Announcements

Assignments:

- HW5
  - Due date Thu, 2/27, 11:59 pm

Midterm

- See Piazza post for details
- Added Friday OH and OH appointments

Feedback

Plan

Last time

- Neural Networks
  - Universal Approximation
  - Optimization / Backpropagation

Today

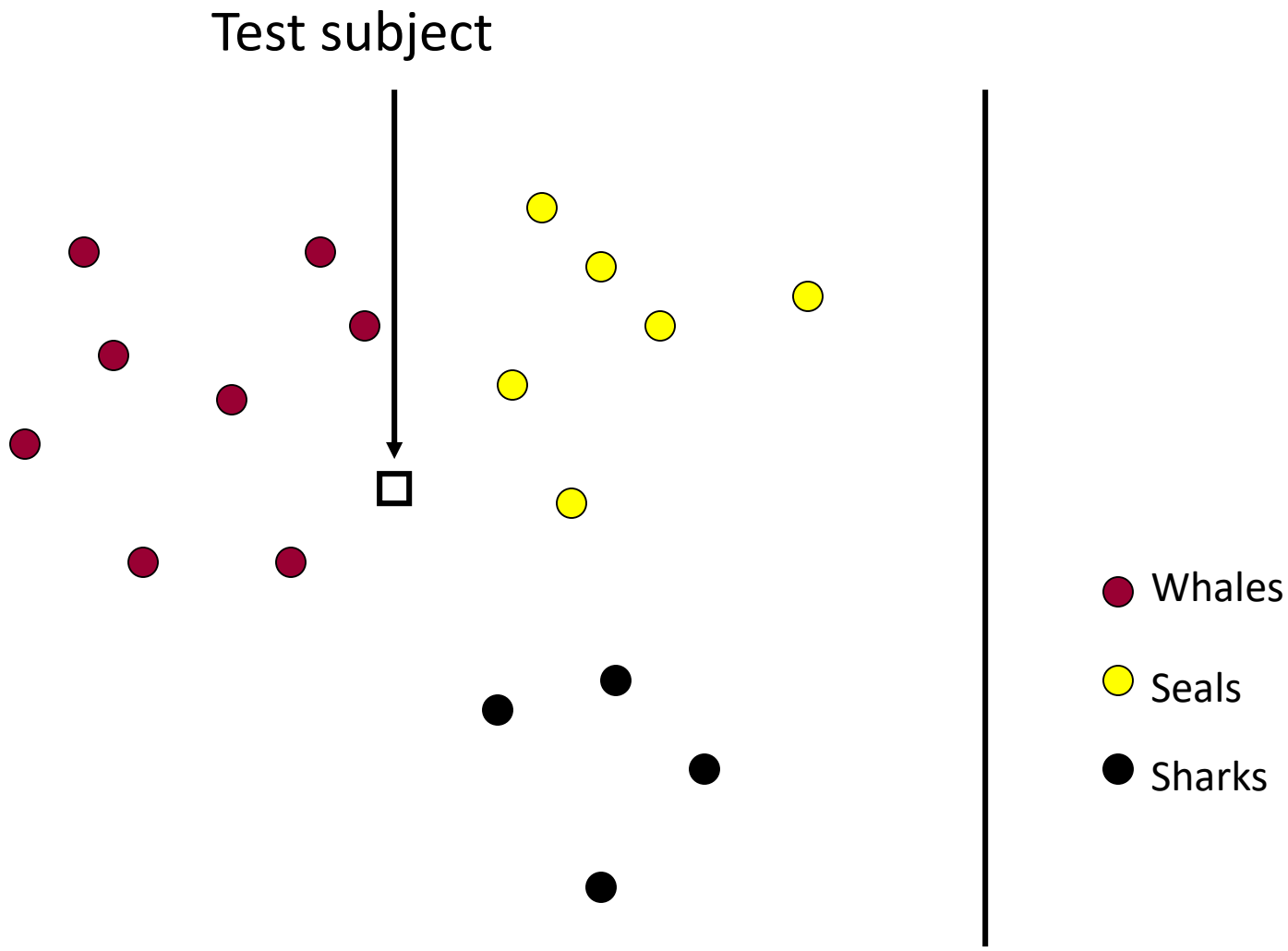
- Wrap-up Convolutional Neural Networks
- Nearest Neighbor Classification

Introduction to Machine Learning

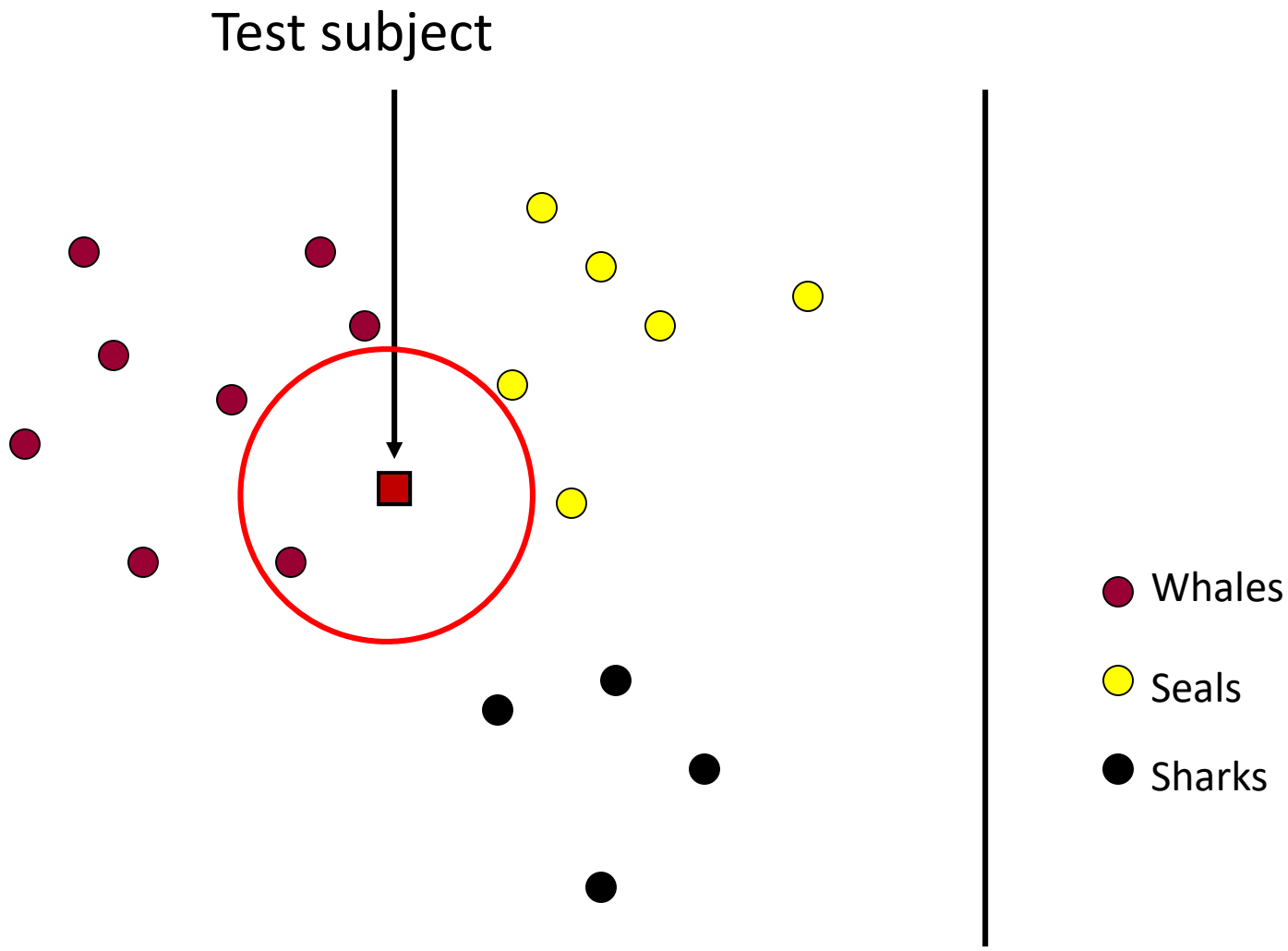
Nearest Neighbor

Instructor: Pat Virtue

Nearest Neighbor Classifier



Nearest Neighbor Classifier



Nearest Neighbor Classification

Given a training dataset $\mathcal{D} = \left\{y^{(n)}, \boldsymbol{x}^{(n)}\right\}_{n=1}^{N}$, $y \in \{1, \ldots, C\}$, $\boldsymbol{x} \in \mathbb{R}^m$

and a test input $\boldsymbol{x}_{test}$, predict the class label, $\hat{y}_{test}$:

1) Find the closest point in the training data to $\boldsymbol{x}_{test}$

$$n = \underset{n}{\operatorname{argmin}}\, d(\boldsymbol{x}_{test}, \boldsymbol{x}^{(n)})$$

2) Return the class label of that closest point

$$\hat{y}_{test} = y^{(n)}$$

Need distance function! What should $d(\boldsymbol{x}, \boldsymbol{z})$ be?

