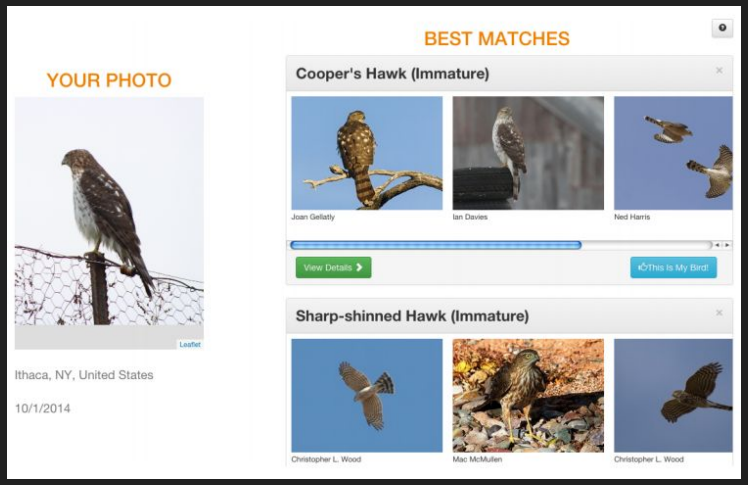
# Classification

# Example application - Species Identification

Given some features derived from an image, audio recording, a dna sample, or some physical measurements can we identify the species from which the data came?




Van Horn et al. Building a bird recognition app and large scale dataset with citizen scientists CVPR 2015

# Example application - Species Identification

Given our feature vector **x**, can we predict
the correct species (i.e. class label) y?

=

Crow, Hawk, …, or Robin?

Van Horn et al. Building a bird recognition app and large scale dataset with citizen scientists CVPR 2015

# Supervised classification

There exists a whole host of different classification algorithms each with their own strengths and weaknesses.

**Popular Algorithms**

Nearest Neighbour

Logistic Regression

Support Vector Machines (SVM)

