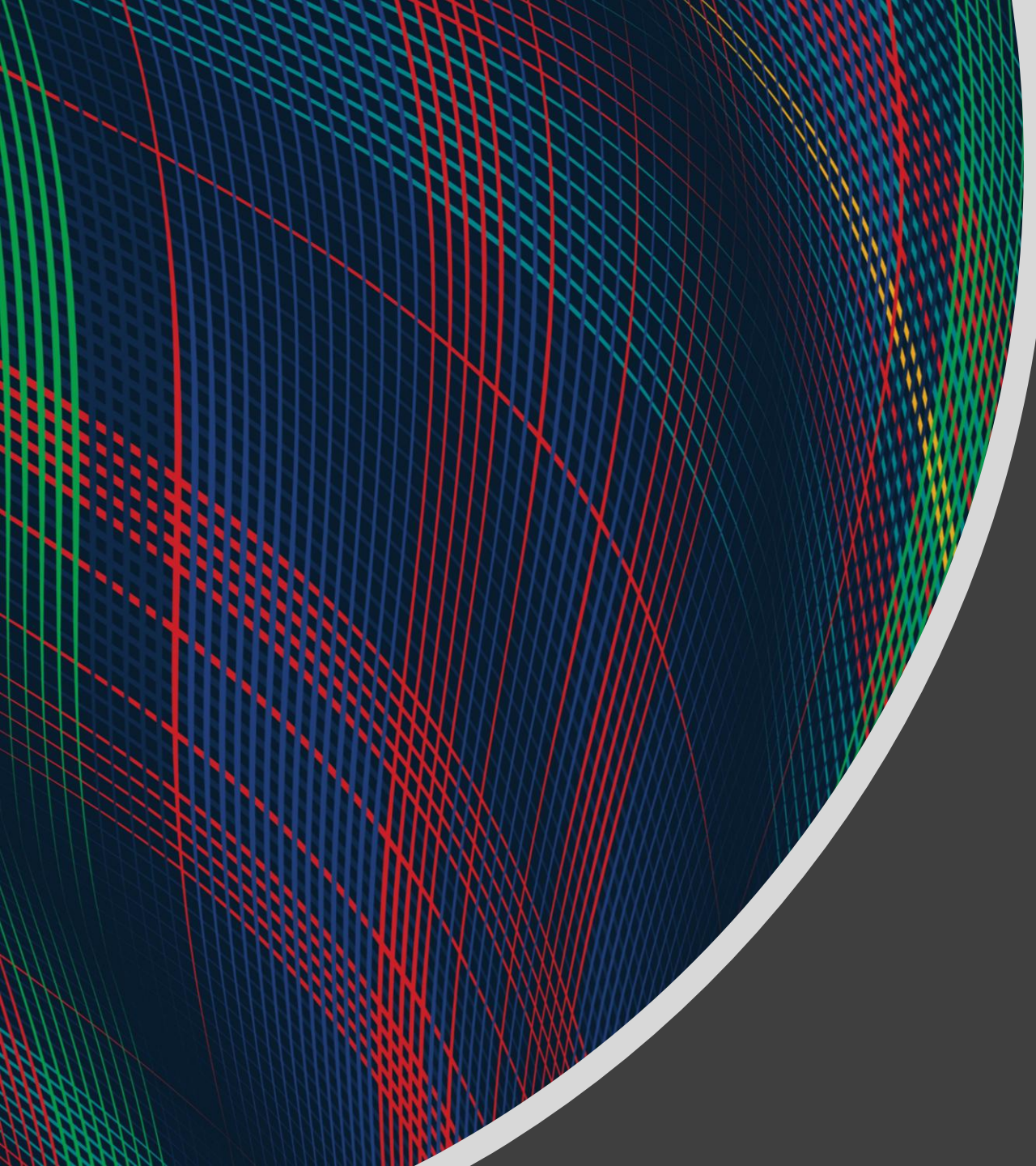


An abstract graphic on the left side of the slide, featuring a sphere-like shape composed of a dense grid of intersecting red, green, and blue lines. The lines are curved and follow the contours of the sphere, creating a complex, woven pattern. The sphere is set against a dark gray background.