Fisher Iris Dataset

Fisher (1936) used 150 measurements of flowers from 3 different species: Iris setosa (0), Iris virginica (1), Iris versicolor (2) collected by Anderson (1936)

Species	Sepal Length	Sepal Width	Petal Length	Petal Width
0	4.3	3.0	1.1	0.1
0	4.9	3.6	1.4	0.1
0	5.3	3.7	1.5	0.2
1	4.9	2.4	3.3	1.0
1	5.7	2.8	4.1	1.3
1	6.3	3.3	4.7	1.6
1	6.7	3.0	5.0	1.7

Full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set

Fisher Iris Dataset

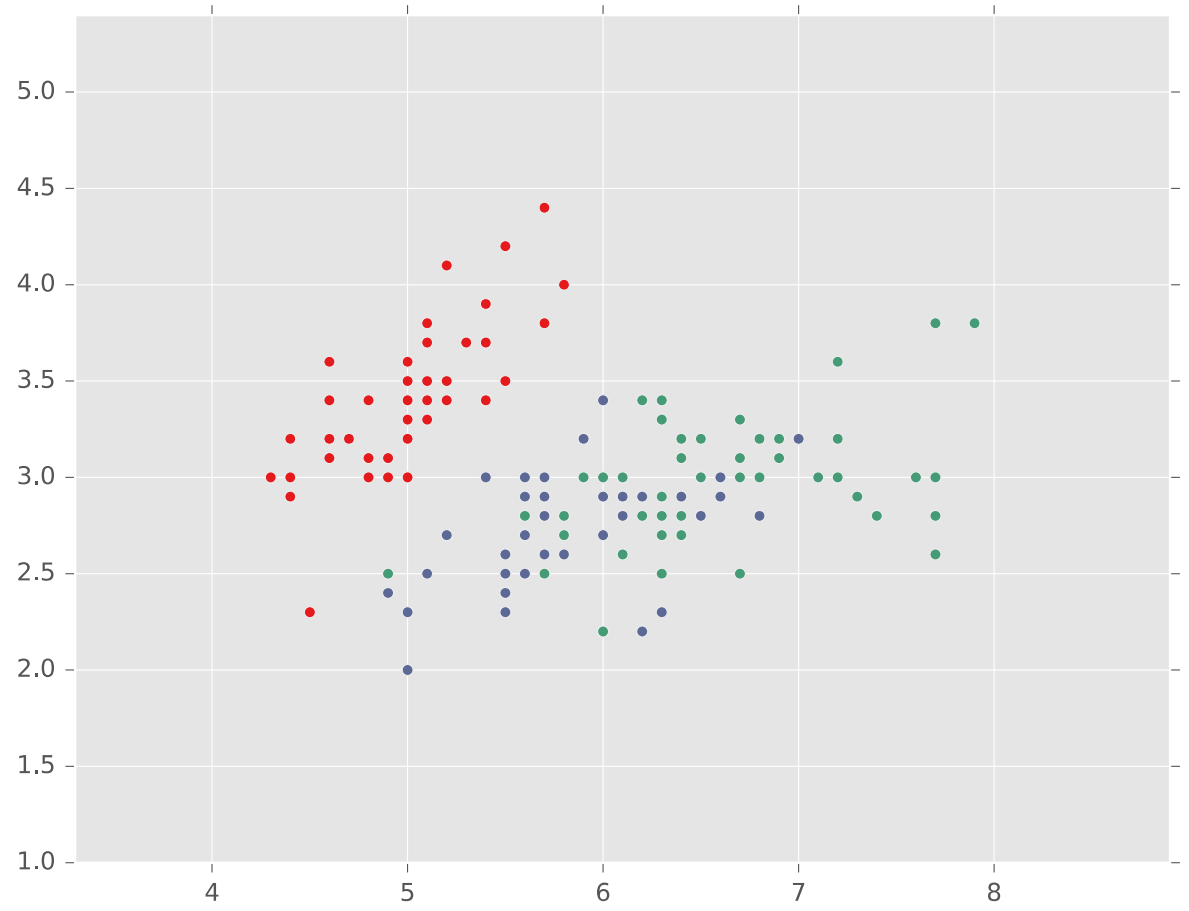
Fisher (1936) used 150 measurements of flowers from 3 different species: Iris setosa (0), Iris virginica (1), Iris versicolor (2) collected by Anderson (1936)

Species	Sepal Length	Sepal Width
0	4.3	3.0
0	4.9	3.6
0	5.3	3.7
1	4.9	2.4
1	5.7	2.8
1	6.3	3.3
1	6.7	3.0

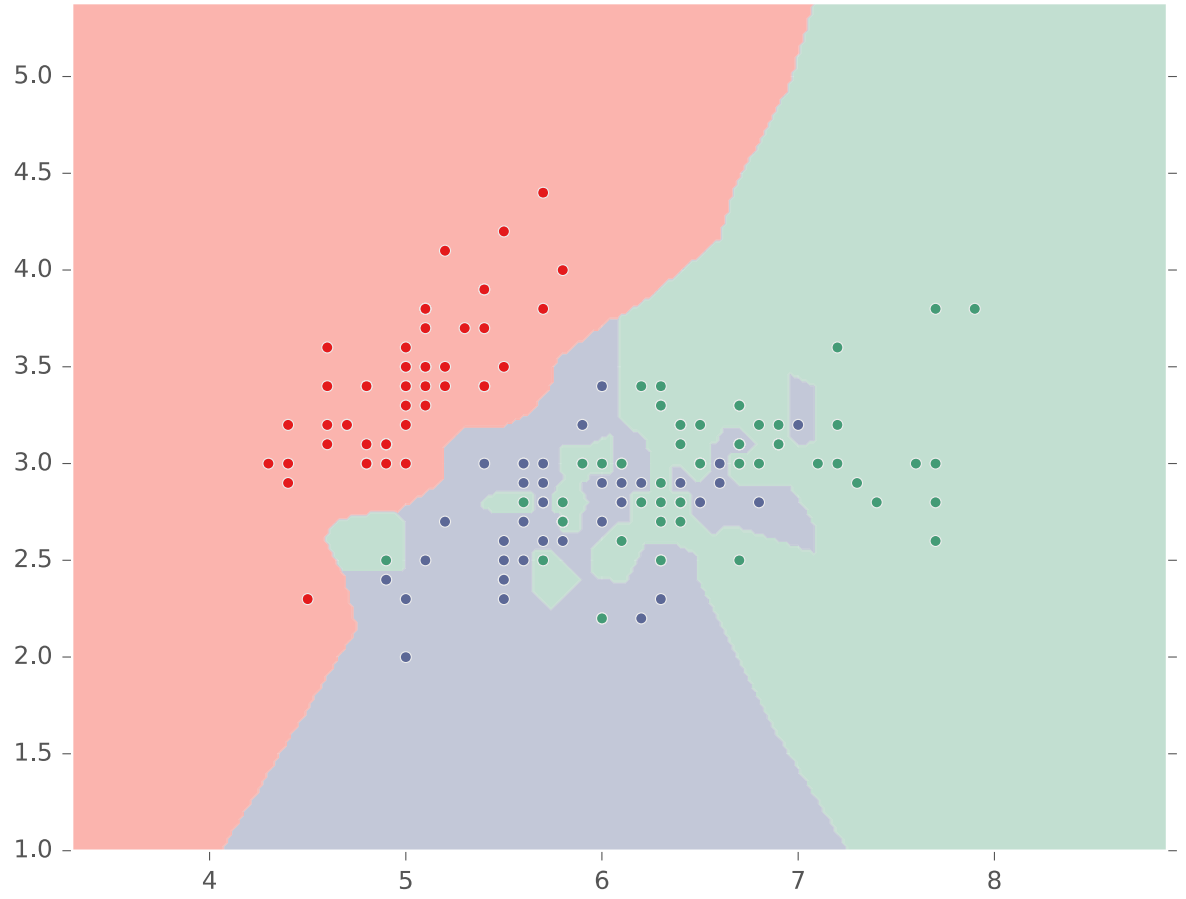
Deleted two of the four features, so that input space is 2D