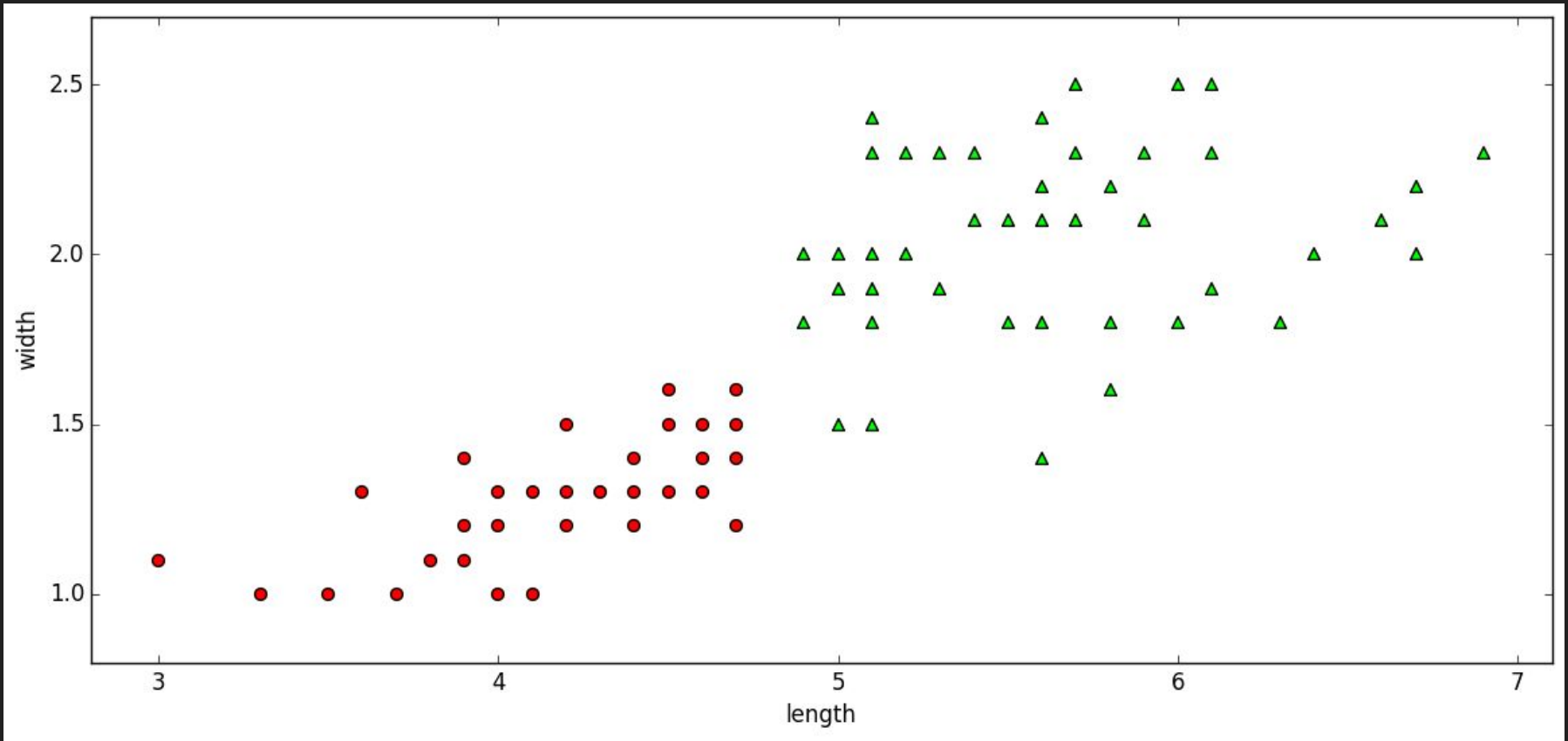
Decision Trees

Random Forests

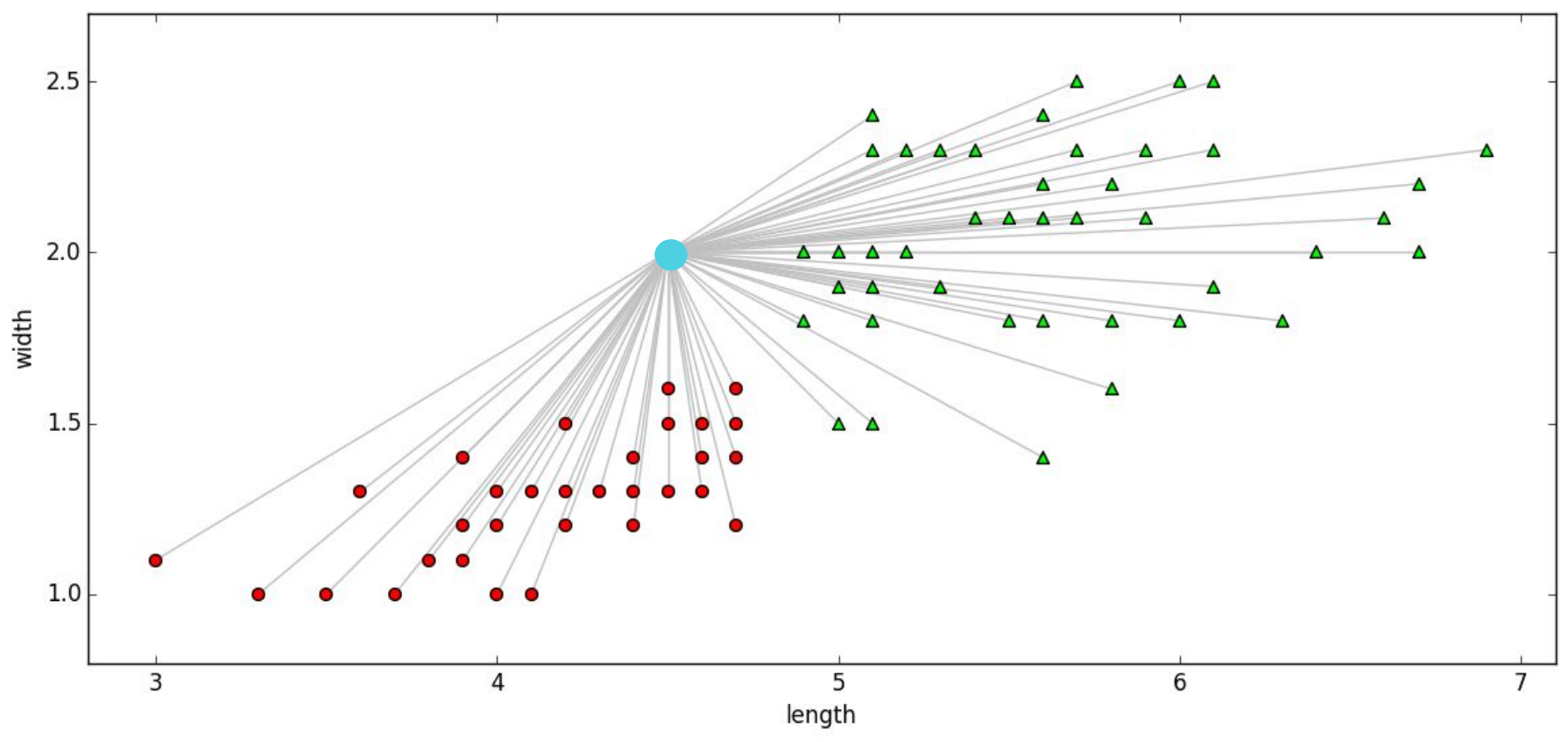
Neural Networks

Gaussian Process Classification

# 2D Dataset with 2 classes

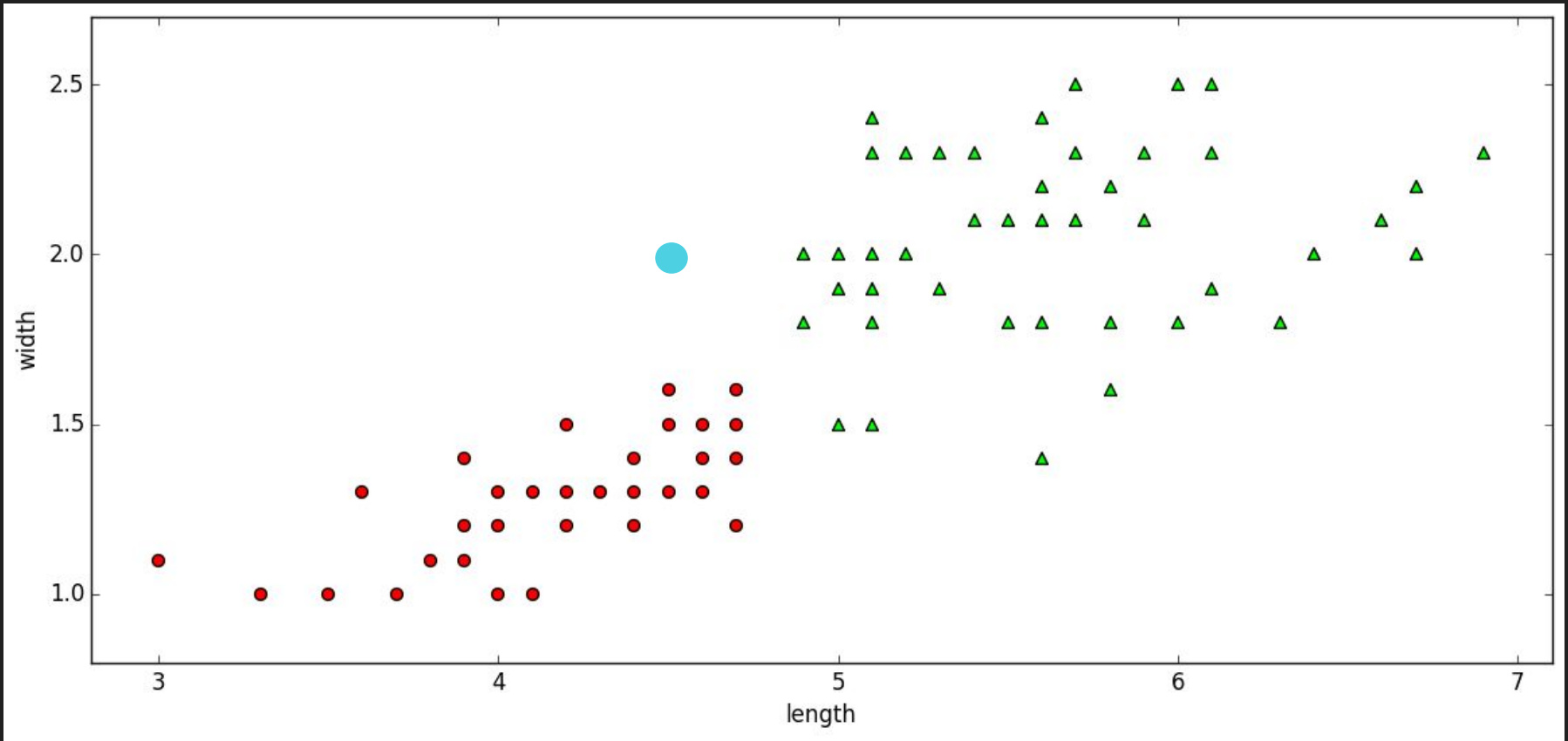


# Recall Nearest Neighbour

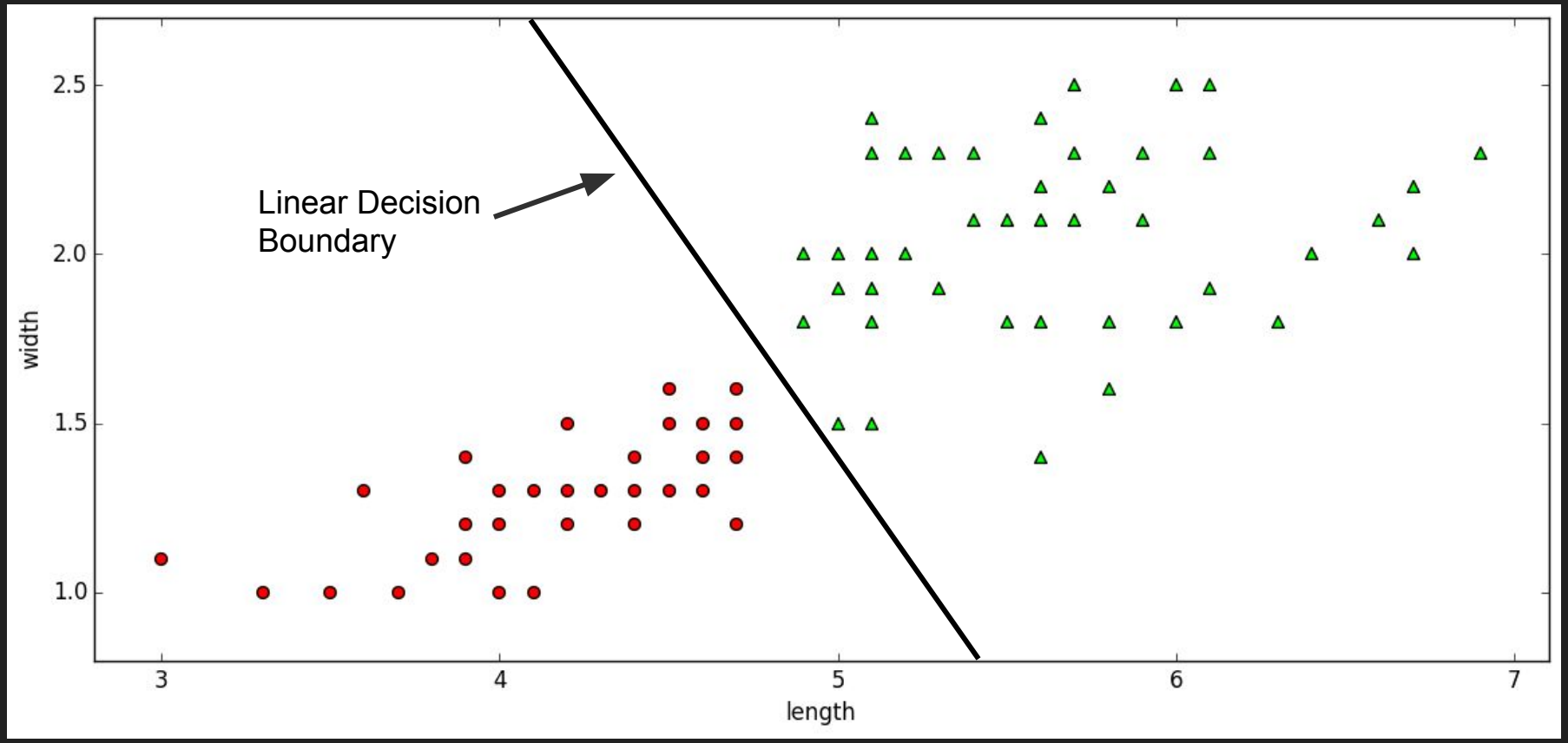



For every test point we have to compute the distance to every training point.