10-315

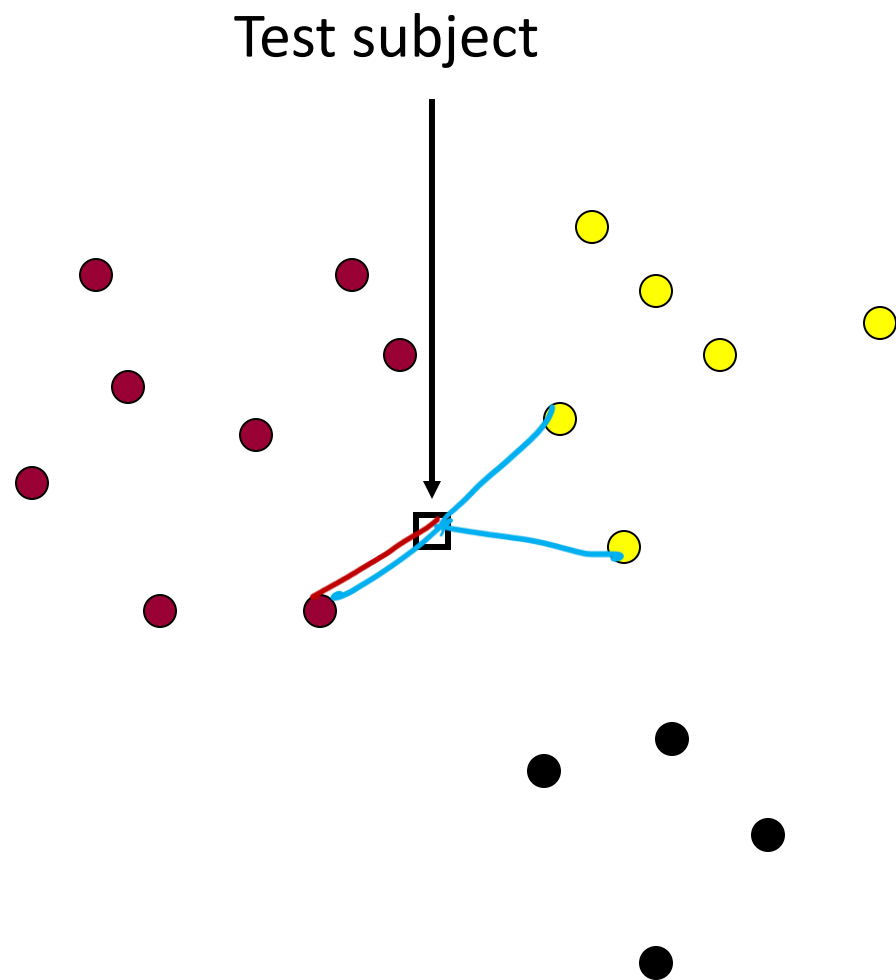
Introduction to ML

K-Nearest Neighbor

Instructor: Pat Virtue

Nearest Neighbor Classifier

$$K=1$$
$$K=3$$



- Whales
- Seals
- Sharks

$$\frac{1}{3}$$
$$\frac{2}{3}$$
$$\frac{0}{3}$$

Nearest Neighbor Classification

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

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

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

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

2) Return the class label of that closest point

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

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

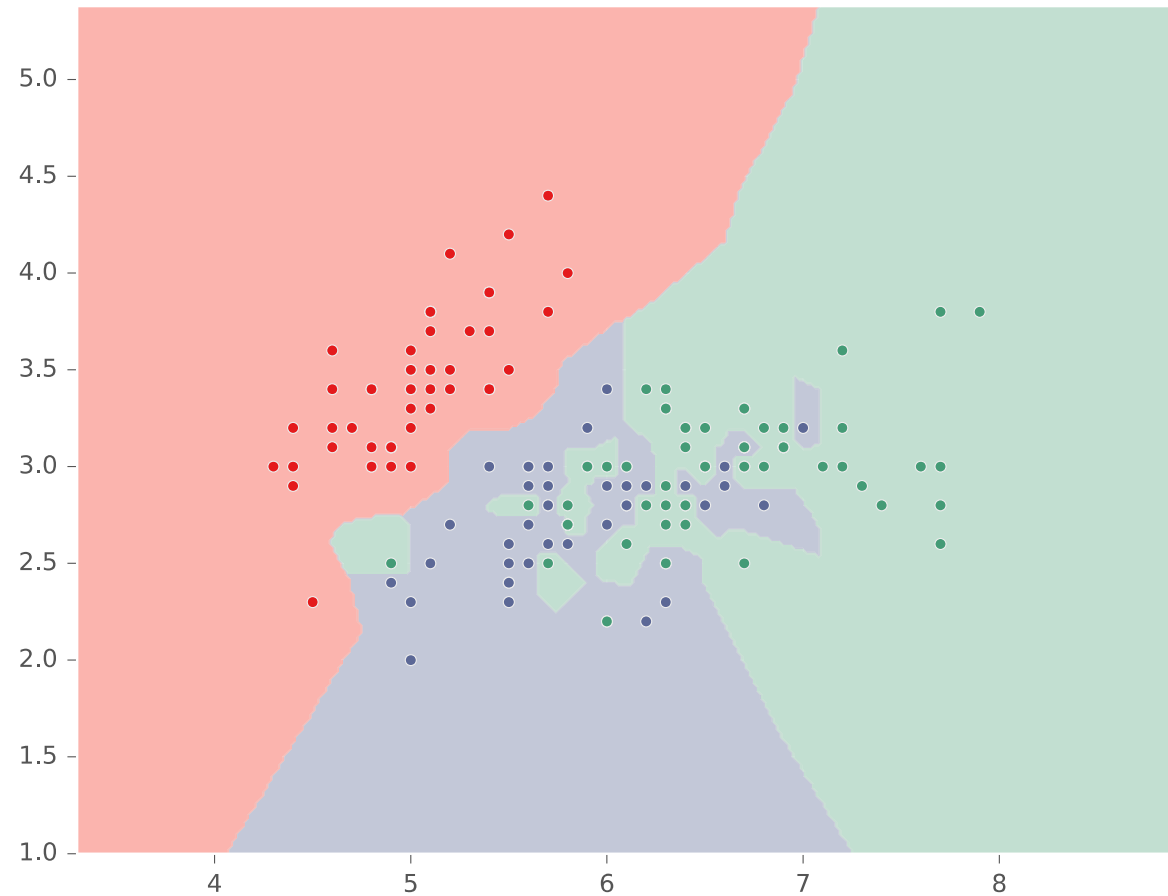
Deleted two of the four features, so that input space is 2D