Full dataset: https://en.wikipedia.org/wiki/Iris_flower_data_set

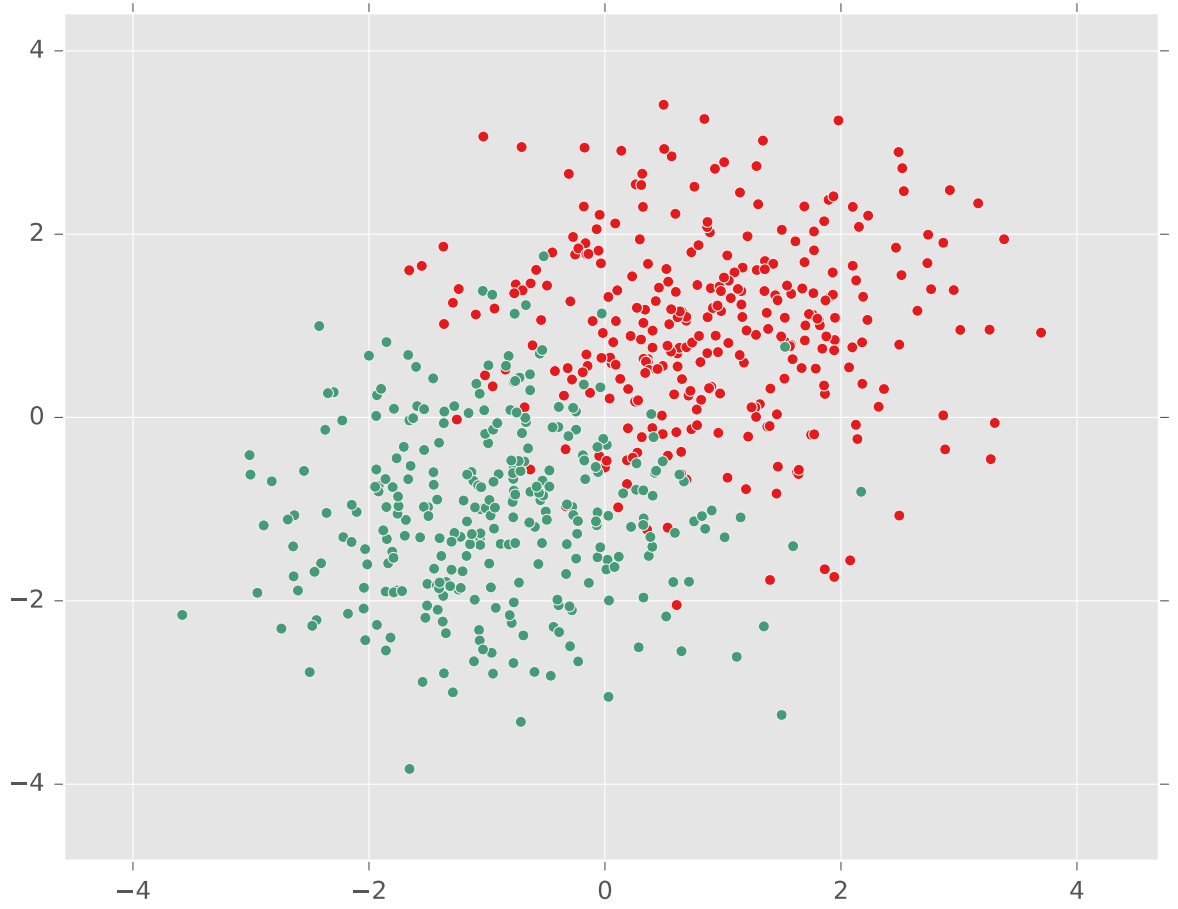
Nearest Neighbor on Fisher Iris Data



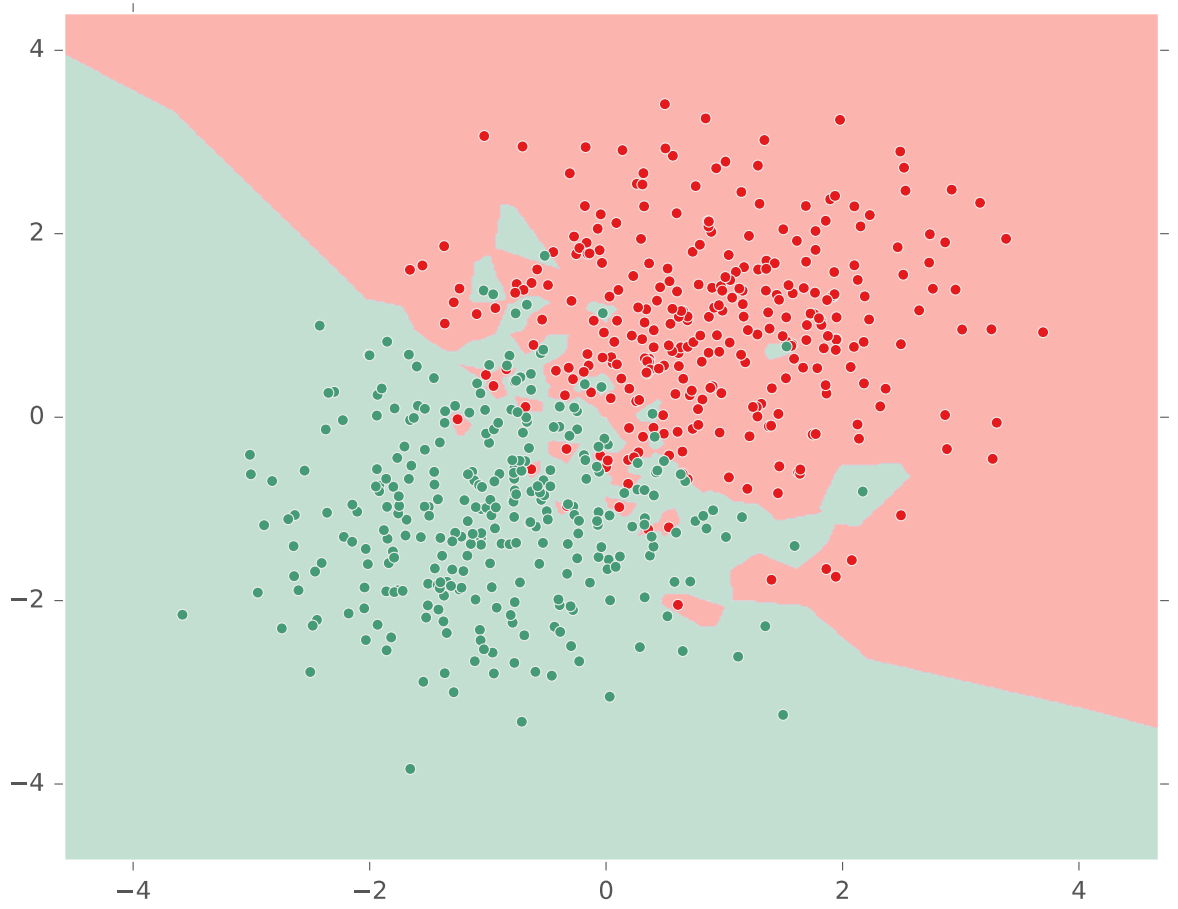
Nearest Neighbor on Fisher Iris Data



Nearest Neighbor on Gaussian Data



Nearest Neighbor on Gaussian Data



Parametric models

Assume some model (Gaussian, Bernoulli, Multinomial, logistic, network of logistic units, Linear, Quadratic) with fixed number of parameters

- Linear/Logistic Regression, Naïve Bayes, Discriminant Analysis, Neural Networks

Estimate parameters ($\mu, \sigma^2, \theta, w, \beta$) using MLE/MAP and plug in

Pro – need few data points to learn parameters

Con – Strong distributional assumptions, not satisfied in practice

Nonparametric models

Nonparametric: number of parameters scales with number of training data

- Typically don’t make any distributional assumptions
- As we have more data, we should be able to learn more complex models

Some nonparametric methods

- Nearest Neighbor (k-Nearest Neighbor) Classifier
- Decision Trees

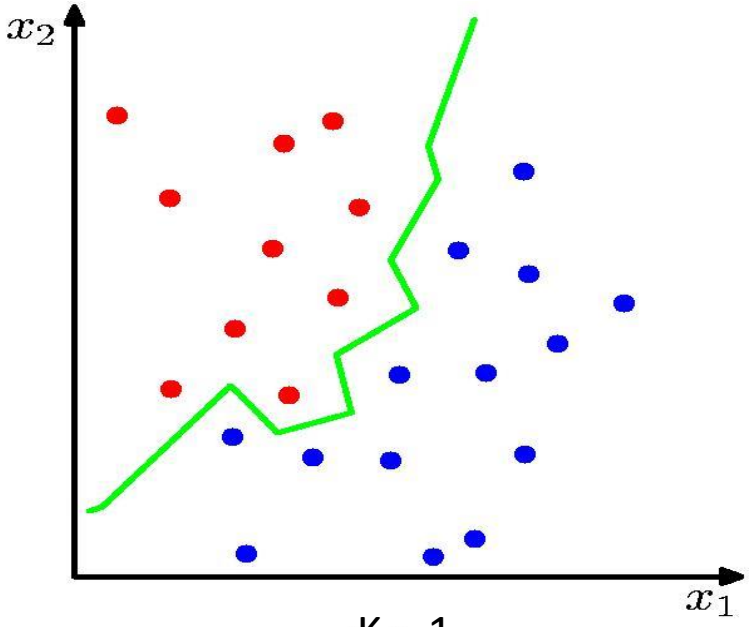
Parametric vs Nonparametric

- Nonparametric models place very mild assumptions on the data distribution and provide good models for complex data
- Parametric models rely on very strong (simplistic) distributional assumptions

- Nonparametric models requires storing and computing with the entire data set.
- Parametric models, once fitted, are much more efficient in terms of storage and computation