# Is there a way that involves less computation?

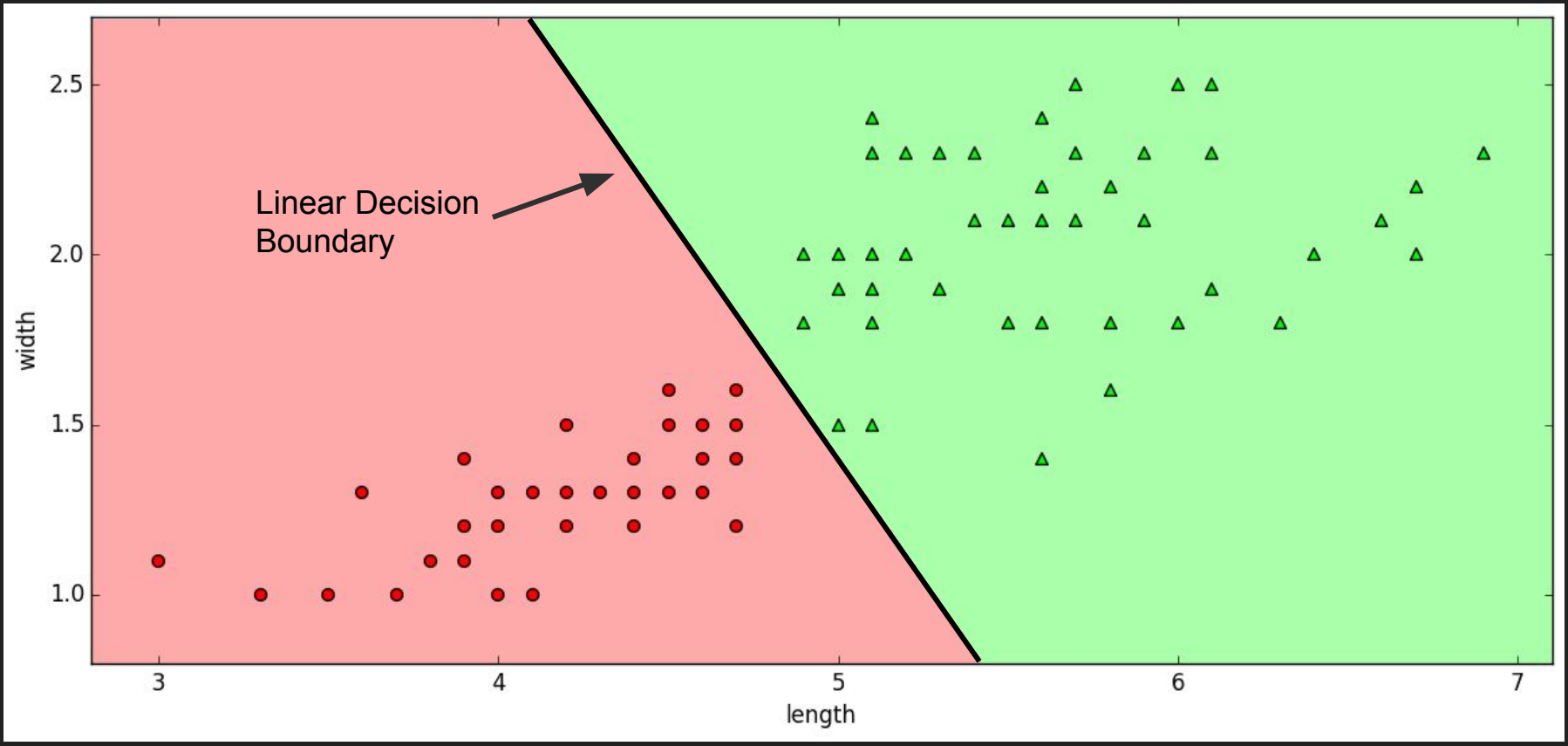


# Linear classifier - separable




Linear SVM C=1

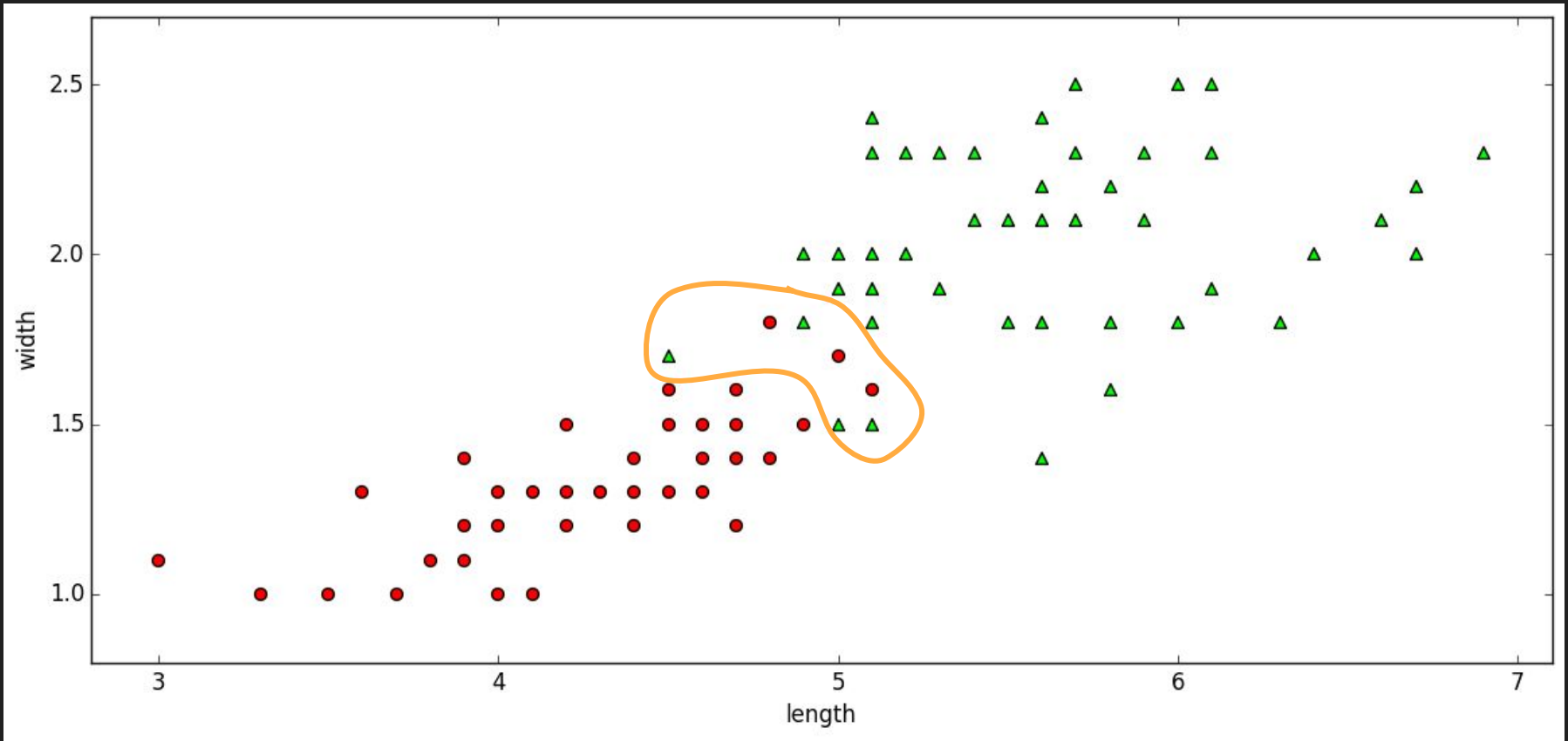
# Linear classifier - separable



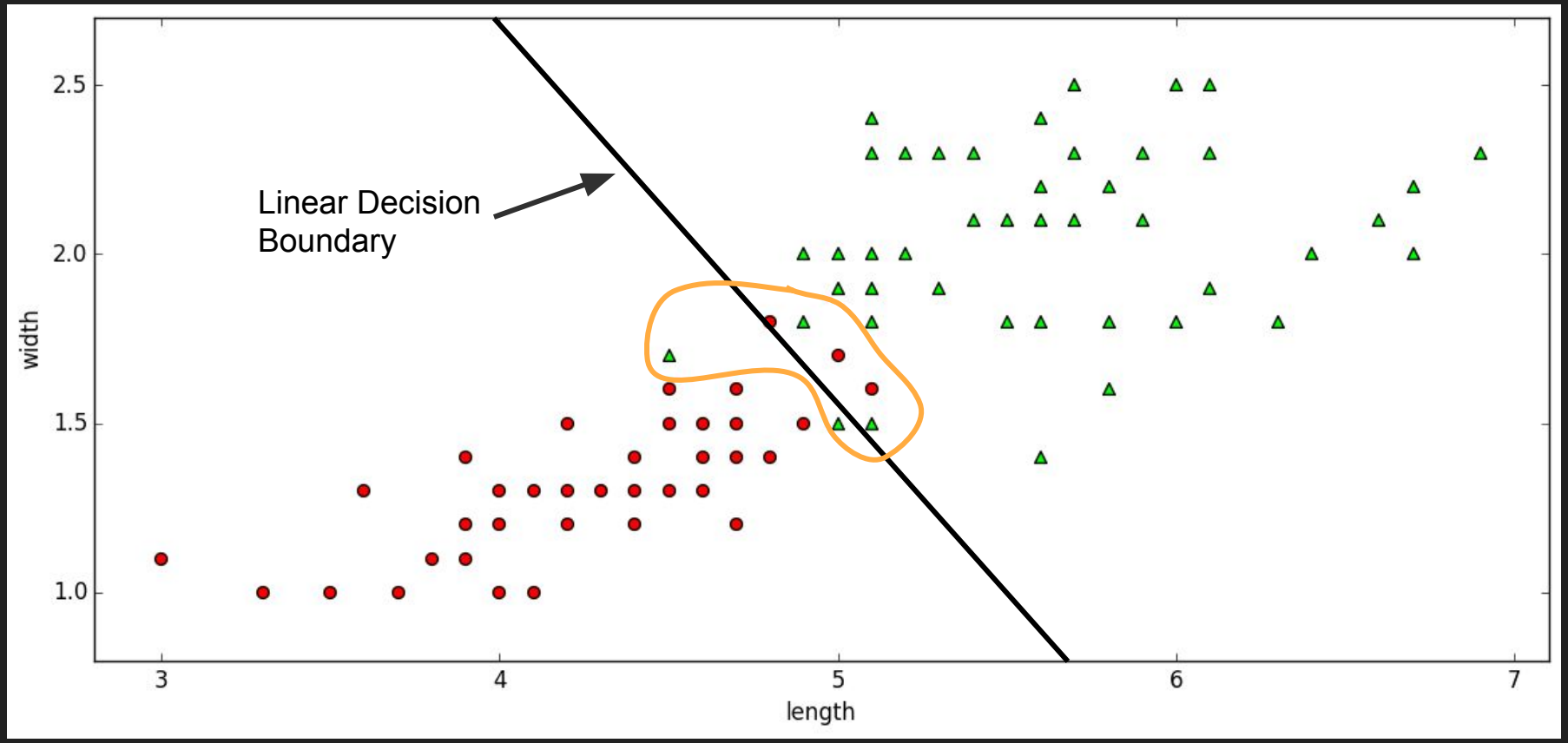

Simply check which side of the decision boundary our test point lies on.