Nearest Neighbor on Fisher Iris Data



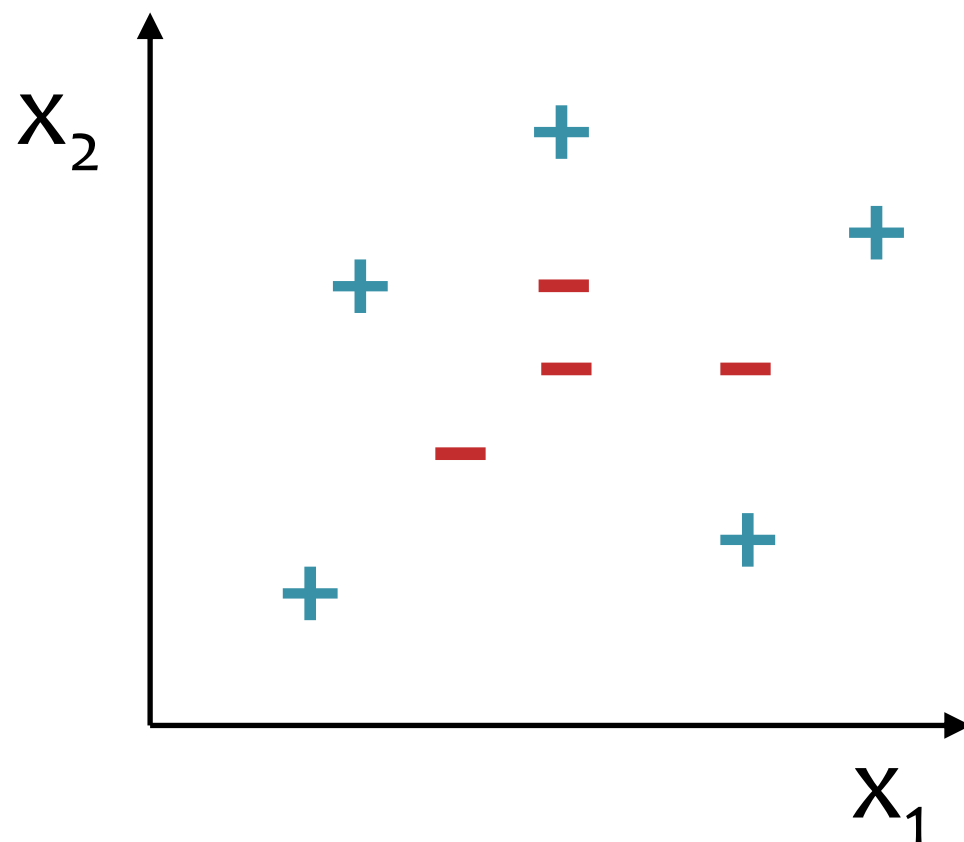
Nearest Neighbor on Fisher Iris Data



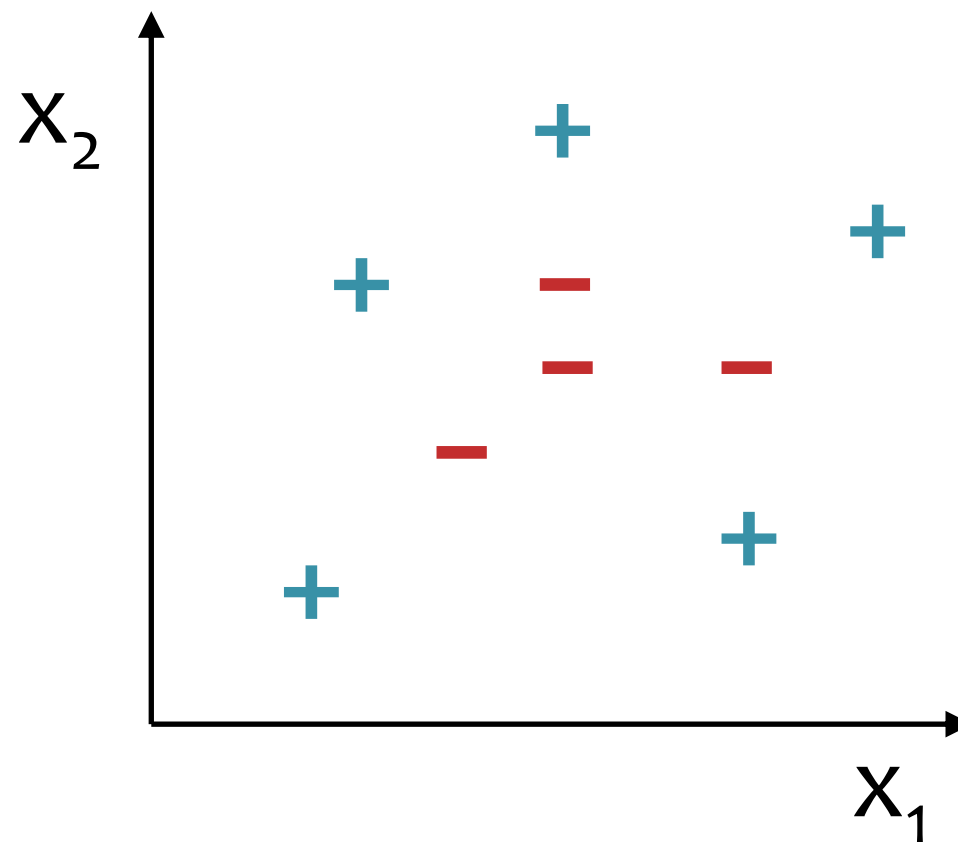
Recitation: Decision Boundaries

$K=1$

Decision tree

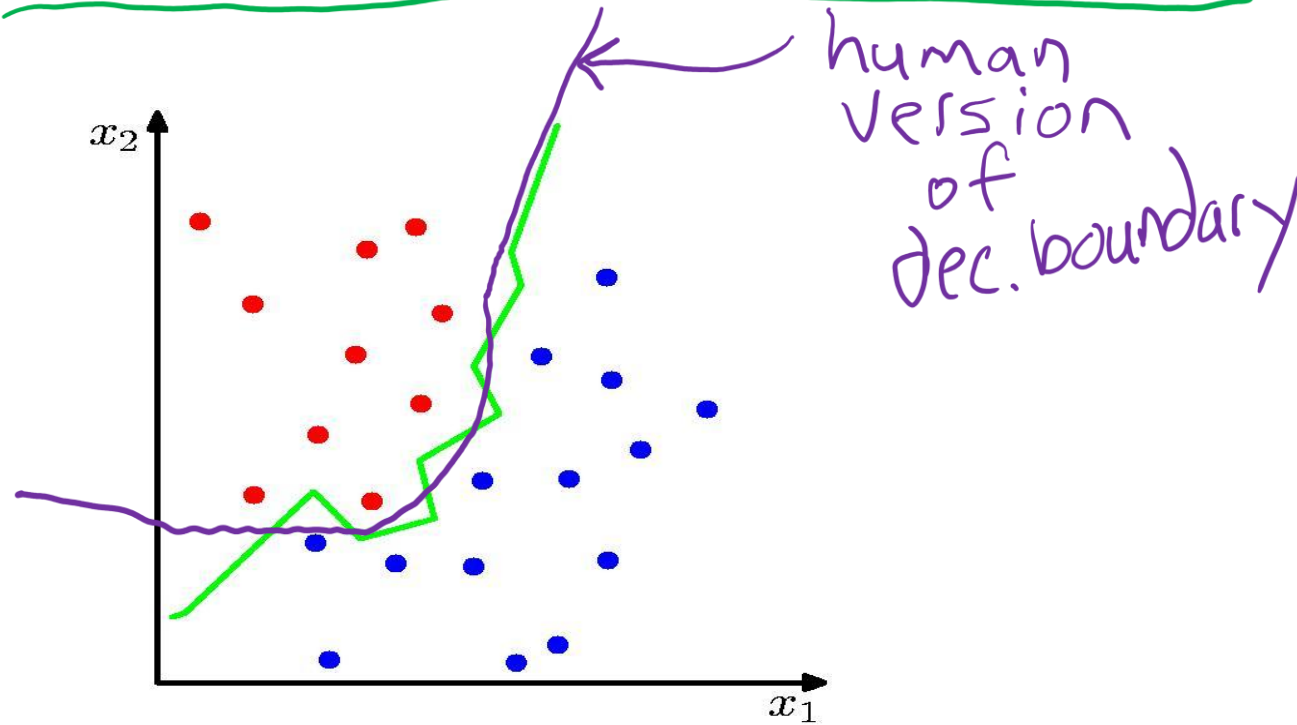


Nearest neighbor

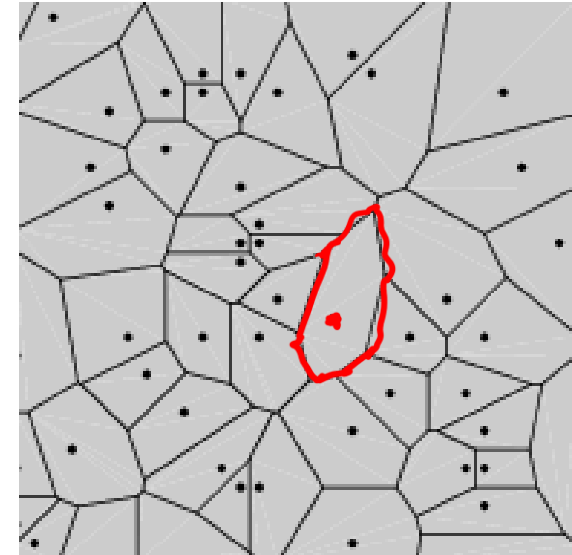