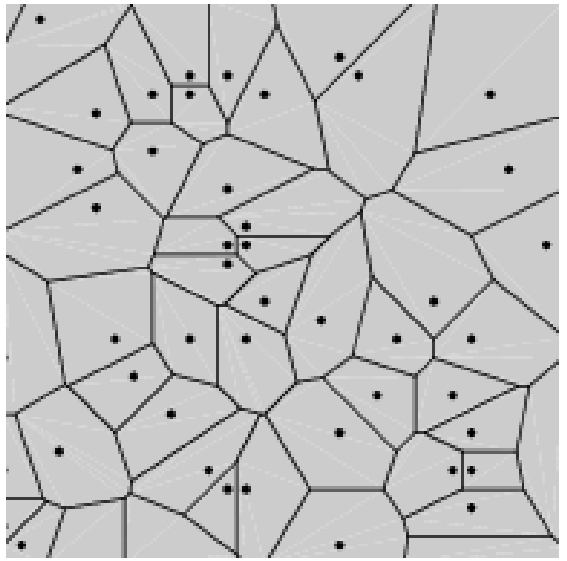
Nearest Neighbor Decision Boundary

Nearest neighbor classifier decision boundary

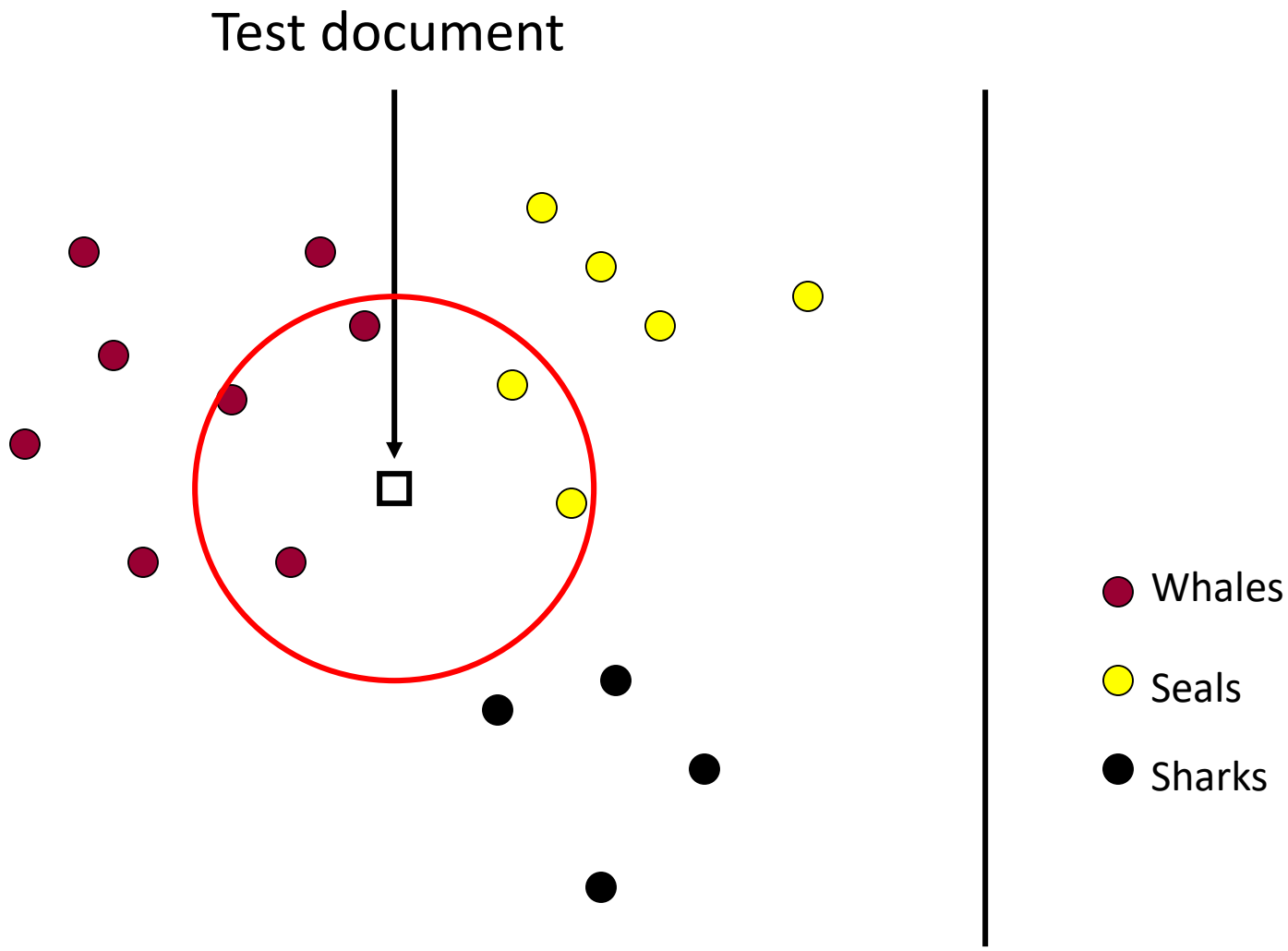


K = 1

Voronoi Diagram



k-NN classifier (k=5)



k-Nearest Neighbor Classification

Given a training dataset $\mathcal{D} = \{y^{(n)}, \boldsymbol{x}^{(n)}\}_{n=1}^{N}$, $y \in \{1, \ldots, C\}$, $\boldsymbol{x} \in \mathbb{R}^m$

and a test input $\boldsymbol{x}_{test}$, predict the class label, $\hat{y}_{test}$:

1) Find the closest k points in the training data to $\boldsymbol{x}_{test}$

$$\mathcal{N}_k(\boldsymbol{x}_{test}, \mathcal{D})$$

2) Return the class label of that closest point

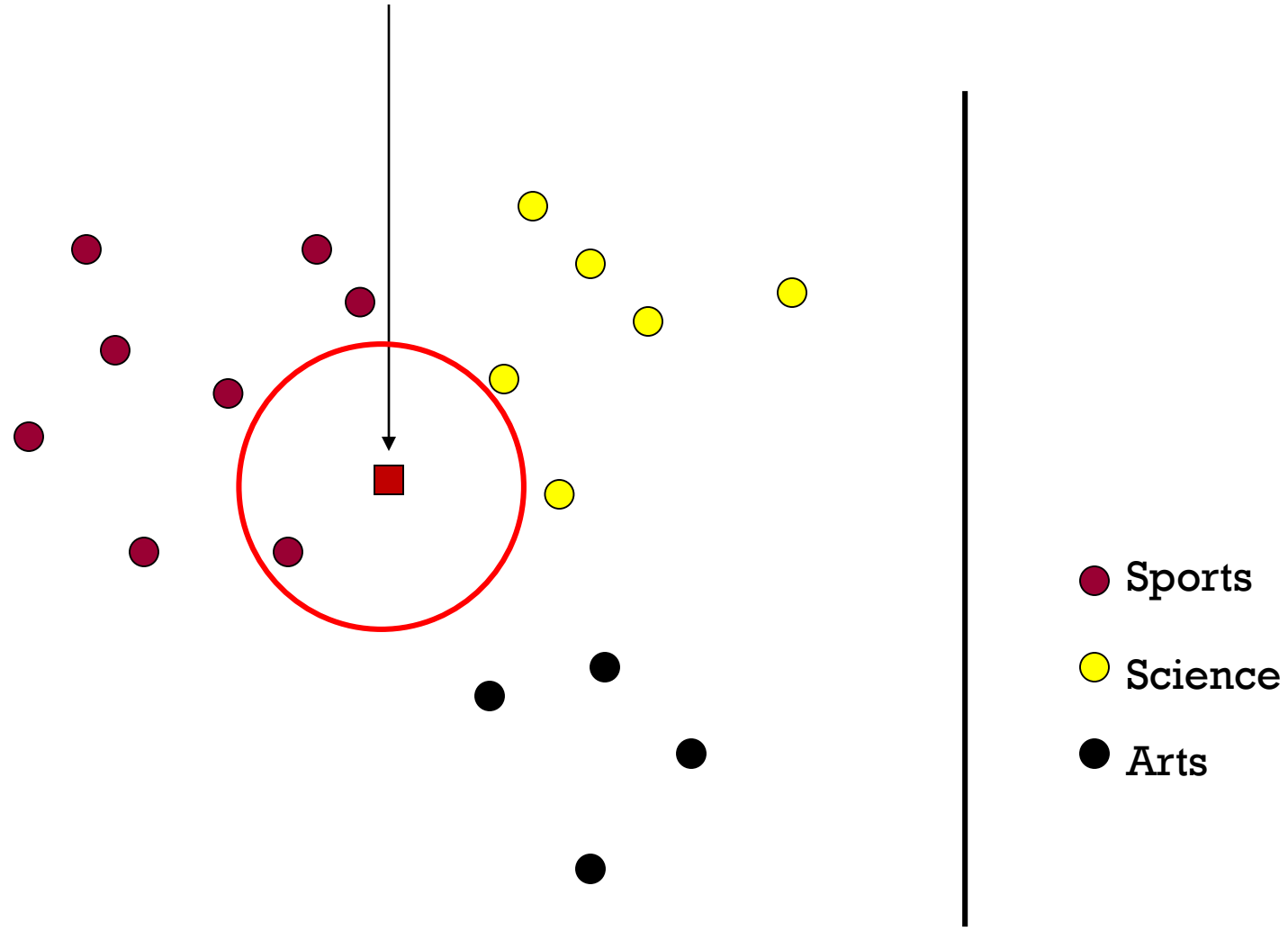
$$\hat{y}_{test} = \operatorname*{argmax}_c p(Y = c \mid \boldsymbol{x}_{test}, \mathcal{D}, k)$$

$$= \operatorname*{argmax}_c \frac{1}{k} \sum_{i \in \mathcal{N}_k(\boldsymbol{x}_{test}, \mathcal{D})} \mathbb{I}(y^{(i)} = c)$$

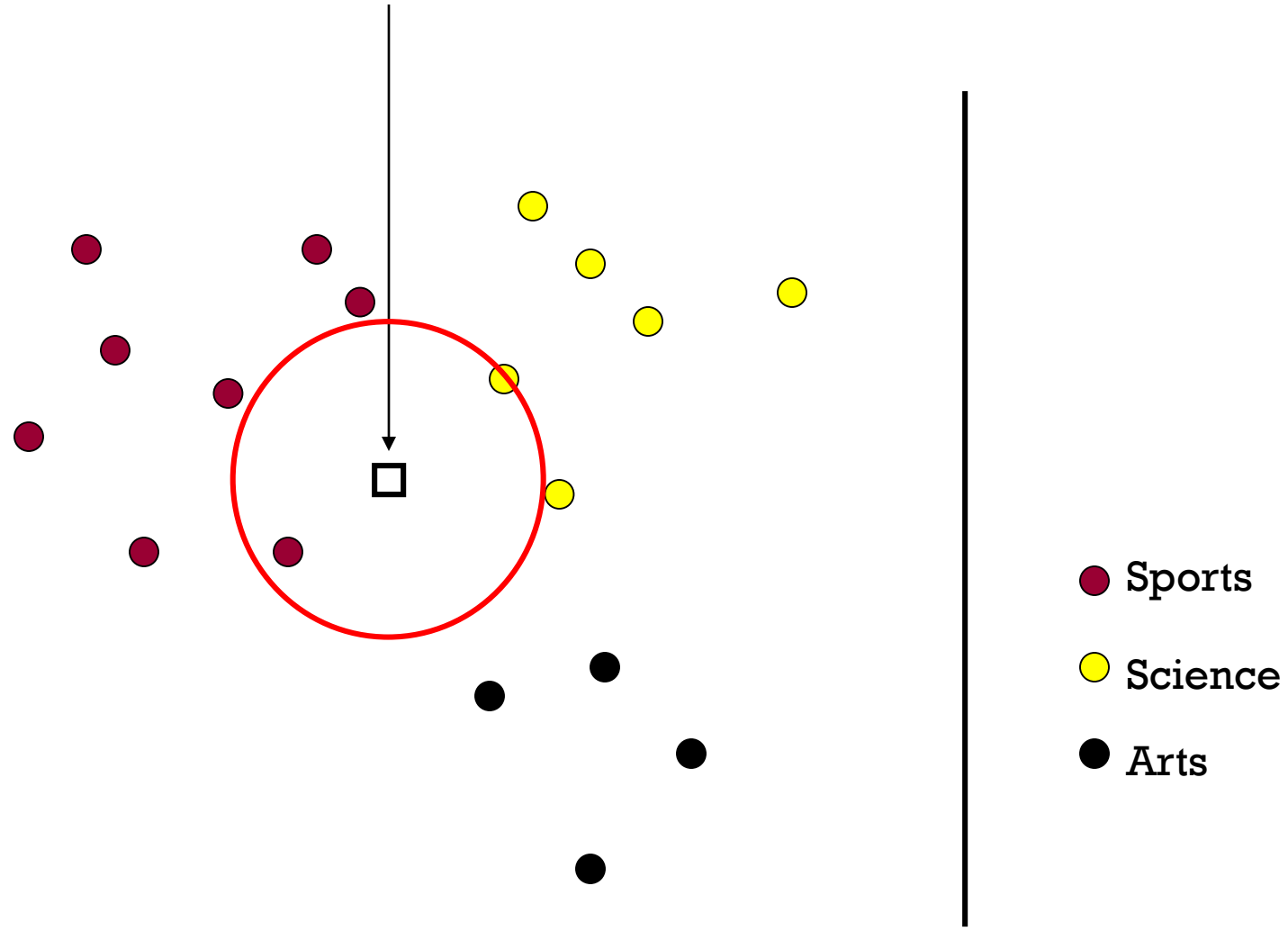
$$= \operatorname*{argmax}_c \frac{k_c}{k},$$

