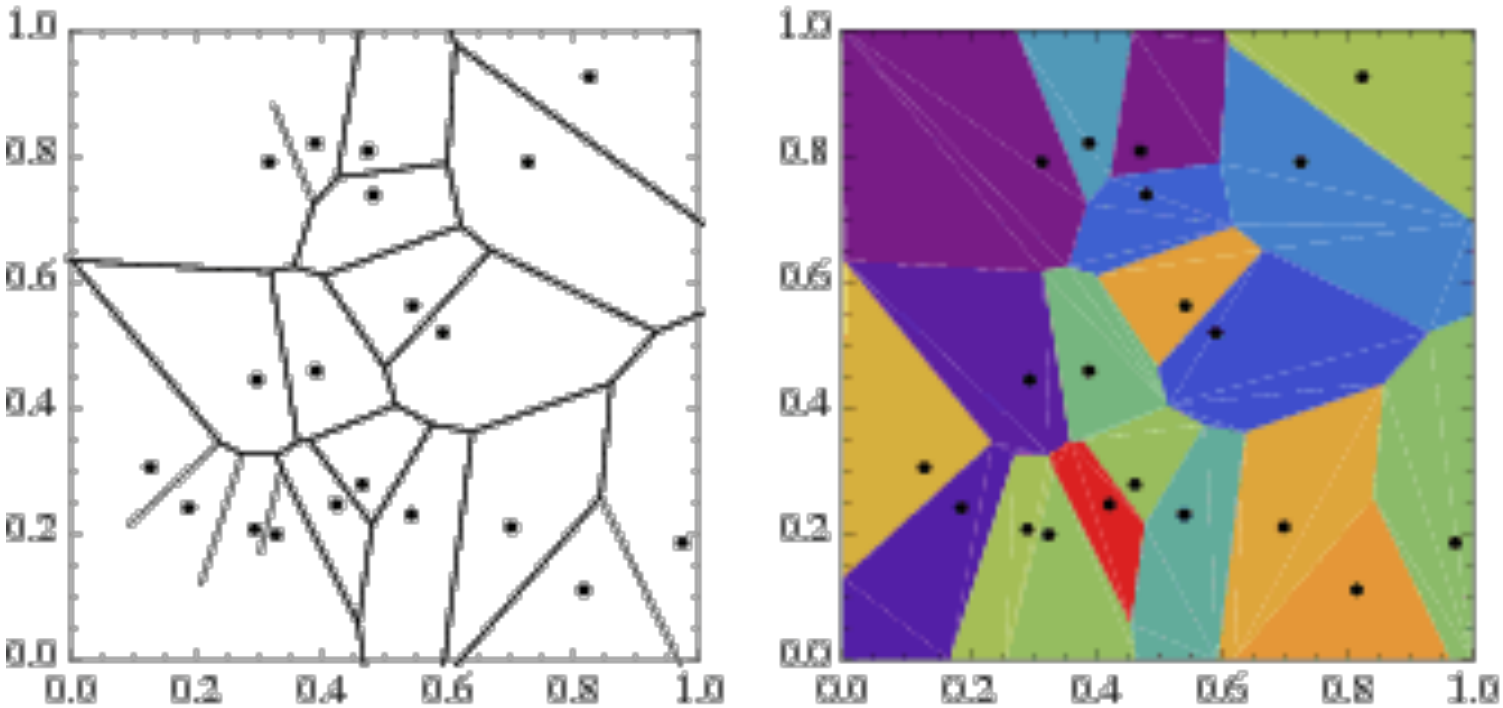
- So the logistic is just a special case that avoids using redundant parameters
- Rather than having two separate set of weights for the two classes, combine into one

$$z' = z_1 - z_2 = \mathbf{w}_1^T\mathbf{x} - \mathbf{w}_2^T\mathbf{x} = \mathbf{w}^T\mathbf{x}$$

- The over-parameterization of the softmax is because the probabilities must add to 1.

Multi-class K-NN

- Can directly handle multi class problems



Multi-class Decision Trees

- Can directly handle multi class problems
- How is this decision tree constructed?