**Assignment 4**

**In brief:**

Using the supplied dataset in the form of a CSV file, and the supplied Shiny App, you are tasked to extend this app to consider 3 more model types. The app demonstrates the issue of model selection and has been designed around the ***caret*** package.

This assignment will demonstrate your end-to-end skills such as

* + learning the ***caret*** style of thinking about model optimisation and selection
  + performing hyper-parameter optimisation through cross validation
  + using parallel processing for cross validation of model types that support this
  + producing a set of candidate models appropriate to the data
  + selecting the best model using an unbiased measures
  + quantifying the performance of the best model

**In detail:**

The template has already been developed for you. You will need to familiarise yourself with this code and be comfortable modifying it.

You should research the caret documentation - in particular ***caret::train()***. I found the resource <https://topepo.github.io/caret/> to be helpful (especially sections 6 and 7). There is bound to be lots of material to read.

Choose an additional 3 regression based model types to add to the Shiny App in a manner similar to and consistent with the three models already in place. The model selection dialogue is a manual one (and should remain a manual process). Ensure that performance of your models will be calculated (in the last tab) when each of your models is selected as the model-of-choice. Research what ***caret::defaultSummary()*** does.

**The Data:**

The data does not have missing values. Nor does it have outliers. Consider how many training observations you have available and choose model types that are appropriate to this scenario - you are definitely in the zone of over-fitting. Model types with hyper-parameters should be high on your shopping list.

**The Code:**

The Shiny code uses the ***caret*** way of training models. This uses cross validation to generate unbiased stats and it allows hyper-parameters to be optimised. This style of training automatically locates the best (with respect to hyper-parameters) model and calls this the "finalModel". However it should be acknowledged that this general way of training stops us seeing many familiar plots and displays to which we are accustomed (e.g. regression vector information.) In fact to display or plot something useful about the training of your additional models do some experimentation and see what is available.

Since cross validation is a slow process (and then add grid-searching for parameter ranges), the code uses parallel processing where possible. This style should work with ***Windows***. Note that ***train(method="glmnet",...)*** does not permit parallel processing; it crashes if you try. You may find one or more of your additional models exhibits the same issue.

Notice that the ***trControl*** variable sets the cross-validation behaviour that all models should share - as this provides a fair basis for comparing them.

**Use the course *LEARN* website to upload the*ui.R*, server.R, (global.R) files. You should ensure that the *Ass4Data.csv* file is sought in the same folder as these files. You do not need to submit the *CSV* file.**

**Marking:**

You must work alone.

**Overdue Work:**

Overdue work will incur a penalty of 10% for each day it is late.