where k_c is the number of the k-neighbors with class label c

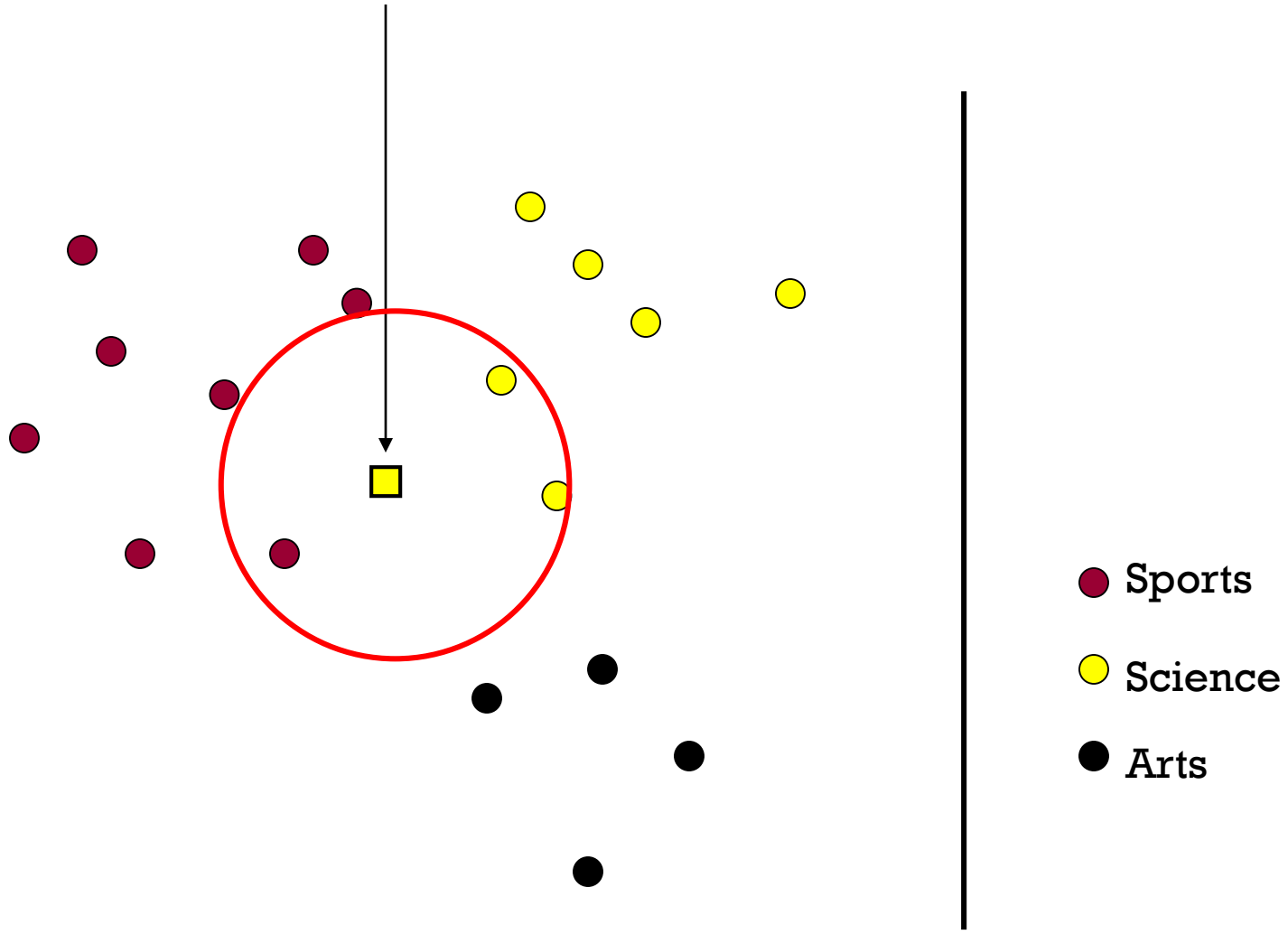
1-Nearest Neighbor (kNN) classifier



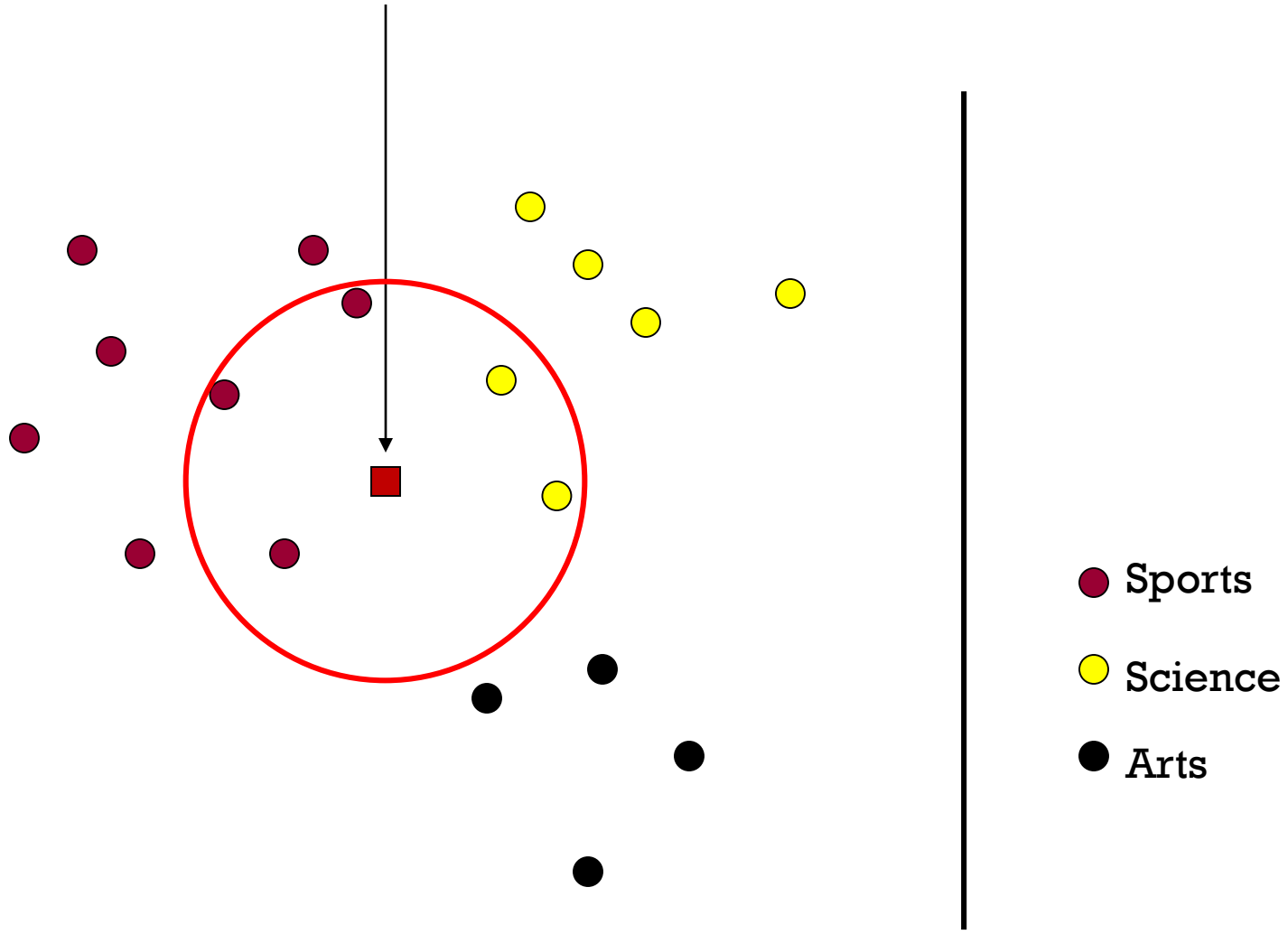
2-Nearest Neighbor (kNN) classifier



3-Nearest Neighbor (kNN) classifier



5-Nearest Neighbor (kNN) classifier



What is the best k?