Nearest Neighbor Decision Boundary

1-nearest neighbor classifier decision boundary

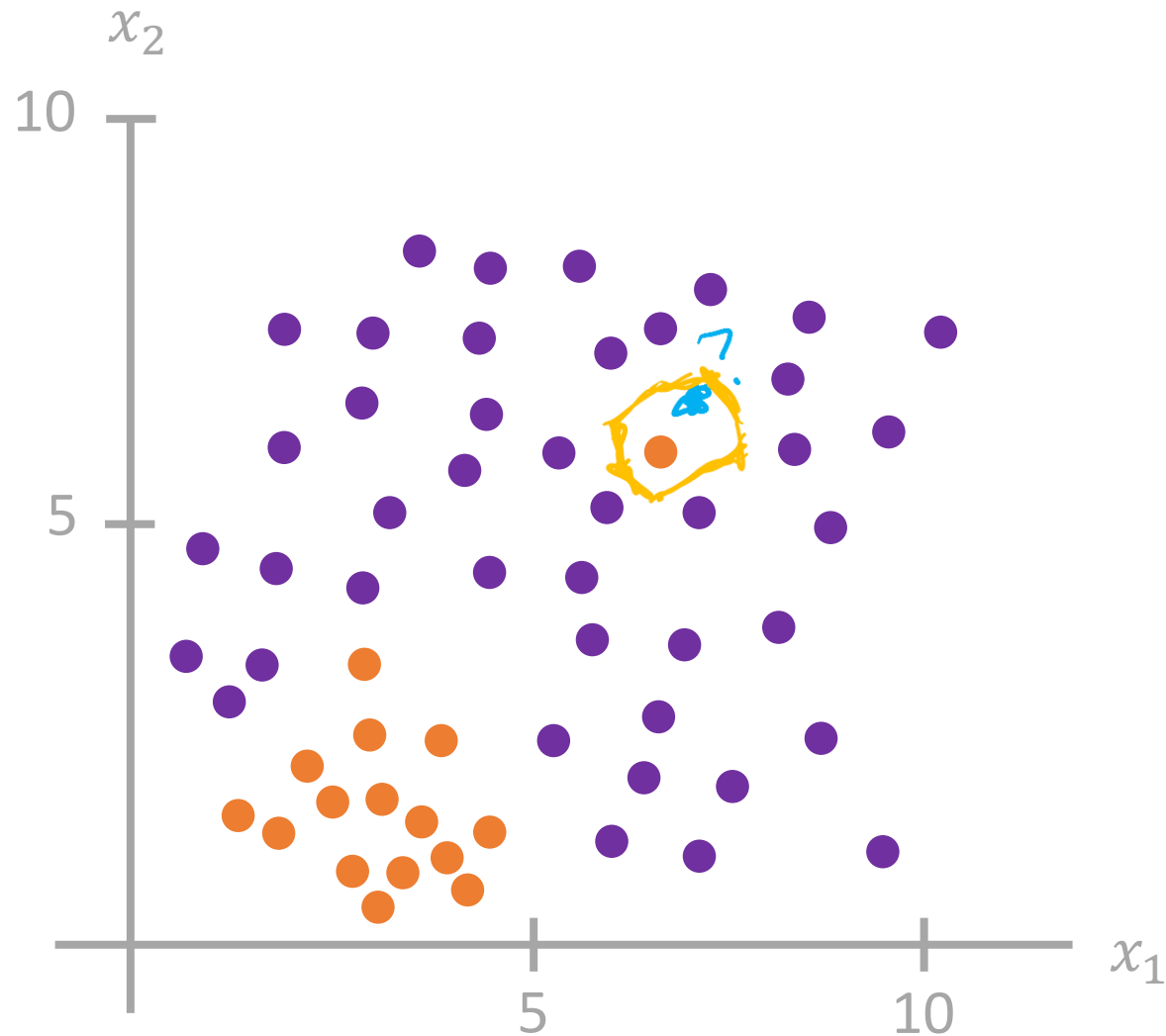


Voronoi Diagram

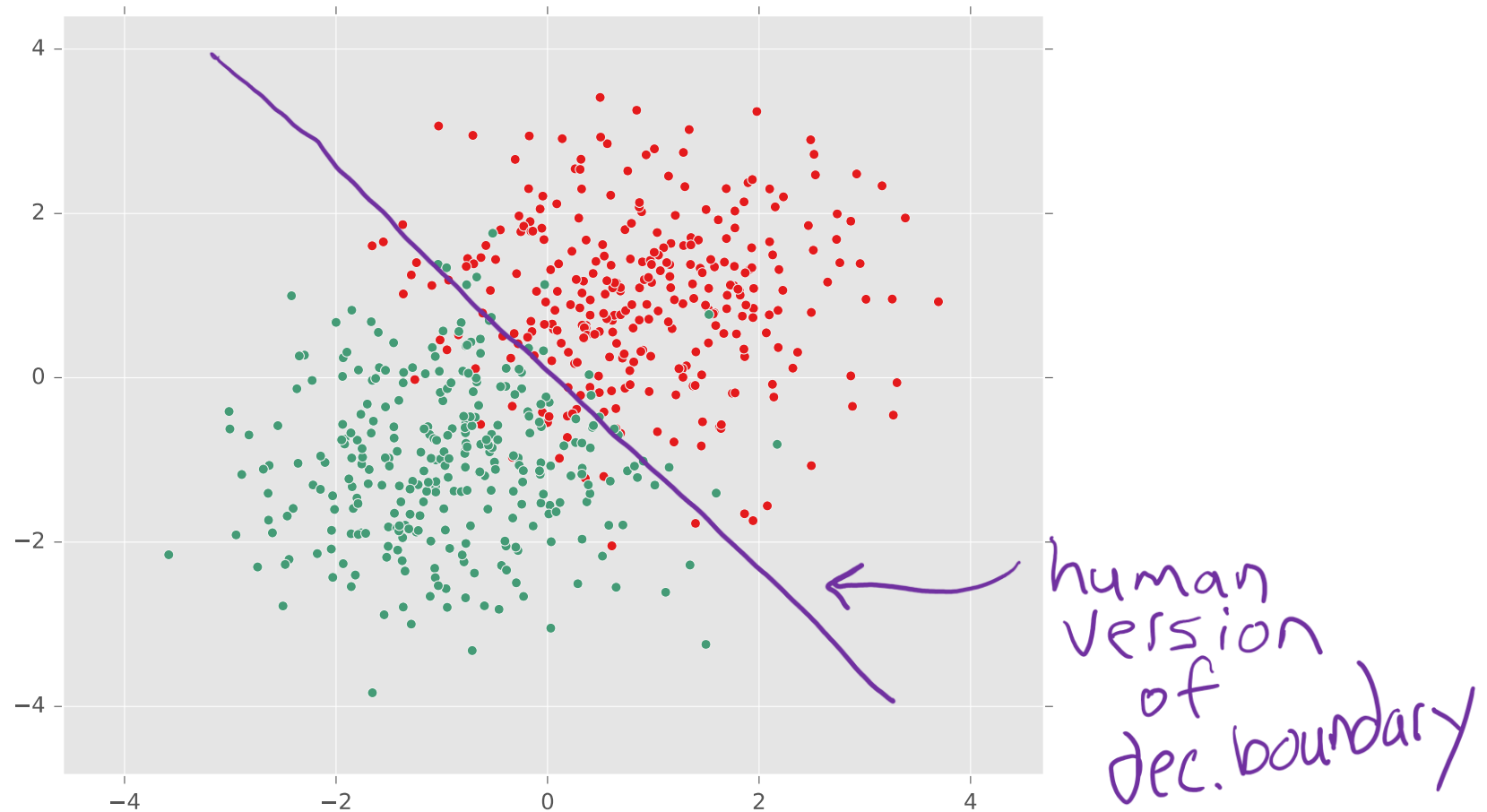


Nearest Neighbor Classification

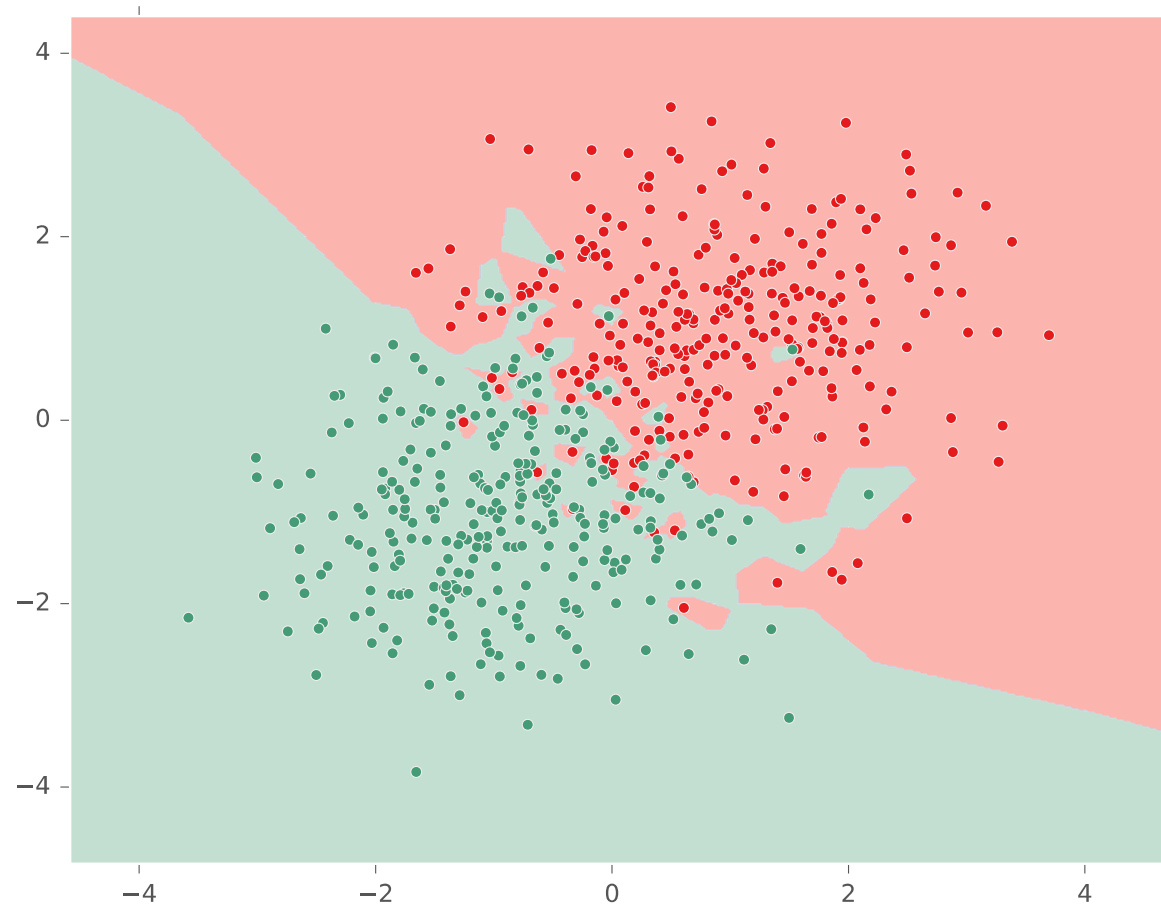
Consider input features $x \in \mathbb{R}^2$.