Linear SVM C=1

# But what if we have nonlinear data?

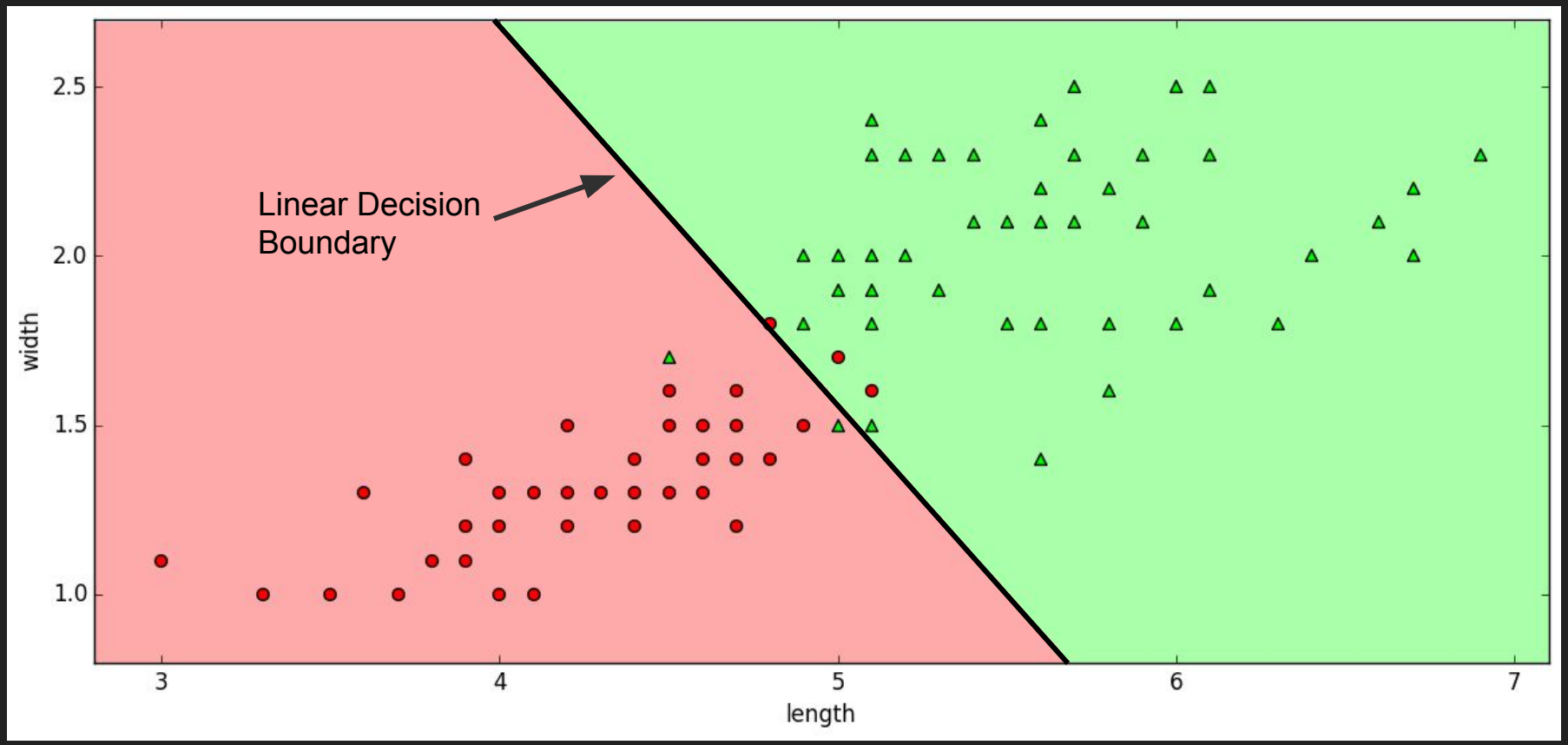


# Linear classifier - not separable



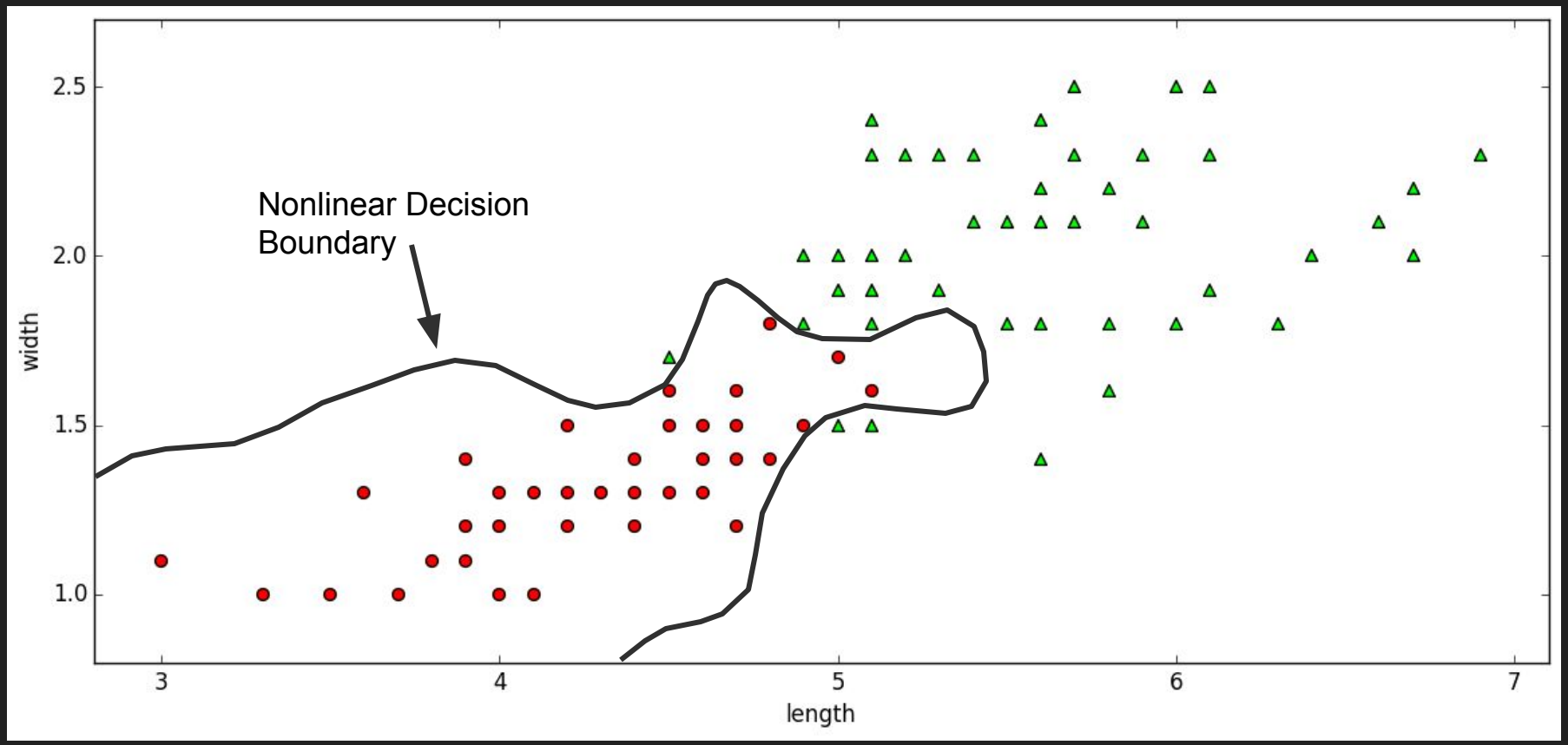

Linear SVM C=1

# Linear classifier - not separable



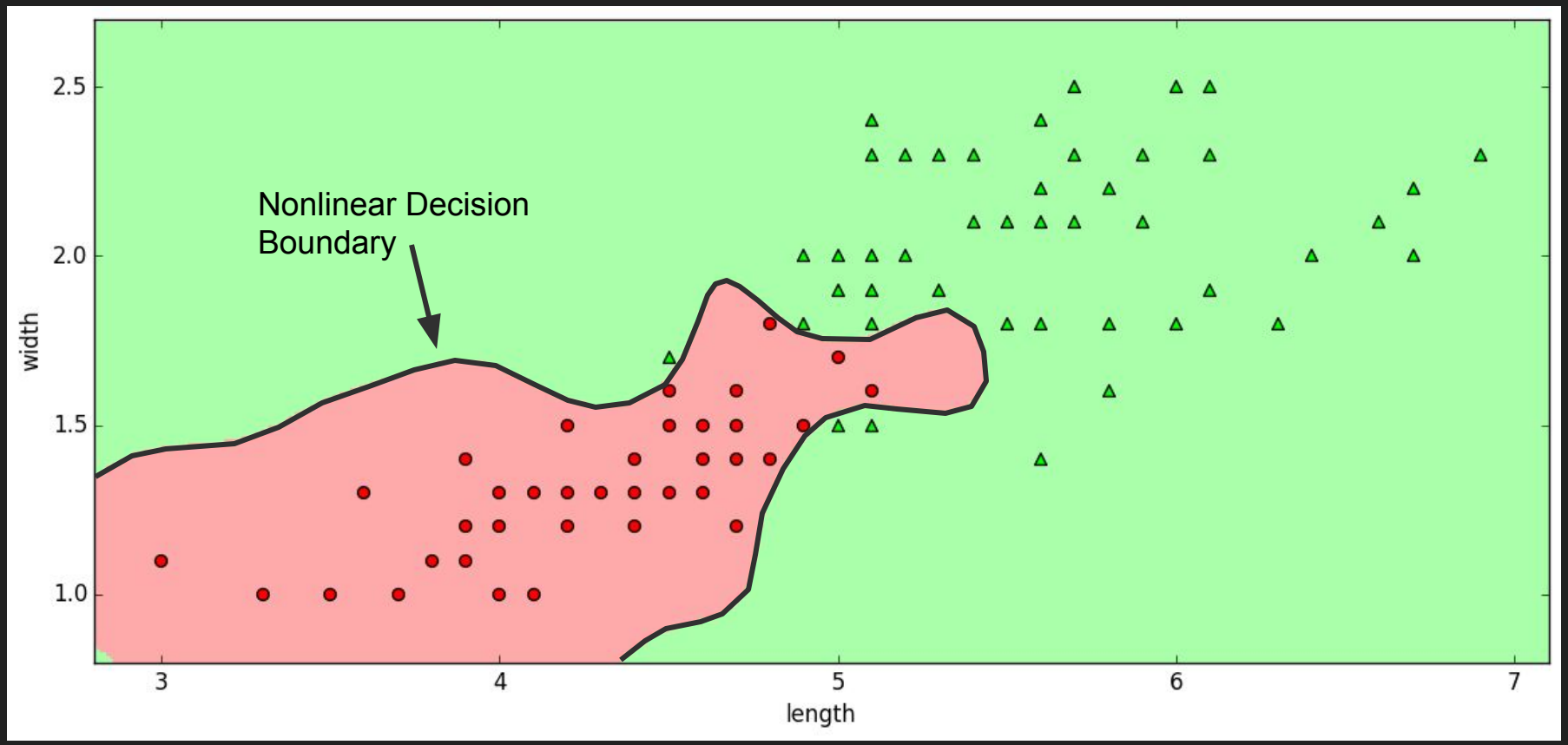

Linear SVM C=1