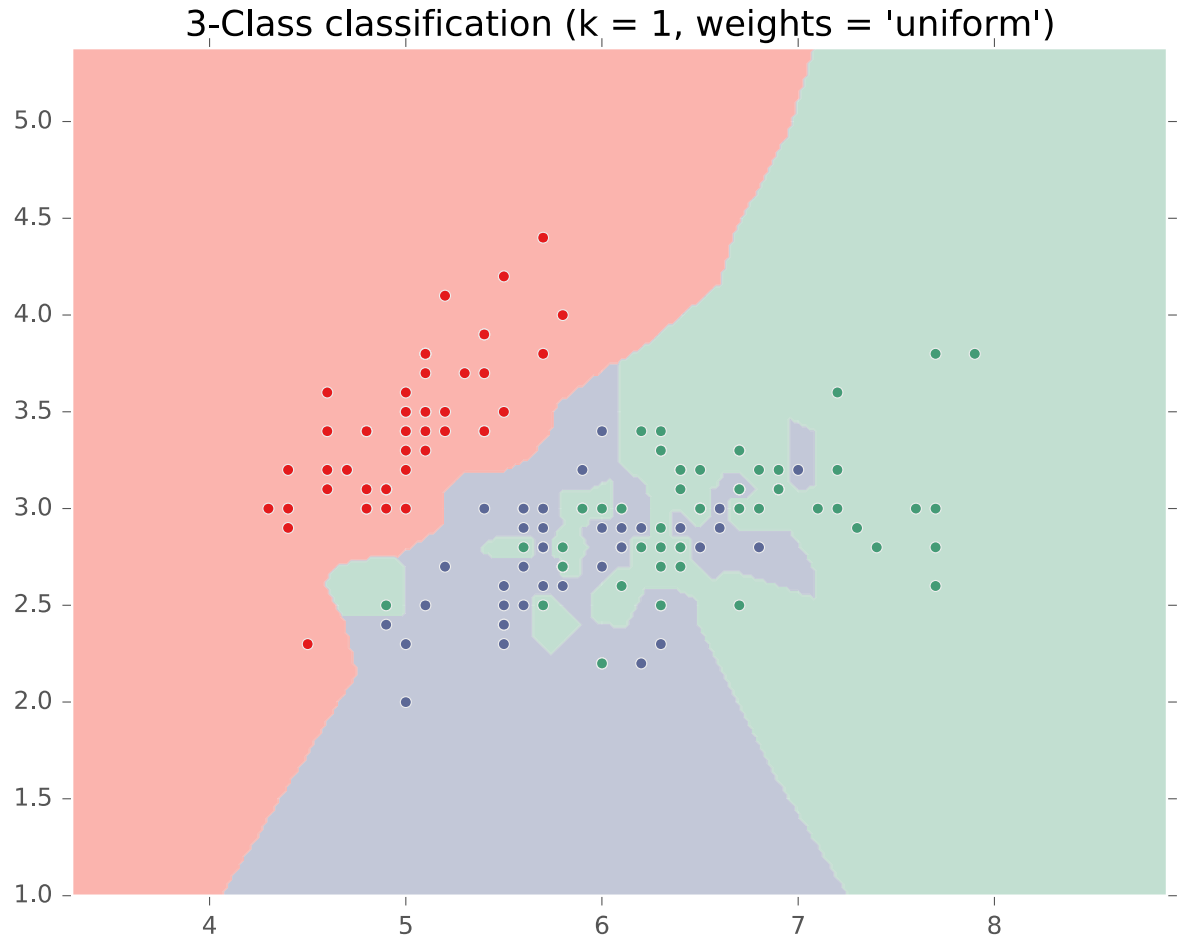
Approximation vs. Stability (aka Bias vs Variance) Tradeoff

- Larger k → predicted label is more stable
- Smaller k → predicted label is more affected by individual training points

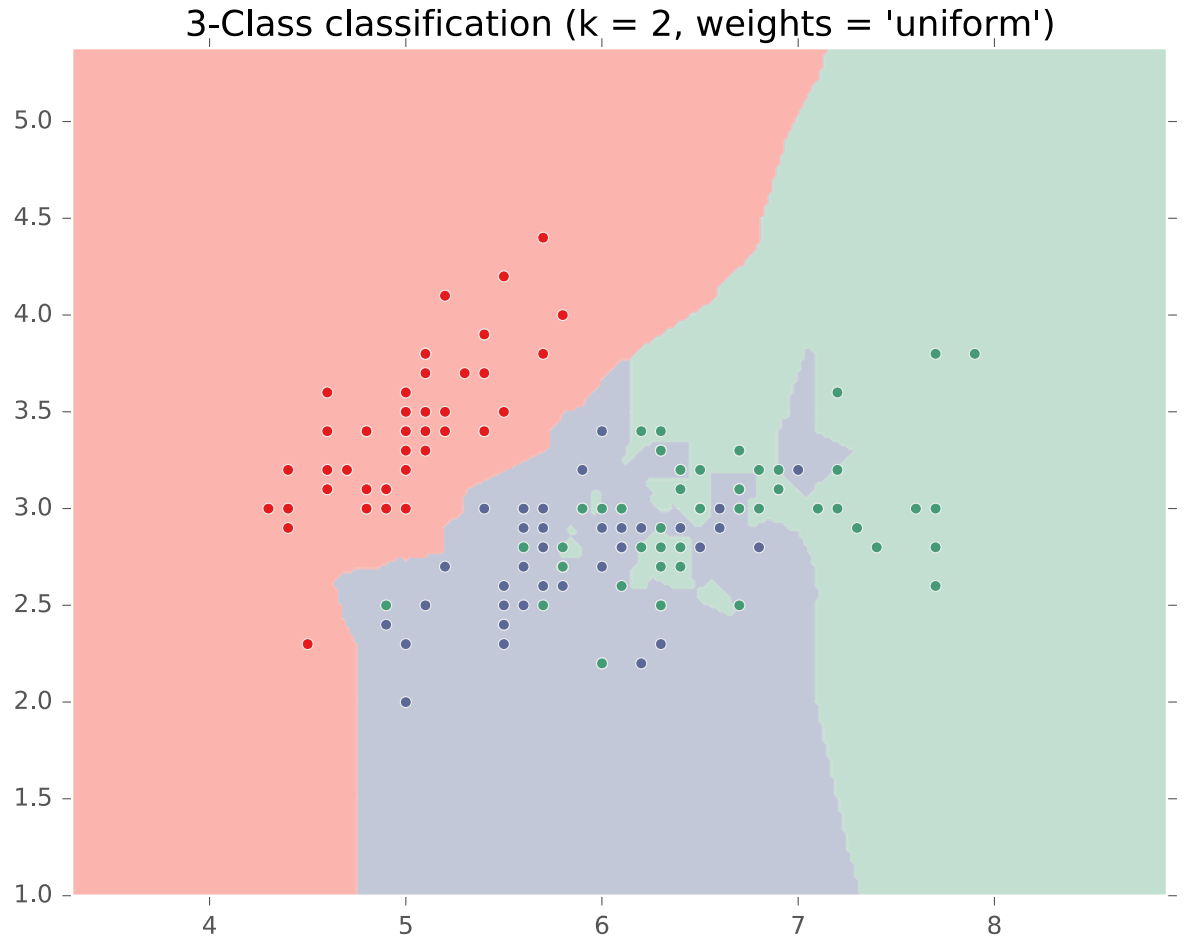
How to choose hyperparameter k?