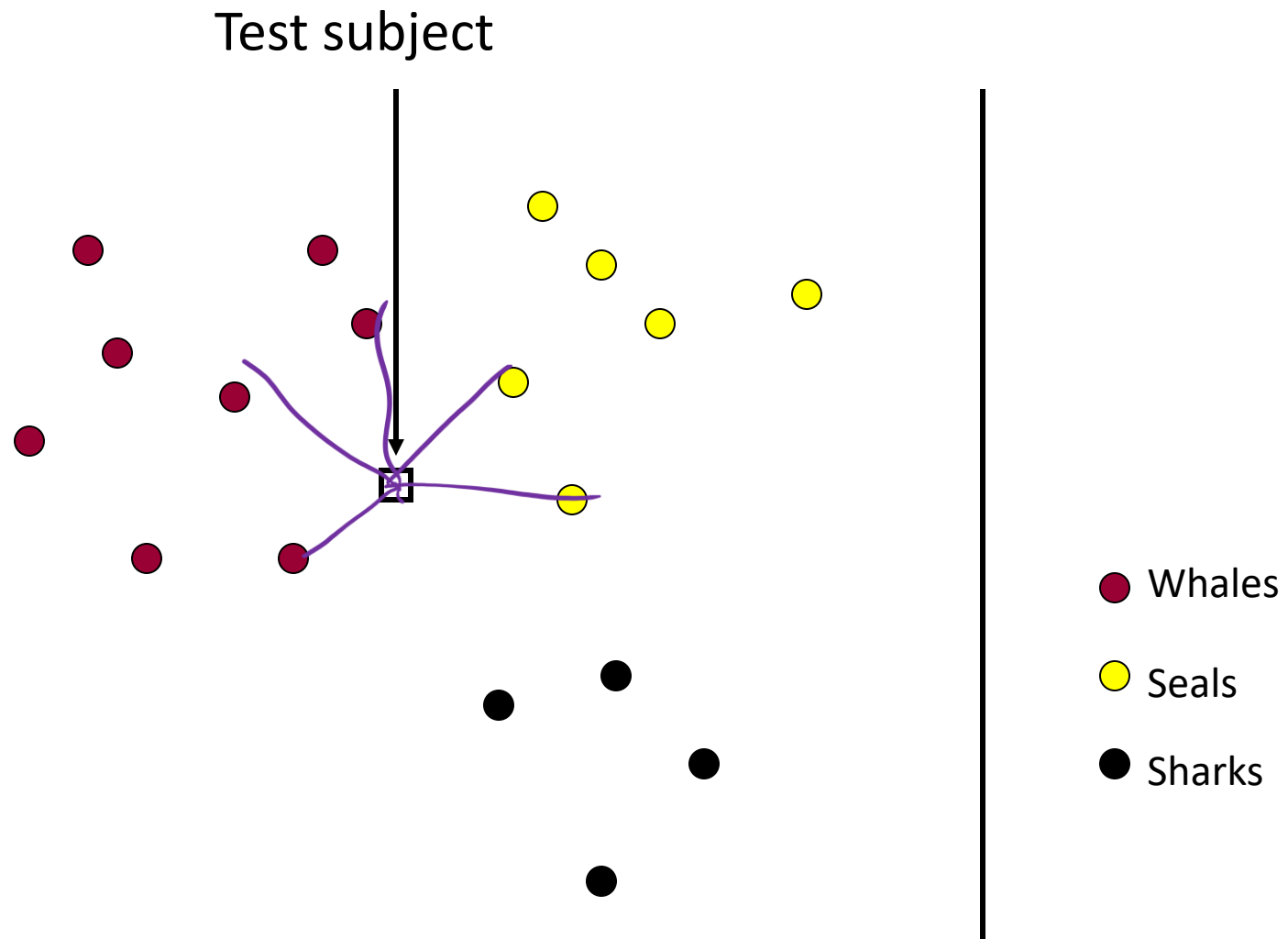
Nearest Neighbor on Gaussian Data



Nearest Neighbor on Gaussian Data



kNN classifier (k=5)



Nearest Neighbor Classification

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

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

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

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

2) Return the class label of that closest point

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

k-Nearest Neighbor Classification

Given a training dataset $\mathcal{D} = \{y^{(n)}, \mathbf{x}^{(n)}\}_{n=1}^N$, $y \in \{1, \dots, C\}$, $\mathbf{x} \in \mathbb{R}^M$ and a test input $\mathbf{x}_{test}$, predict the class label, $\hat{y}_{test}$:

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

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

- 2) Return the class label of that closest point

$$\begin{aligned}\hat{y}_{test} &= \operatorname{argmax}_c p(Y = c \mid \mathbf{x}_{test}, \mathcal{D}, k) \\ &= \operatorname{argmax}_c \frac{1}{k} \sum_{i \in \mathcal{N}_k(\mathbf{x}_{test}, \mathcal{D})} \mathbb{I}(y^{(i)} = c) \\ &= \operatorname{argmax}_c \frac{k_c}{k},\end{aligned}$$

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

What is the best k ?

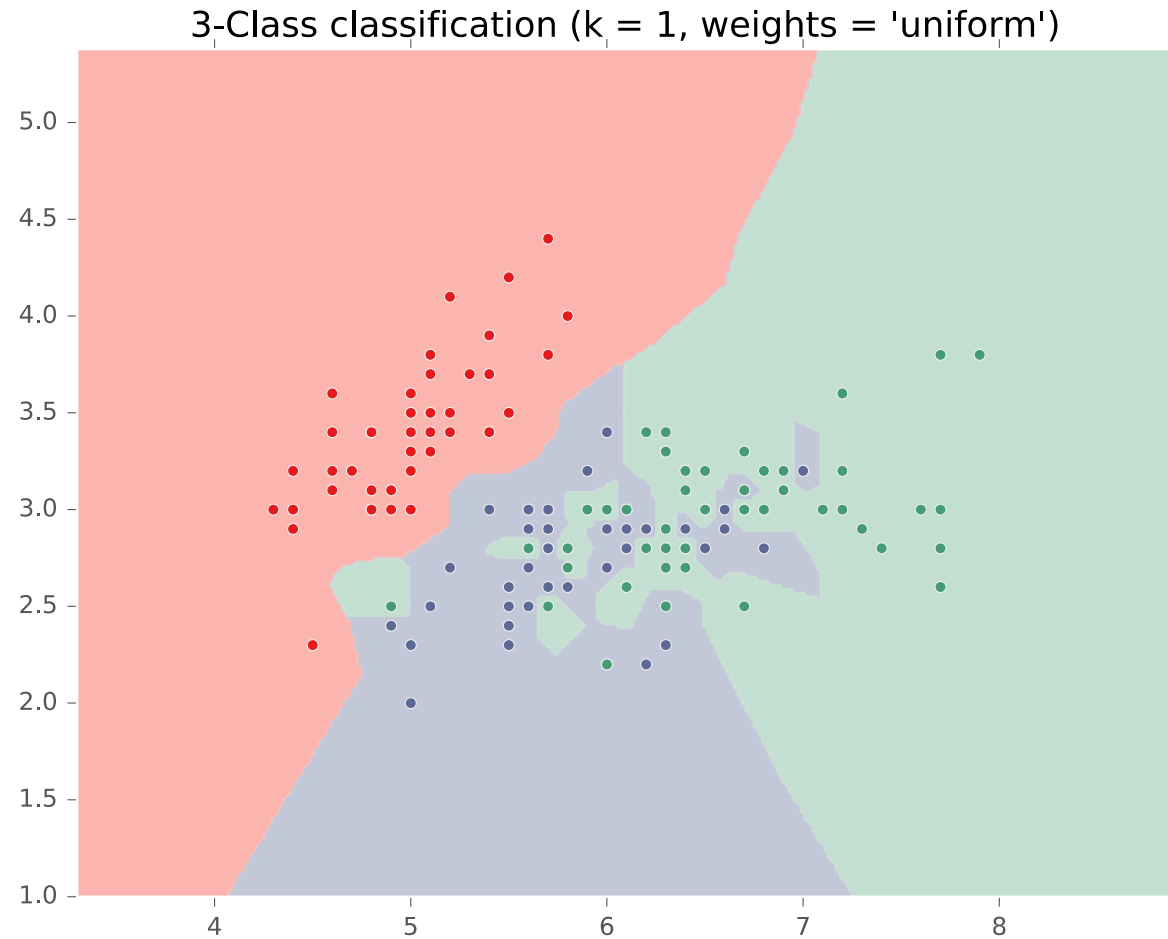
How do we choose a learner that is accurate and also generalizes to unseen data?