# Nonlinear classifier




SVM with RBF Kernel
C=100, gamma = 10

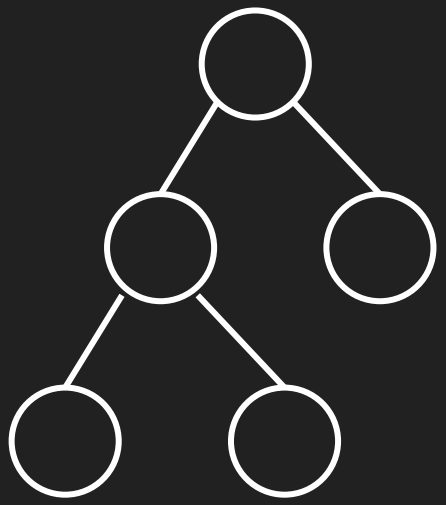
# Nonlinear classifier



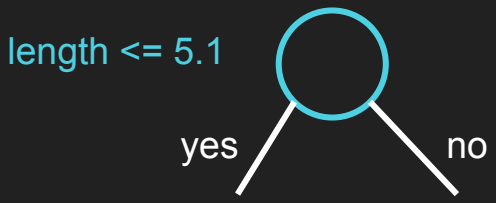

SVM with RBF Kernel
C=100, gamma = 10

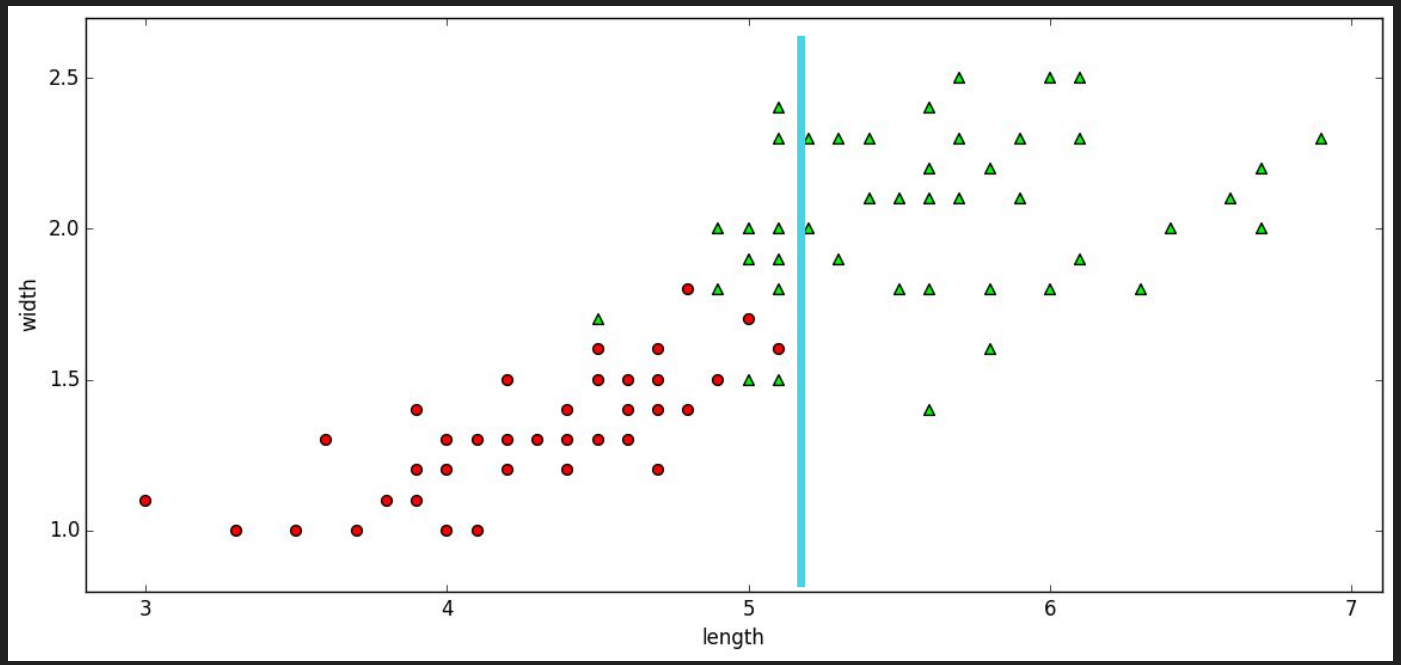
# Decision Tree

Partitions up the feature space using very simple decision rules.