k-NN on Fisher Iris Data

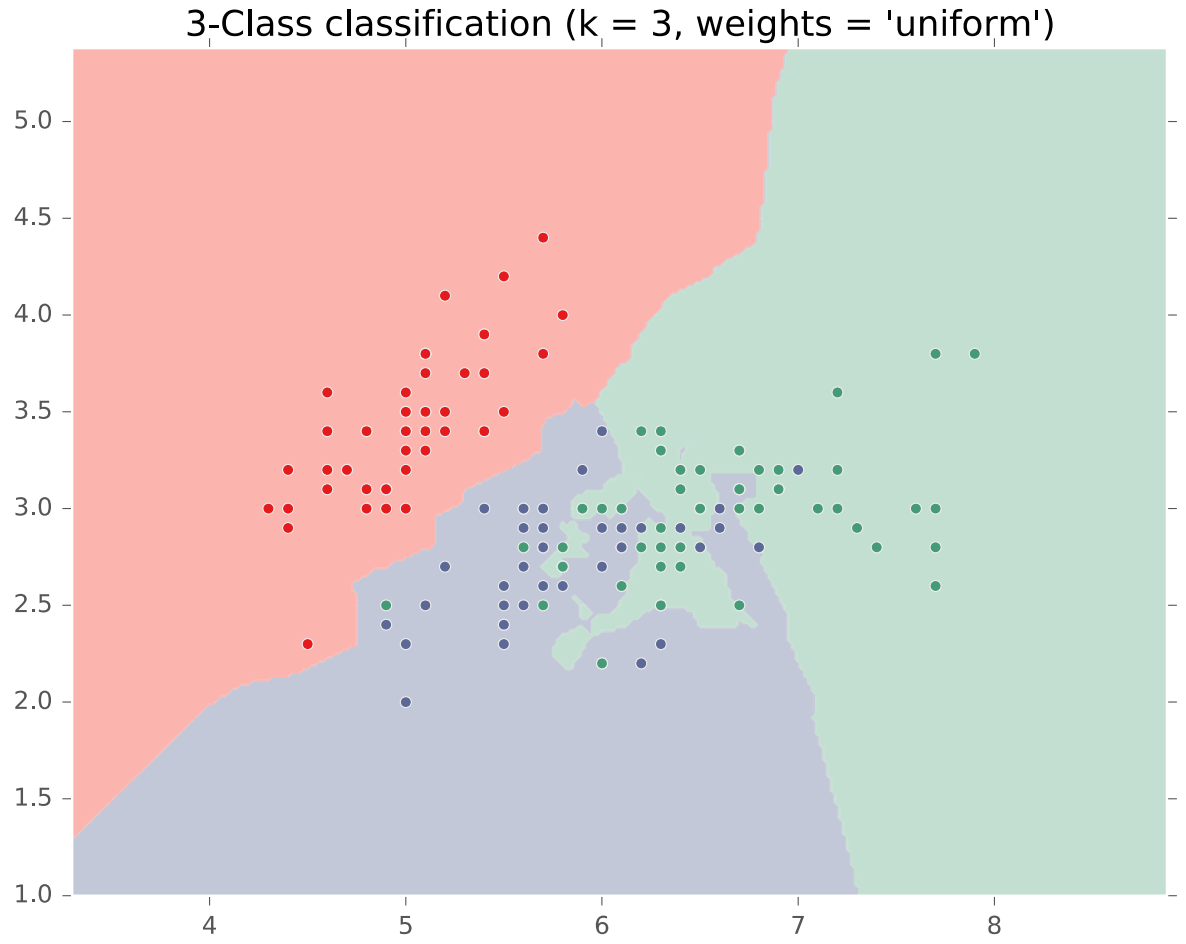
Special Case: Nearest Neighbor



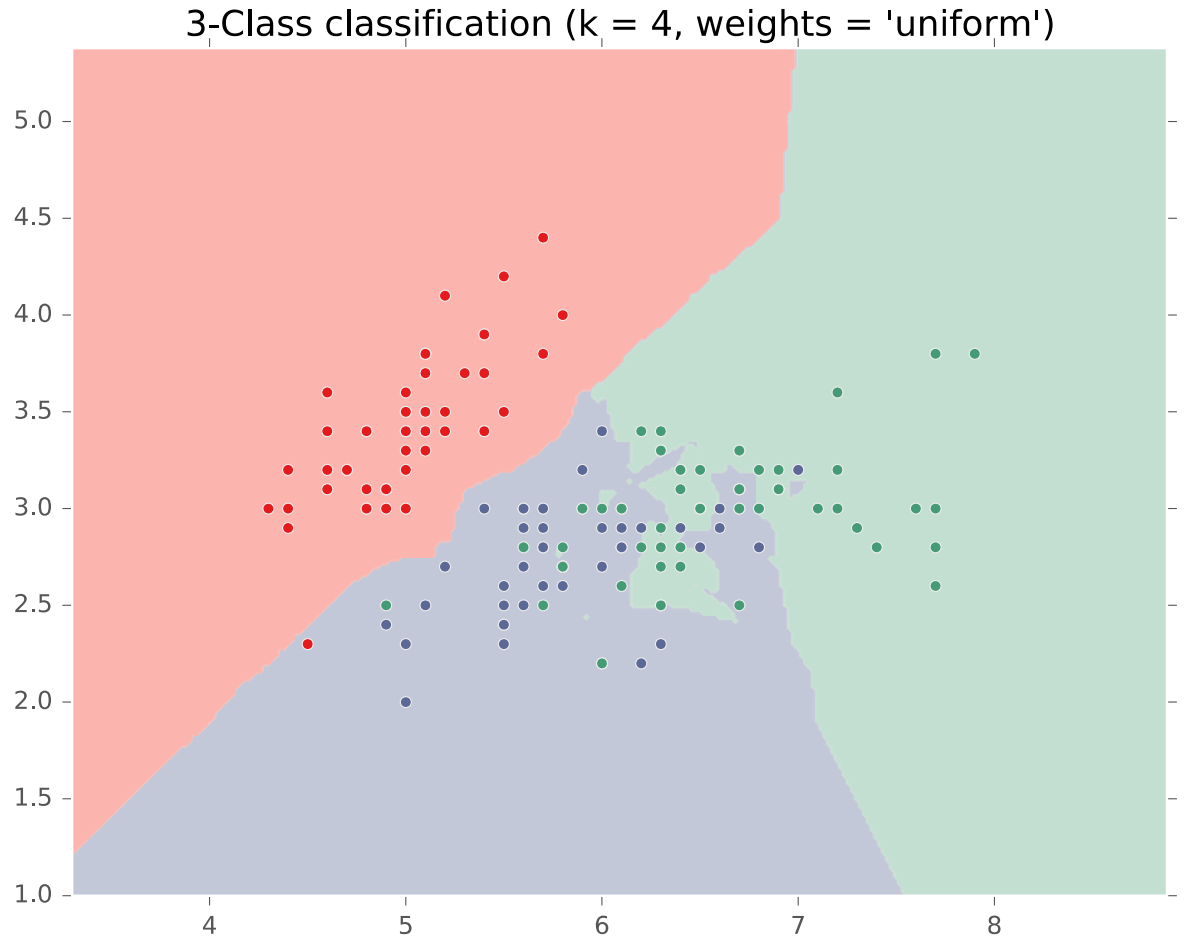
k-NN on Fisher Iris Data



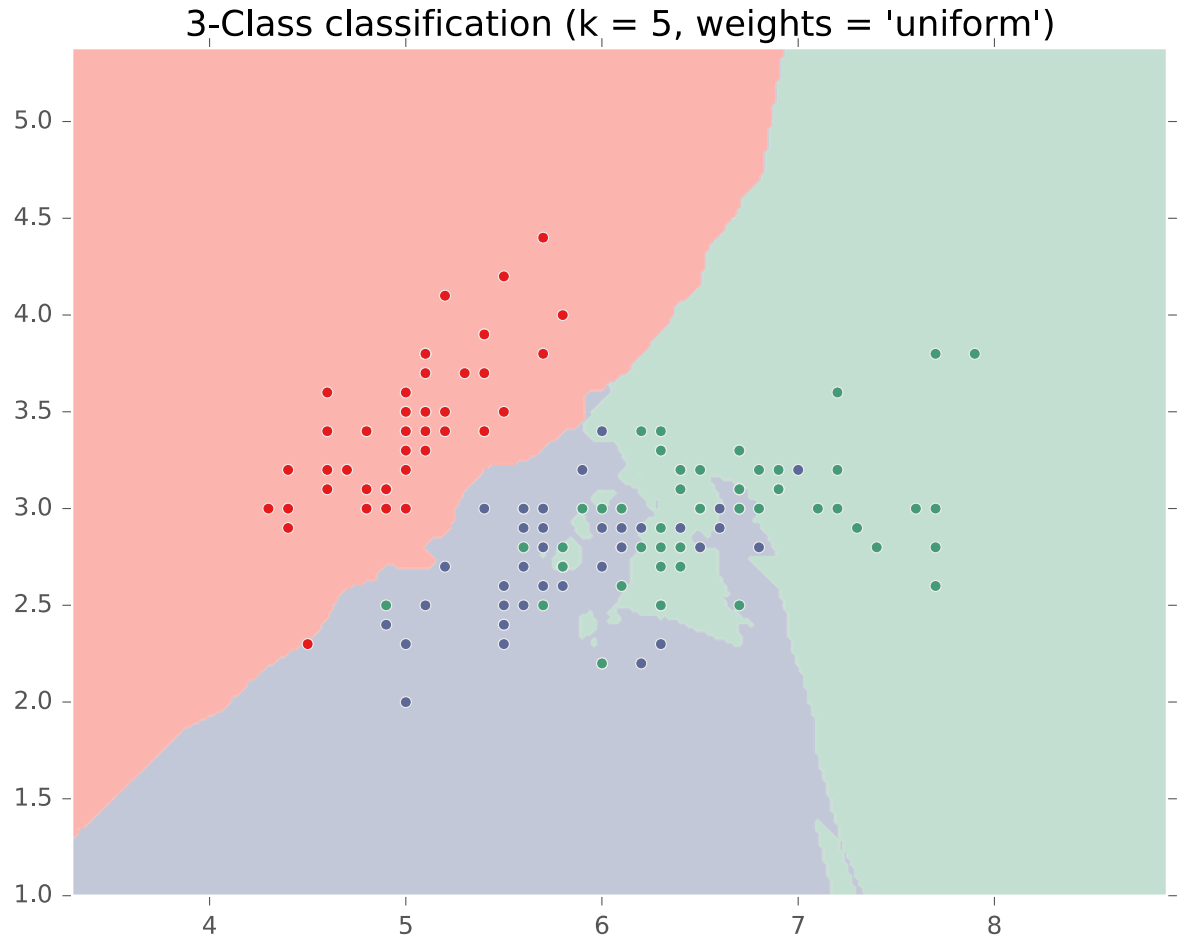
k-NN on Fisher Iris Data



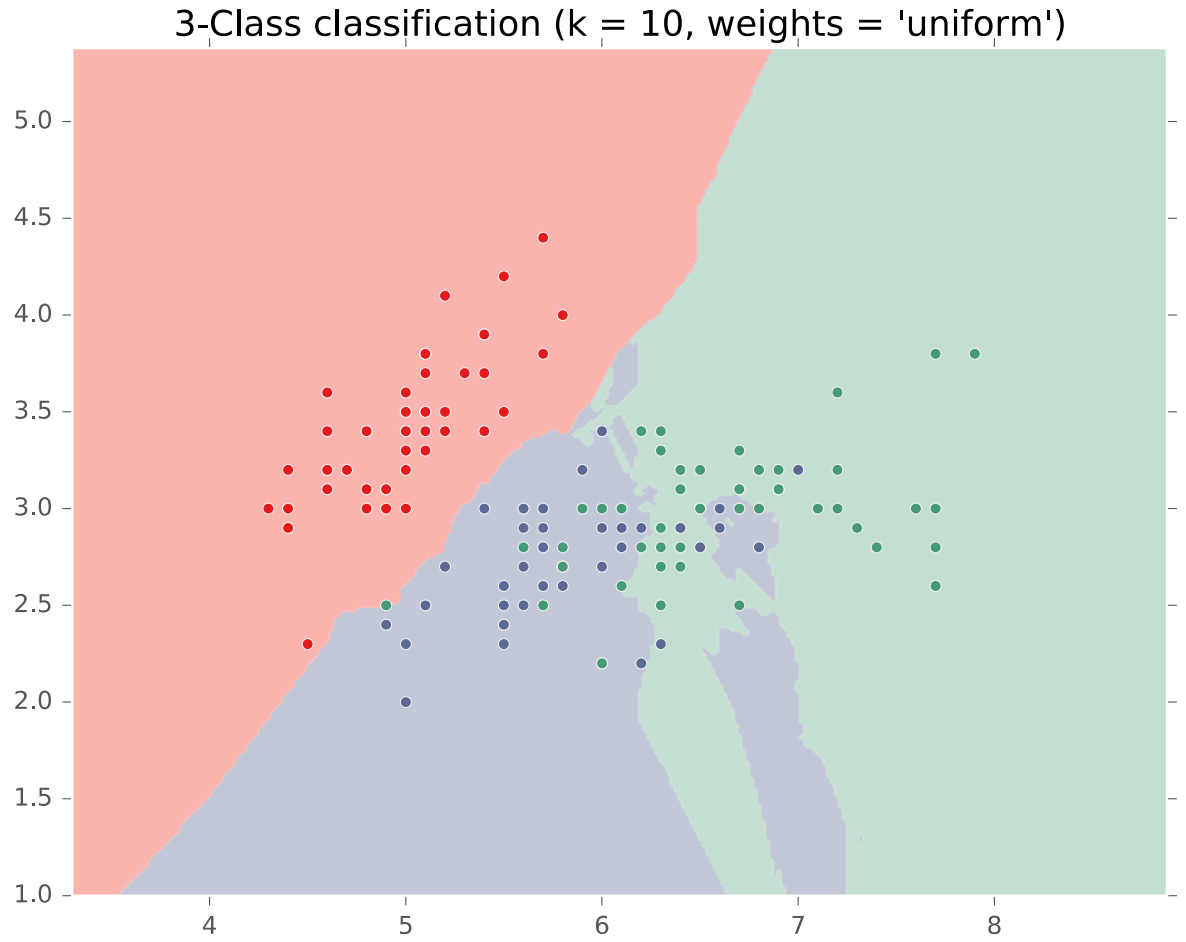
k-NN on Fisher Iris Data



k-NN on Fisher Iris Data

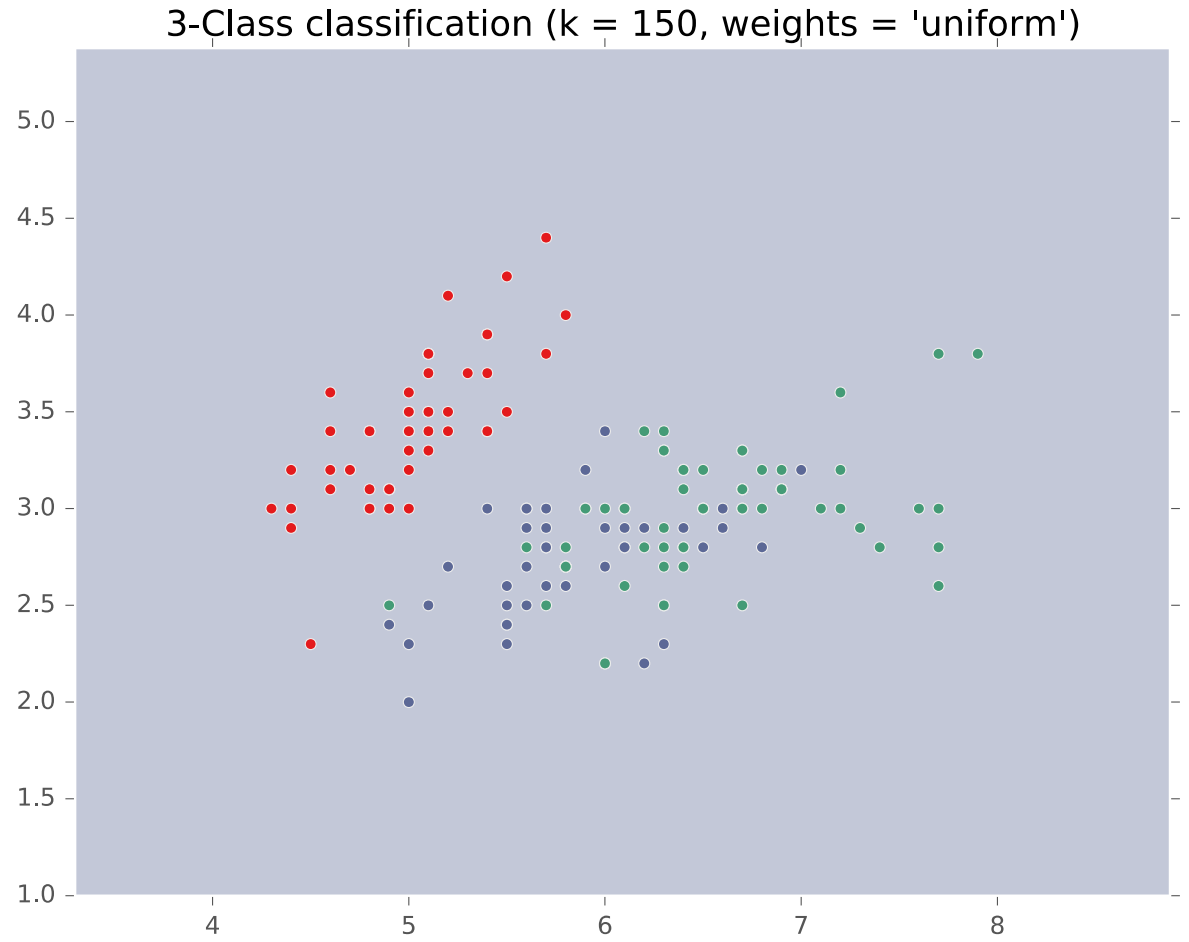


k-NN on Fisher Iris Data