- Larger $k \rightarrow$ predicted label is more stable
- Smaller $k \rightarrow$ predicted label is more affected by individual training points

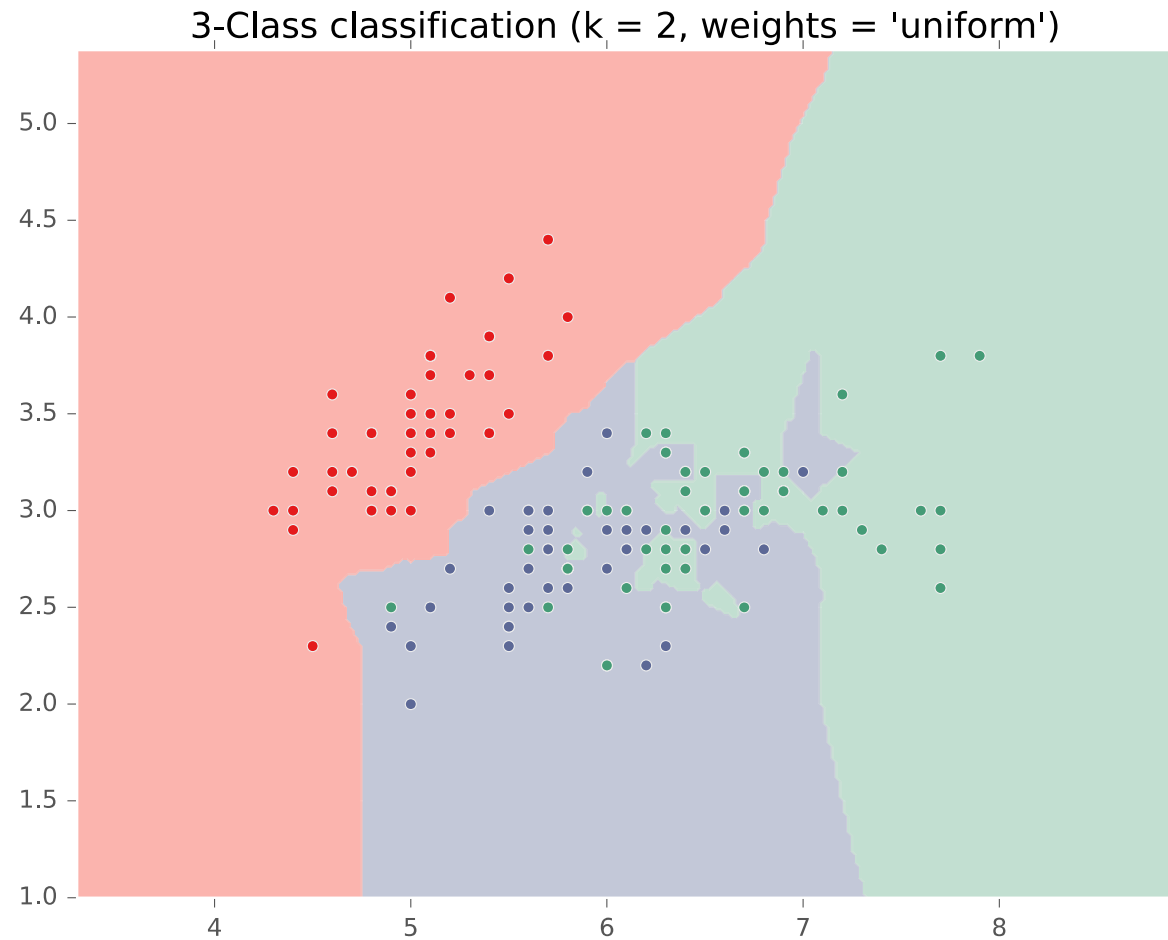
But how to choose k ?

