

CFCS1

Vector-based Machine Learning

Miles Osborne

School of Informatics
University of Edinburgh
miles@inf.ed.ac.uk

January 21, 2008

Overview

Much of cognition deals with *learning*:

- Children learn how to talk.
- Cats learn how to wake people in the morning.
- Our eyes learn how to recognise objects.

Machine Learning deals with getting computers to learn to solve some task.

1 Overview

2 Example

3 Knn

4 Perceptrons

Overview

A simple model of learning is as follows:

- Take a set of *examples* of the task.
- Have a *teacher* mark each example with a *label*.
- Build a computer program to *predict* the labels of each example.

The better we can predict labels of examples, the better we are at learning.

Overview

Machine learning in Cognitive Science has two roles:

- We can build models to mimic some process:
 - Models of neurons are claimed to behave as real neurons.
- We can build models to study Cognitive Science:
 - How does eye gaze connect to reading ability?
- In either case, we use models to predict some phenomena.

Example

Here is what our teacher labelled:

Name	Facts			Label
Fluffy	Meows	Has fur	Bites	Cat
Tiddles	Meows	Has fur	Does not bite	Cat
Fido	Does not meow	Has fur	Does not bite	Cat
Rex	Woofs	Has fur	Bites	Dog
Rover	Woofs	Has fur	Bites	Dog

Note also we have facts about our animals.

Example

Suppose we want to learn whether an animal is a cat or a dog:

- The teacher takes a set of animals.
- For each animal, the teacher attaches a label (*cat* or *dog*).
- Each animal has a single label.

This is a cut-down version of how people learn to tell objects apart from each other.

Example

The learning task is to predict whether an animal is a cat or a dog based upon observed facts and what our teacher has told us.

Example

Our first job is to represent the data:

- A *feature* captures some aspect of an example:
 - Does the animal bite?
 - Does the animal meow?
 - Does the animal have fur?
 - Is the animal called Fido?
- Not all features are equally useful!

Example

Now, each example becomes a vector:

Fluffy	[1, 1]	Cat
Tiddles	[1, 0]	Cat
Fido	[0, 0]	Cat
Rex	[0, 1]	Dog
Rover	[0, 1]	Dog

This is our *training set*.

Example

Suppose we use the following features:

Feature	Identifier
Does the animal meow?	F1
Does the animal bite?	F2

We can now represent each animal in terms of features:

Name	F1	F2	Label
Fluffy	1	1	Cat
Tiddles	1	0	Cat
Fido	0	0	Cat
Rex	0	1	Dog
Rover	0	1	Dog

Knn Classifier

A learner needs to predict the label of a new example:

- Represent each animal as points in a vector space.
- Associate the supplied label with each vector.
- Compare the example against all training examples.
 - This uses a *distance metric*.
- The k nearest examples are found.
- We use the label(s) of the k nearest example(s) as the predicted label.

This is a K nearest neighbour (Knn) classifier

Knn Classifier

Is Spot a dog or a cat?

Spot Does not meow Has fur Bites

- Feature representation:

f1: 0 f2: 1

- Vector representation:

[0, 1]

Perceptron

Our *Knn* model has not really told us much about the task:

- We simply compare our test example against all training examples.
- There is no notion of generalisation.
- We cannot (directly) reason about the value of individual features.

A *perceptron* is a simple model of neural processing.

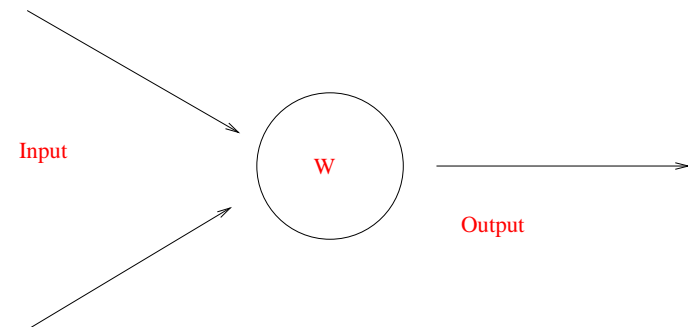
Knn Classifier

The final column is the distance between the representation of Spot and the training examples:

Name	Vector	True label	Absolute Distance
Fluffy	[1, 1]	Cat	1
Tiddles	[1, 0]	Cat	1.4
Fido	[0, 0]	Cat	1
Rex	[0, 1]	Dog	0
Rover	[0, 1]	Dog	0

Spot is probably a dog.

Perceptron



Perceptron

Under a perceptron, we label example as follows:

$$f(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} > 0 \\ 0 & \text{otherwise} \end{cases}$$

- Here \mathbf{x} is our example (a vector).
- $f(\mathbf{x})$ is the predicted label.
- \mathbf{w} is a *weight vector*:
 - The weight vector tells us how to treat vector components.
 - Each weight vector component is a *parameter*.
 - Numerical optimisation can be used to set these values.

Perceptron

Suppose we guess \mathbf{w} as the weight vector $[1, 0]$.

Name	Vector	$f(\mathbf{x})$	$I(\mathbf{x})$	Loss
Fluffy	$[1, 1]$	1	1	0
Tiddles	$[1, 0]$	1	1	0
Fido	$[0, 0]$	0	1	1
Rex	$[0, 1]$	0	0	0
Rover	$[0, 1]$	0	0	0

- Our total loss is 1 (we think Fido is a dog, when Fido really is a cat).
- We predict that Spot is a dog ($[0, 1] \cdot \mathbf{w} = 0$)

Perceptron

Setting the weight vector:

- For each example, we know what the *correct* label is: $I(\mathbf{x})$
- We can work-out the mistakes our model makes:

$$\text{loss}(\mathbf{x}, \mathbf{w}) = \begin{cases} 0 & \text{if } f(\mathbf{x}) = I(\mathbf{x}) \\ 1 & \text{otherwise} \end{cases}$$

- Learning is now the job of making as few mistakes as possible on the training set.
- We can solve this as a numerical optimisation problem:
 - Search for a weight vector \mathbf{w} which has the smallest total loss for all training examples.

Summary

- We looked at two vector-based machine learning methods:
 - *Knn*
 - Perceptrons
- Concepts:
 - Using features to represent examples.
 - Using a distance function to relate examples to each other.
 - Using parameters to give insight into the value of features.
- General question: how do these models relate to cognitive science?