k-NN on Fisher Iris Data

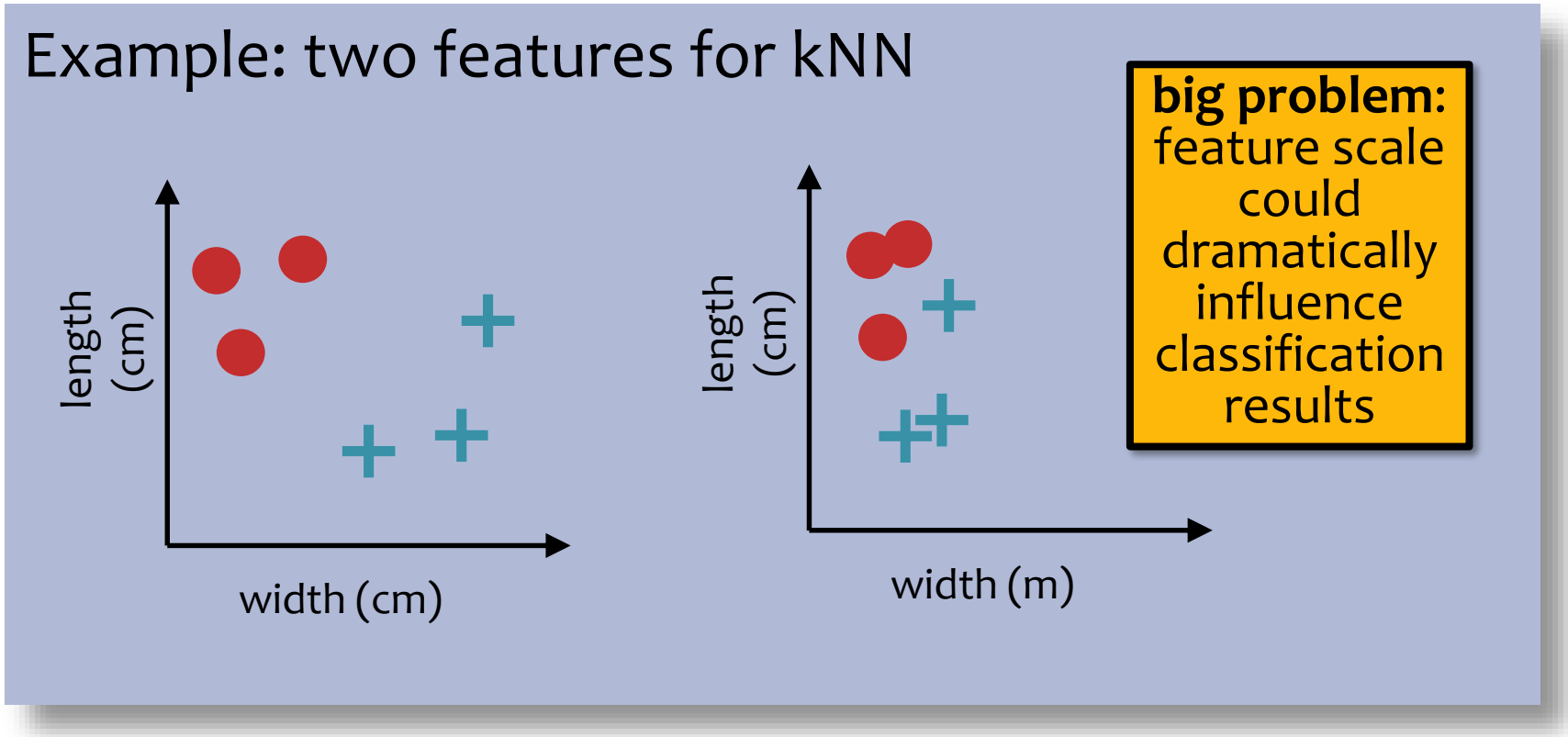
Special Case: Majority Vote



Nearest Neighbor

Assumptions!

- Similar points have similar neighbors
- All feature dimensions are created equally!



Nearest Neighbor and Deep Learning

Use neural networks to learn to transform input data into a feature space where distance is more meaningful

- Example: Face recognition