k-NN on Fisher Iris Data

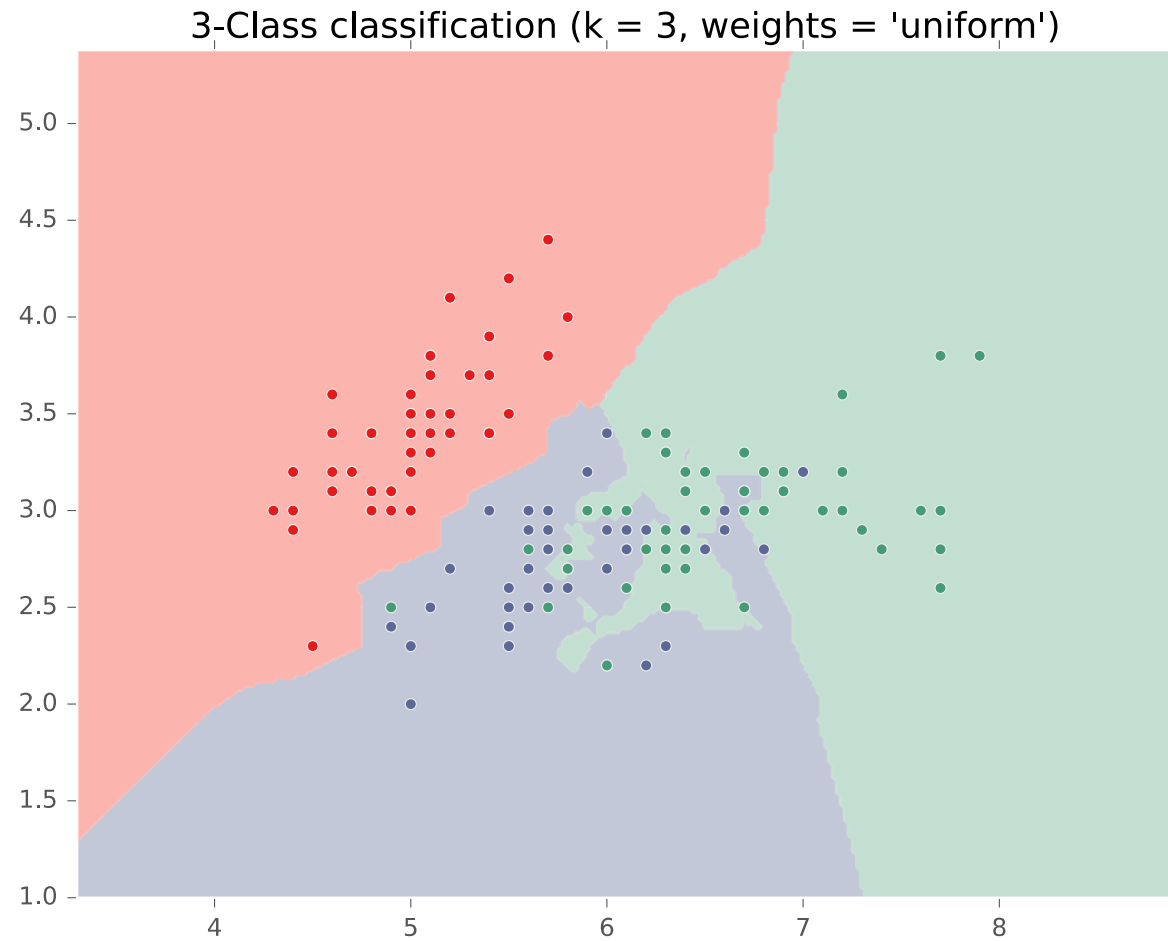
Special Case: Nearest Neighbor



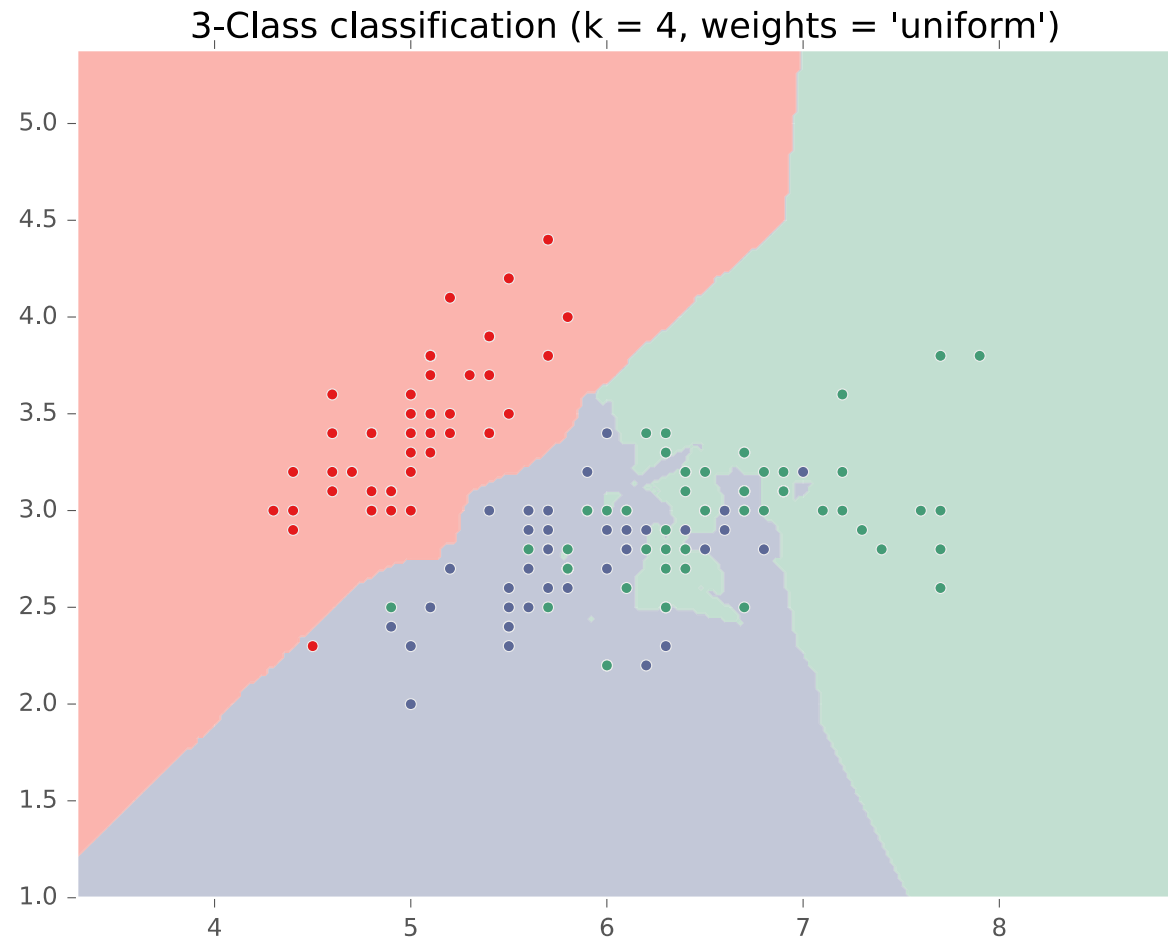
k-NN on Fisher Iris Data



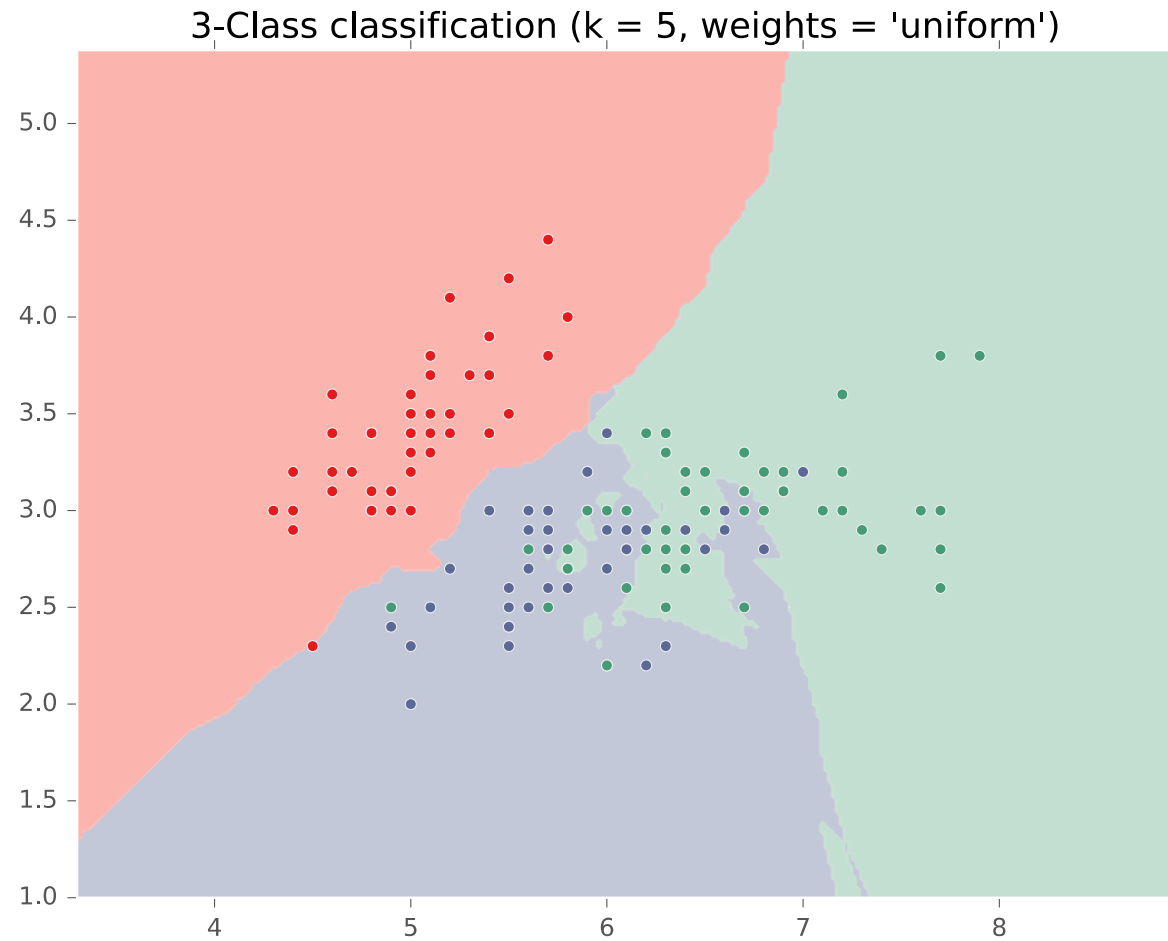
k-NN on Fisher Iris Data



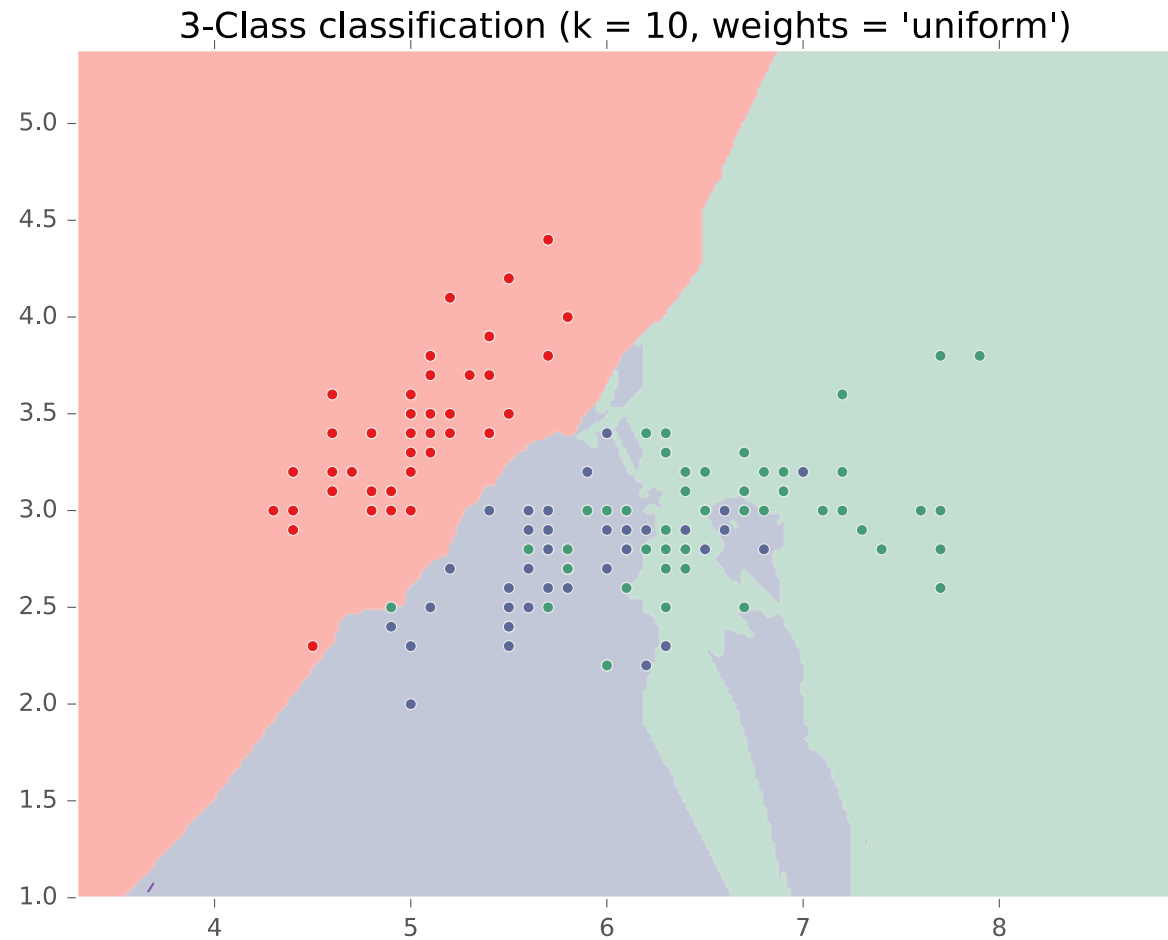
k-NN on Fisher Iris Data



k-NN on Fisher Iris Data

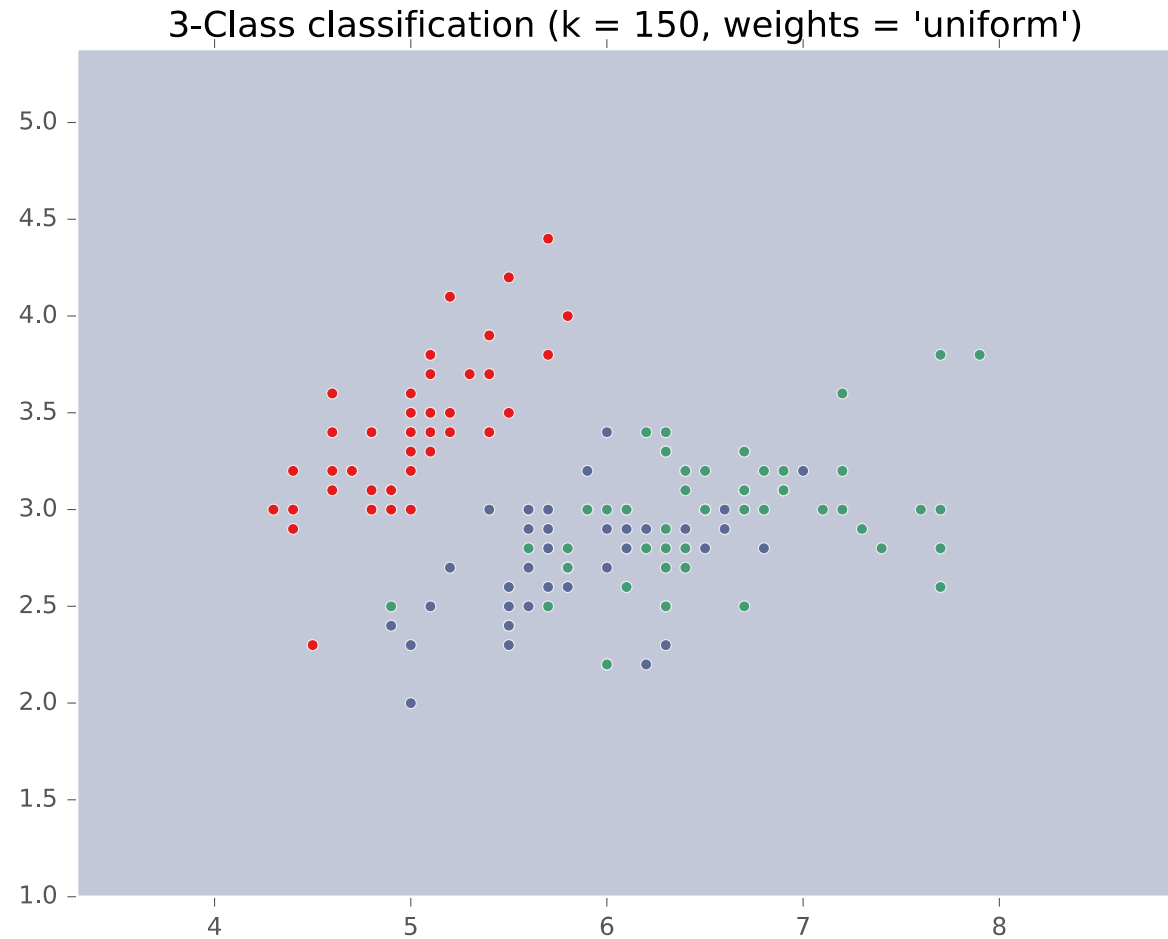


k-NN on Fisher Iris Data



k-NN on Fisher Iris Data

Special Case: Majority Vote



K-NN Details

Poll 1: Two questions (train) and (predict)

Suppose we have N training examples, and each one has M features
Computational complexity for the special case where $k=1$:

- A. $O(1)$
- B. $O(\log N)$
- C. $O(\log M)$
- D. $O(\log NM)$
- E. $O(N)$
- F. $O(M)$
- G. $O(NM)$ 40
- H. $O(N^2)$
- I. $O(N^2M)$

def train($\mathcal{D}$):
 self.data = $\mathcal{D}$
 -- or --
 self.data =
 deepcopy($\mathcal{D}$)

def predict($\vec{x}_{\text{new}}$):
 for i in range(N):
 ...
 dist($\vec{x}_{\text{new}}, \vec{x}^{(i)}$)
 ...

k-NN: Details

Computational Efficiency:

Suppose we have N training examples, and each one has M features

Computational complexity for the special case where $k=1$:

Task	Naive
Train	$O(1)$
Predict (one test example)	$O(MN)$



In practice:

- k-d trees
- stochastic approximations (very fast, and empirically often as good)

k-NN: Details

The **inductive bias** of a machine learning algorithm is the principal by which it generalizes to unseen examples

Inductive Bias:

1. Close points should have similar labels
2. All dimensions are created equally!

k-NN: Details

Inductive Bias:

1. Close points should have similar labels
2. All dimensions are created equally!