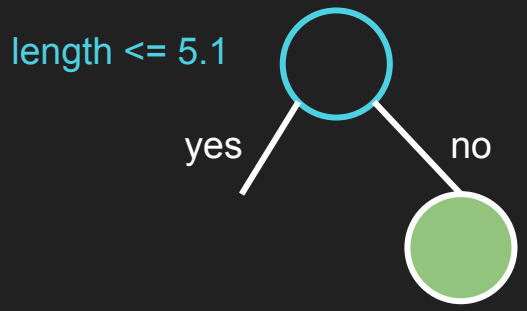


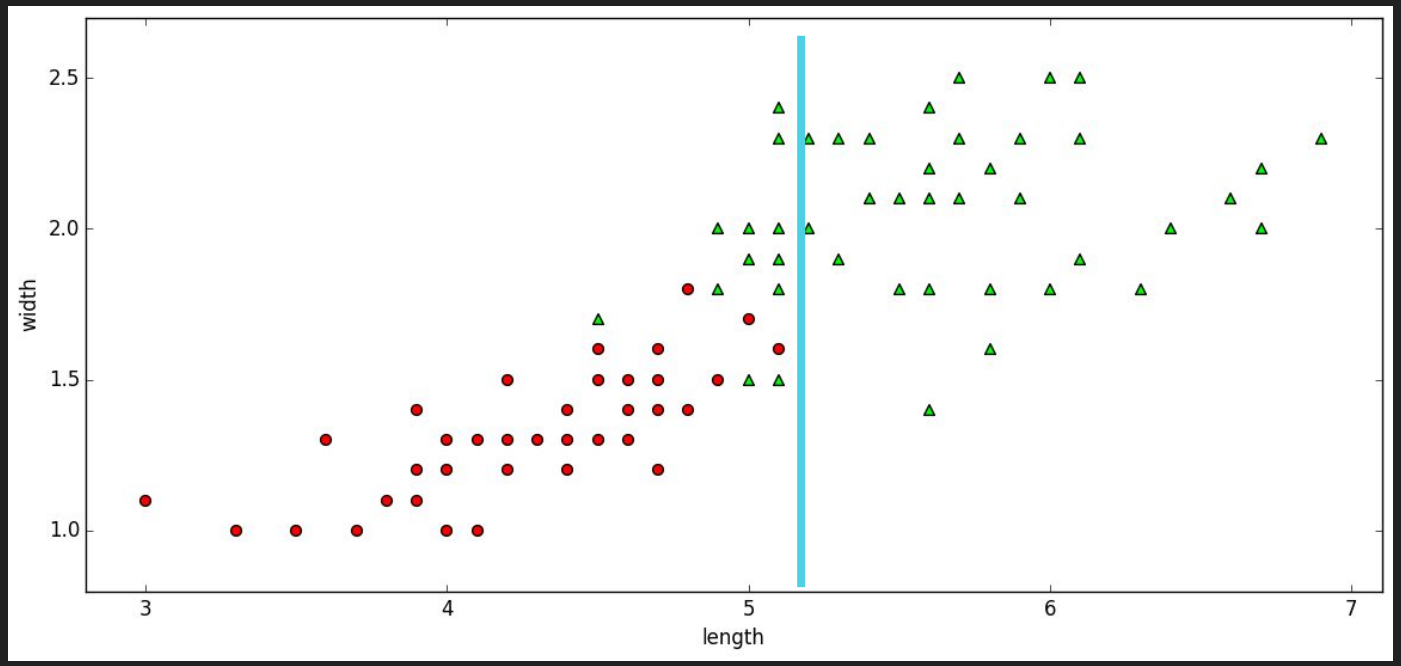
# Decision Tree



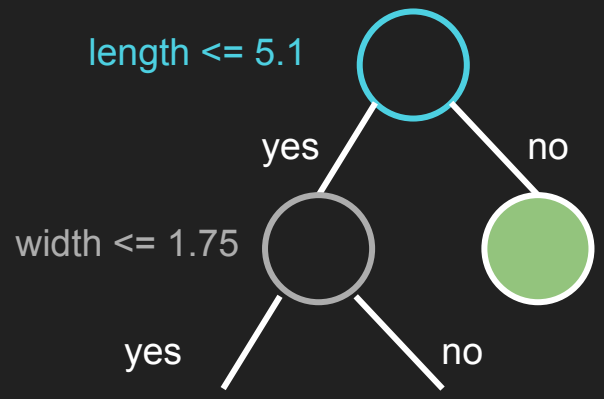



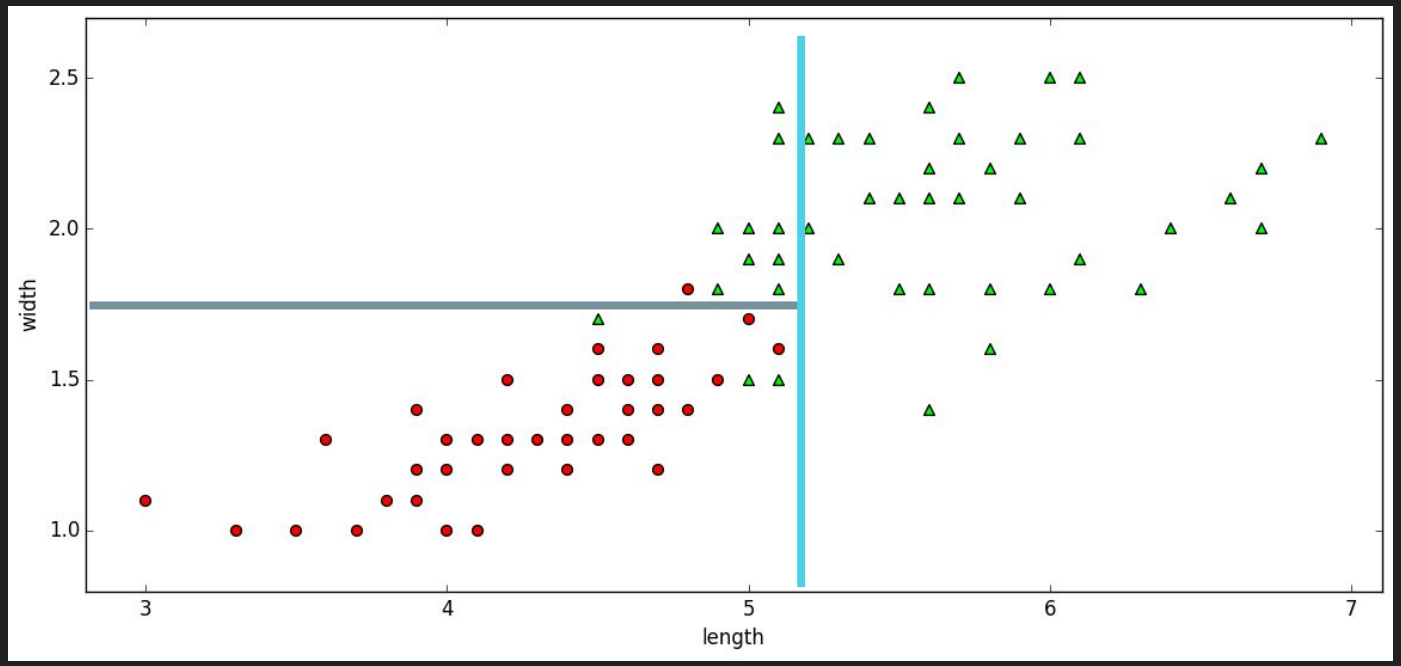
# Decision Tree



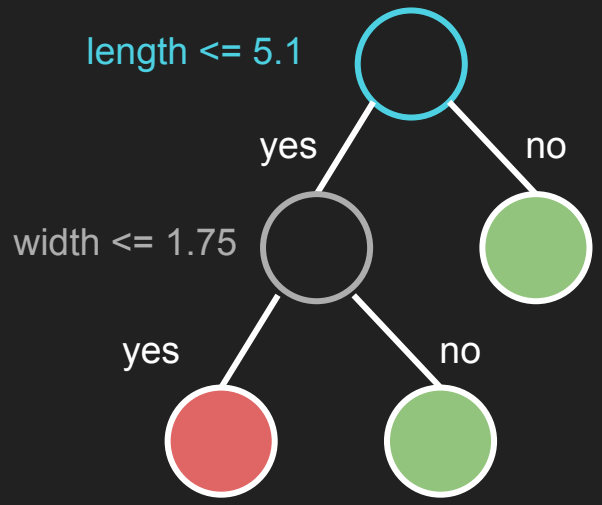



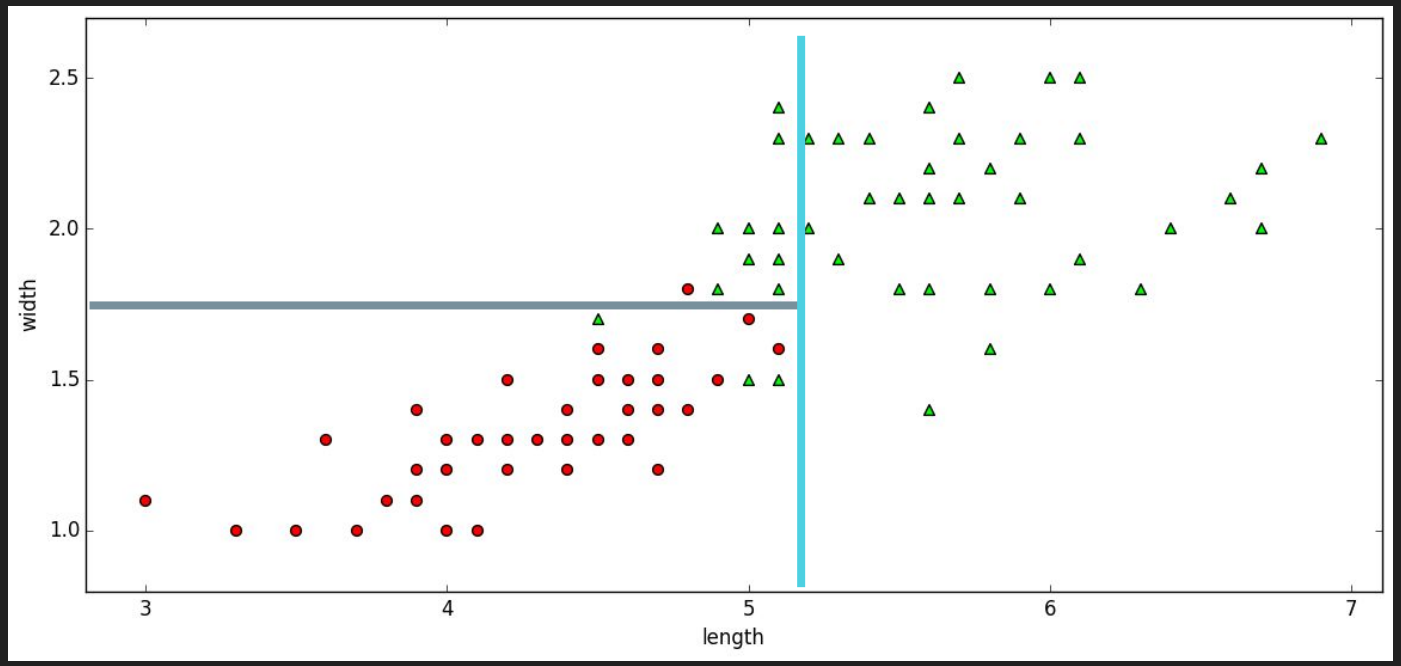
# Decision Tree



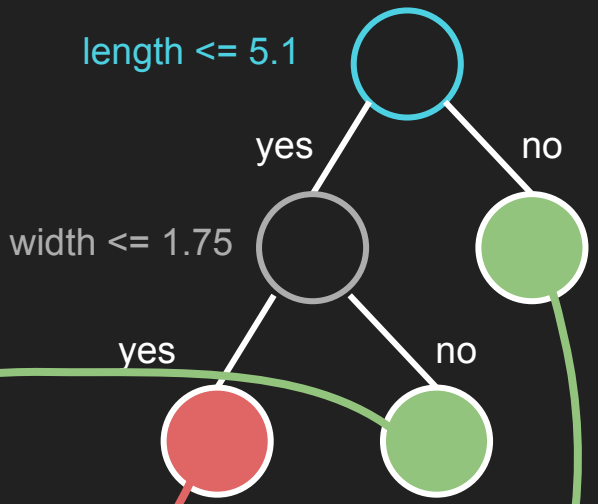



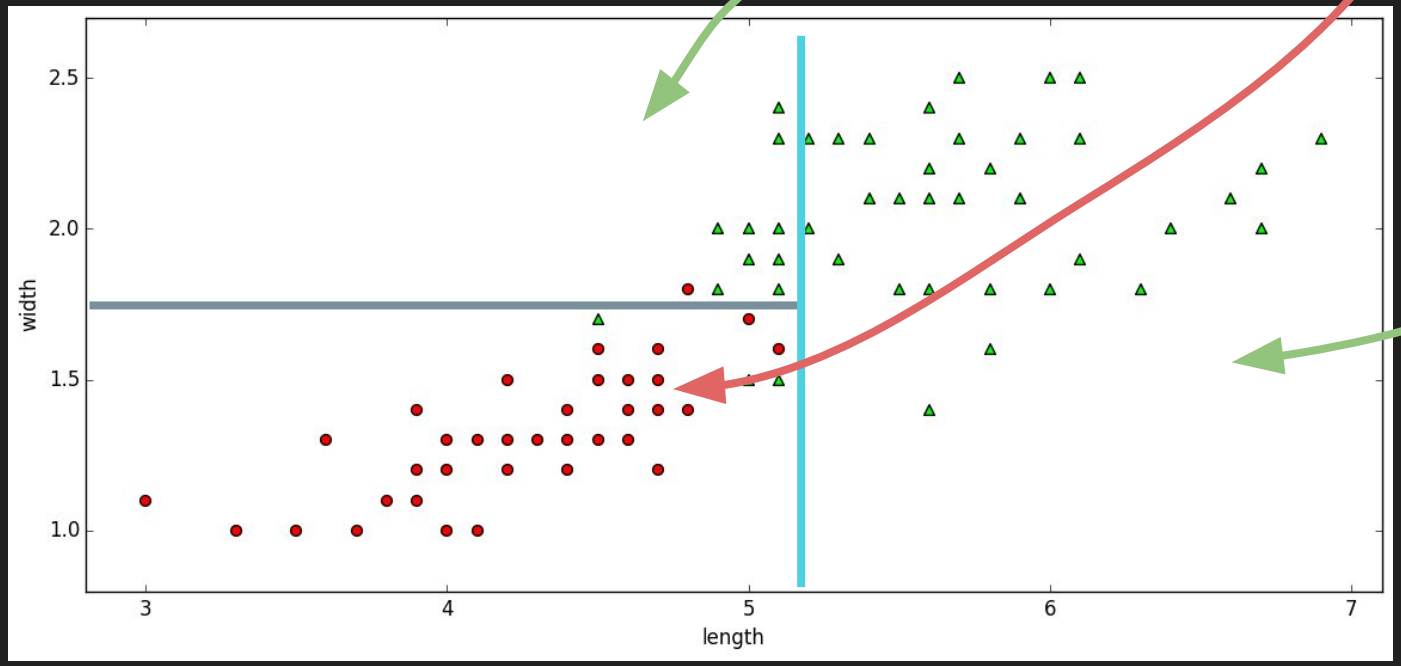
# Decision Tree



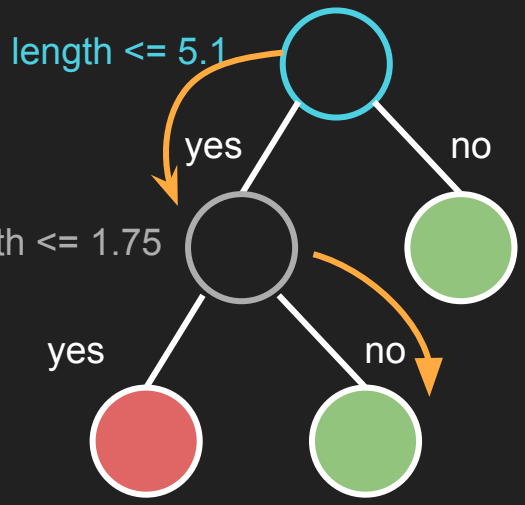
# Decision Tree

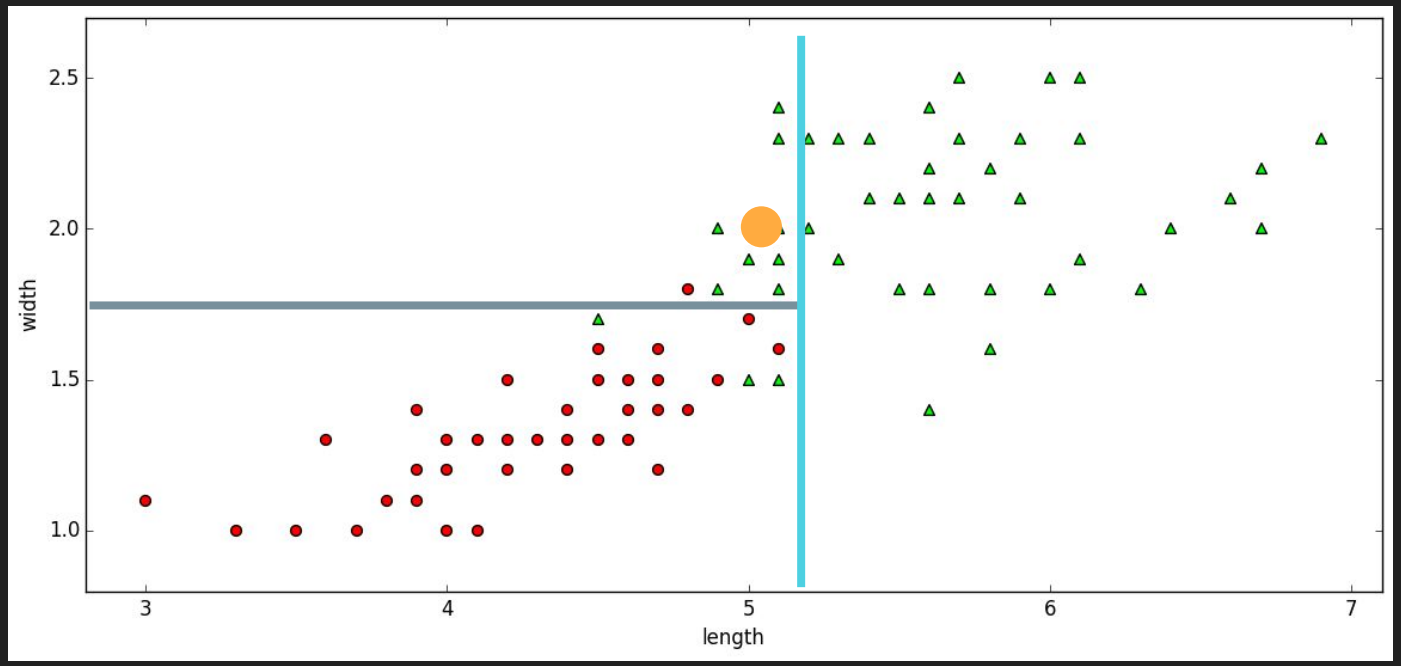


# Decision Tree vs Nearest Neighbour

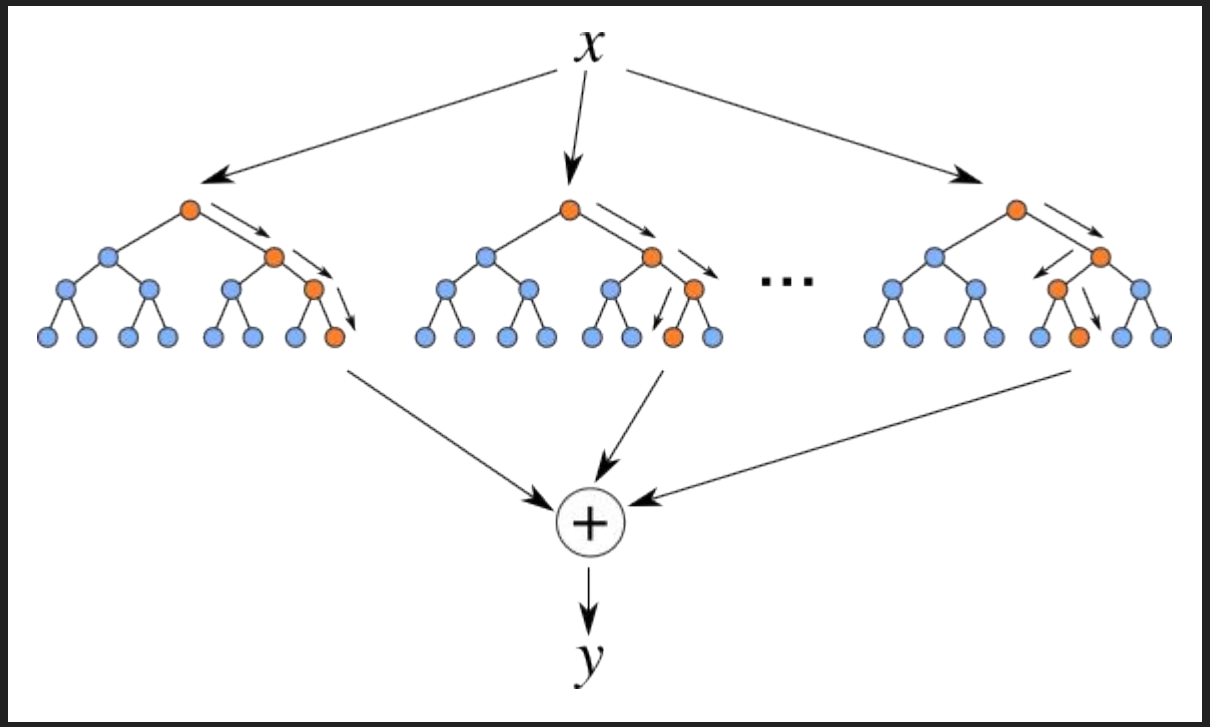
