Assignment 3

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 in order to find the best performing regression model. The app demonstrates the issue of candidate model selection and has been designed around the ***caret*** package. This demonstrates the idea of "no-free lunch" - there is no method that works optimally in all situations.

The caret package is a framework for best-practice in regression and classification. It enables you to apply each of 237 different methods from Data Science to your data and optimally generate any hyper-parameters through resampling. The candidate models can then be fairly compared in order to select several top models that should then (and only then) be assessed against the official **test** data.

This assignment will demonstrate your skills such as

- learning the ***caret*** style of thinking about model optimisation and selection
- choosing appropriate feature engineering (in the right order)
- performing hyper-parameter optimisation through resampling
- using parallel processing for methods that

support this
    - producing a set of candidate models appropriate to the data
    - assessing candidate models relative to a base (null) model
    - selecting the best model using an unbiased measure
    - quantifying the performance of the best model
    - writing up your investigation in a report

In detail:

The template has already been developed for you. You will need to familiarise yourself with this code and be comfortable modifying it. It should be a matter of adding method-tabs and populating them with plots templated off the existing ones. You may also change the pre-processing of the existing method-tabs. Each model has different different summary and plot facilities that are unique to that method. For example, observe how different the outputs are for the supplied models. Use your judgement to show the most relevant information in your app.

You should research the caret documentation https://topepo.github.io/caret/ (especially sections 6 and 7). Notice that caret maintains a list of the methods and their characteristics (tags) which may help in

choosing which methods to try. Clearly we are only dealing with regression methods in this assignment. These are made available to you in the tab called "Available methods" - notice that the ban icon indicates that the package is not yet installed. You will need to deal with that yourself.

Choose additional regression based method types to add to the Shiny App in a manner similar to and consistent with the three models (and the null model) already in place.

Before you submit your revised app, ensure that:

- the code runs after you have done **Session/Restart R** & **Session/Clear Workspace...** (in RStudio)
- the default for the radio button selecting the best model is set to your best model.
- the defaults for the pre-processing are sensible & optimal for the particular method. Take care to get the order of the steps sensible & optimal, as well.

You also need to submit a static report on your process of finding the best model. It should include (as a minimum)

- a screenshot of the model selection visualisation
- a table of the methods, pre-processing employed, resampling results (hyperparameters and metrics)
- a statement as to what is the best performing model. **Assume "best" means "minimum RMSE".**

- an explanation on how the best model works - do some research on your best performing method **and** hypothesize why it suits the data so well.
- a screenshot of the predicted versus actual visualisation of test data
- statistics for the best performing model based on test data
- how would the optimum model change if transparency were very important? That is, what would be the best performing transparent model?
- whether you recommend utilising an ensemble of some of the top models and why? - do some research on when you should not ensemble

Some hints:

1. Slower-to-train does not mean better. Avoid slow methods unless you have plenty of spare time.
2. You may add extra columns (using the existing columns). For example non-linear functions and/or interactions.
3. If your method crashes, turn off parallel processing in order to catch the error messages and warnings. This will run slower.
4. You should not need to substantially alter the shiny app. If you think you need to, come and speak with me first. You may add further recipe steps (in global.R) but I suspect you will not need to.
5. While bedding things down (i.e. testing & debugging), making a smaller train-set (i.e. train/test

slider set to say 30%) may make your training run faster.

6. Because there is no ***set.seed*** used anywhere, the results are not precisely reproducible. This is deliberate and might be important when I try to reproduce your work. Do check the stability of your conclusions. In particular - are the best hyper-parameters on the edge of the "grid"? Why might this be a problem?
7. A diverse set of methods makes a good candidate model set.
8. If a package is required that you have not yet installed, caret will prompt you from the console. If your app seems to freeze, check the console for a dialogue about installing a missing package.
9. You may need to look at 10-30 methods before you can be confident that you have found an optimal model. There is an element of luck involved so maximise your luck by trying many methods.
10. Set your method specific preprocessing initial value to your recommended choices so I can reproduce your tabulated results in the report.
11. Be careful not to sneak ahead and peek at the test results to choose your best model. Use the resampled results to make this decision (the resampled results are NOT training results and are fair to use to choose the best model)
12. The "SavedModels" folder contains files that reload your trained models when you start the app. Clean

this out of models that you abandon or startup will become quite slow.

13. When you think you have finished re-read this material to ensure you have not forgotten anything.

Submit:

- The source code for the modified app: server.R, ui.R, global.R
- A pdf report (I reckon 4 pages is about right)
- The "SavedModels" folder

Marking

Since there is an element of luck involved in whether you find a good model, I will **also** judge you on

- the number and choice of candidate methods coded in the app
- your choice of method specific pre-processing steps (especially their consistent order - for example "knnimpute, naomit" is pointless if knnimpute removes all missing values)
- the completeness of the report
- clarity of the report