Subject: Re: slow processing of my k-nearest neighour code
Posted by Karl Schultz on Mon, 14 Aug 2006 15:08:38 GMT
View Forum Message <> Reply to Message

On Mon, 14 Aug 2006 10:11:30 -0400, Ben Tupper wrote:

> humphreymurray@gmail.com wrote:
>
>>> ; Calculate the squared distance for each attribute.
>>> squared = make_array(num_training_elements, num_attributes)
>>> for attrib = 0, num_attributes-1 do begin
>>> squared[*,attrib] = (testing_data[i, attrib] -
>>> training_data[*,attrib])^2
>>> endfor
>>>
>
> Hi,
>
> You might try replacing the above for inner-loop with the following
>
> squared = (testing_data - training_data)^2
>
> Since IDL is array saavy it will perform the operation element by
> element for you quite quickly (as well as make the "squared" array for you).
>
> You might be able to eliminate the outer-loop, too, but I am less sure
> of that. Take a peek at the for-loop bible at
>
> http://www.dfanning.com/tips/forloops.html
>
> Good luck,
> Ben

The above will help a great deal. But if the OP is going to move to
larger data and/or it gets really important to lower the execution time,
he may want to implement a compiled-code DLM. There's a package out there
called ANN (Approximate Nearest Neighbor) that computes the k-nearest
neighbors using either an exact calculation or a faster approximate
calculation.

I needed this once as part of a surface reconstruction project and wrote a
really narrow C DLM interface wrapper that exposed the ANN features I
needed. This isn't that hard to do, but you do need to know how to code
IDL DLM's. ANN is provided in source code form and is pretty easy to work
with. And it does the job nicely.

Karl