Here, for each test point we only have to do, at most, two simple tests to determine its class.



# Random Forests

A Random Forest is a collection (or ensemble) of decision trees, where each tree is trained on a different random subset of the data.




Wisdom of the crowd!

http://en.likefm.org/artists/images/Random+Forest

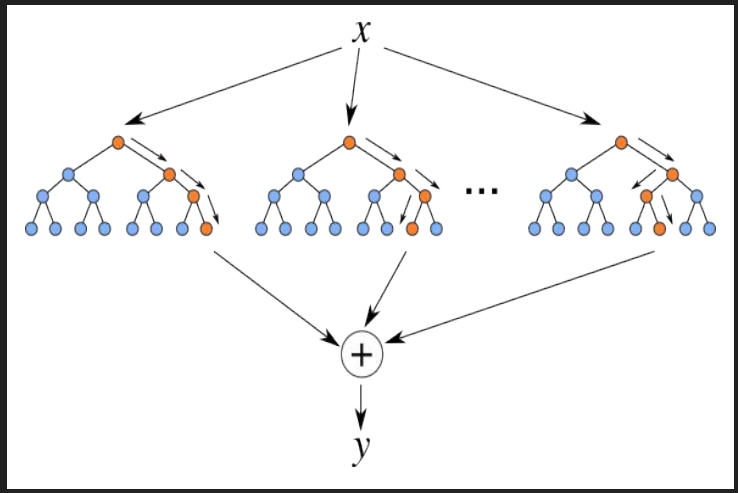
# Random Forests

Are fast to train and test.

Can deal with noisy features.

Handle features of different units.

Can cope with large datasets.




http://en.likefm.org/artists/images/Random+Forest

# Random Forests interactive demo

forestjs

http://cs.stanford.edu/people/karpathy/svmjs/demo/demoforest.html

# Which algorithm to choose?

Short answer: It depends!

No silver bullet, but often it is sensible to first try a Support Vector Machine or Random Forest.

This will give you an idea of how separable your data is. The next step is to try different features, and perhaps even collect more training data.

# How much training data do I need?

Short answer: It depends!

It depends on how easy it is for your classifier to separate your data.

Some problems are relatively easy and don’t require lots of data, others such as species identification in images can require 10,000s.

# Practical example

3_classification.R