Growth Classifier

Walkthrough and presentation

Index

- R Code Breakdown
- Example of model run
- Example of model output and analysis
- Ideas for improvements going forward

R CODE – Explanatory notes (Part I)

```
election  Match case  Whole word  Regex  Wrap
---
title: "Growth_Classifer_Apsara_V015"
author: "Samuel King"
date: "26 October 2017"
output: html_document
---

## ------------------------------------------------------ PREPROCESSING ------------------------------------------------------------------

## Set memory

```{r Memory}
```

Load libraries

```{r Library}
```

Load generic caret models

```{r Caret}
```

Data cleaning and derived variables

```{r Data cleaning}
```


-- DATA ANALYSIS OF REAL DATA TO USE AS COMPARISON TO MODEL OUTPUTS ----------------------

switch rate analysis - historic data

```{r}
```

Up/down sales growth analysis - historic data

```{r}
```

Distribution analysis - historic data

```{r}
```


-- PREVIOUS BENCHMARK CHECKS --

Previous benchmark model - function (plus parallel processing)

```{r}
```

Previous benchmark model

```{r}
```

-- FUNCTIONS FOR FIRST MODEL (GROWTH / NOT GROWTH) ---------------------------------

Function for converting model to output
Chunk 22   R Markdown
```

All straight forward pre-processing steps (cleaning memory, loading libraries, loading Caret packages, data cleaning and variable derivation)

This section is some stand-alone data analysis on the underlying data to be used as comparison to the modelling results. For example a look at real switch-rates, up/down sales growth, and distribution analysis.

Code to run the previous benchmark model (KNNX2) to use as comparison. Also has the previous model combined with the ensemble model here to see if we can improve the second step. You only need to run this if you want to build the outputs for this model again, otherwise skip.

R CODE – Explanatory notes (Part II)

Functions needed to run the first model (Growth/Not Growth) and compile output into results for analysis. The most important function here is the last one 'SKhibacktestnonode_m1'. This is the main function for running models for the G/NG classifier. The input variables are fairly self explanatory but allow you to change a range of parameters from model type to sampling method parameters etc. The other two functions are simply used to take the output of the model and create performance results and distribution analysis results and save them to file.

This set of scripts runs the first model (Growth / Not Growth). The first section 'create input table' in simply used to subset the training data if you want to make model runs quicker (for sense-checking and testing purposes). The Ensemble approach code runs the three models and averages them at a given threshold. The threshold modelling is the code used to select that optimum threshold.

R CODE – Explanatory notes (Part III)

```
4282
4283 ## ---------------------------------------------------------- FUCNTIONS FOR MODEL 2 (MATURE CYCLICAL / MATURE STABLE) ----------------------------
4284
4285 # Function to combine model 1 and 2
4286
4287 ```{r}    ```
4300
4301 # Function for converting model 2 and 3 (combo of mod1 and 2) to output & switch rate analysis
4302
4303 ```{r}    ```
4378
4379 # Function to smooth switching
4380
4381 ```{r}    ```
4420
4421 # Function to save mod 2 and mod 3 (combo of mod1 and 2)
4422
4423 ```{r}    ```
4454
4455 # Function for model training - MODEL 2
4456
4457 ```{r}    ```
4527
4528
4529 ## ---------------------------------------------------------- RUN SECOND MODEL (MATURE CYCLICAL / MATURE STABLE) ----------------------------
4530
4531 # MODEL - 2 - create input table (N can be set to < 100% of data for fast processing test runs)
4532
4533 ```{r}    ```
4550
4551 # MODEL - 2 - Spot checking algorithms (STILL A WORK IN PROGRESS: MORE MODELS/PARAMETER TUNING TO DO DONE HERE)
4552
4553 ```{r}    ```
4726
4727 # MODEL - 2 - Systematic caret algorithm loop (STILL A WORK IN PROGRESS: FOR LOOP TO TRY EVERY CLASSIFICATION MODEL - SOME MODELS VERY SLOW)
4728
4729 ```{r}    ```
4757
4758 # MODEL - 2 - Current best model
4759
4760 ```{r}    ```
```

Functions needed to run the second model (MC/MS) as well as combine it with the first model to create the final classifier (mod3). Again there are also functions here to compile output into results for analysis. Again the most important function here is the last one 'SKhibacktestnonode_m2'. This is the main function for running models for the MC/MS classifier. The other functions are used to take the output of the model and create performance results and switch rate analysis results and save them to file. There is also a function to smooth model results to force reduced switching

This set of scripts runs the second model (MC/MS). The first section 'create input table' much like in model 1, though it also removes MD and G from the training data to create a pure input. The next section is for spot checking different classification algorithms through the first one (KNN) is currently leading in terms of performance and switch rate analysis. However there is still a lot of scope for iterative improvement here, both in terms of model selection, parameter tuning, and ensemble approaches. Finally, the last section is an experimental for-loop I constructed to try every classification model in caret, this may not be advisable as some models take a lot of computing power to run (though it may become feasible once AWS server is running). Current best model (ensemble1_threshold 36 + KNN (smoothed) in available in the final section.

Running models (example using model 2)

```
# XGBOOST
mod1 = "xgbLinear"
para_sam1 = NULL
n = N
mseed = 82

SKttBTkNNalldata_ANN_m2  = SKhibacktestnonode_m2(trainingmx = inputtraingrowthEVAlogit_s3,predictionmx =
predvalsgrowthEVAlogit_ANN,class_method1=mod1,train_method="repeatedcv",seed=mseed,para_sampl1 = NULL,outputprobs = FALSE)

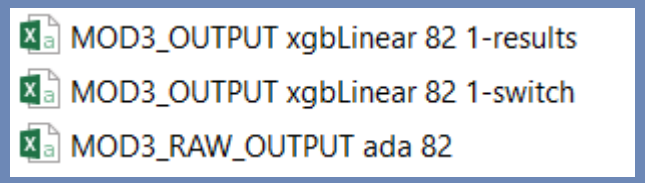
model1 = read.csv('ENSEMBLE_1_THRES36_FULLOUTPUT_GROWTHONLY.csv')
model2 = SKttBTkNNalldata_ANN_m2
write.csv(model2,paste("MOD2_RAW_OUTPUT",mod1,mseed,n,".csv"))
model3 = SKmodcombine(model1=model1,model2=model2)
write.csv(model3,paste("MOD3_RAW_OUTPUT",mod1,mseed,n,".csv"))

resultsmod2 = SKoutputs_mod2(model2)
SKoutputsave2(resultsmod2,paste("MOD2_OUTPUT",mod1,mseed,n))
resultsmod3 = SKoutputs_mod2(model3,finalrun = TRUE)
SKoutputsave2(resultsmod3,paste("MOD3_OUTPUT",mod1,mseed,n),finalrun = TRUE)
```

The code is setup to make spot checking new models very simple. Simply put in the parameters at the top here (e.g. model name)

This will call the model function and pass your parameters through it. Outputting a matrix with the classification predicted for each stock at each point in time. At the moment model 1 and model 2 are two separate functions (as requested) but these can easily be combined into a single function using a switch at a later dates.

Finally this set of codes runs a function to get performance and switch-rate analysis metrics for both model 2 alone and once combined with the best model 1, again saving the results to file. These have uniform file names and can be analysed separately or in R:



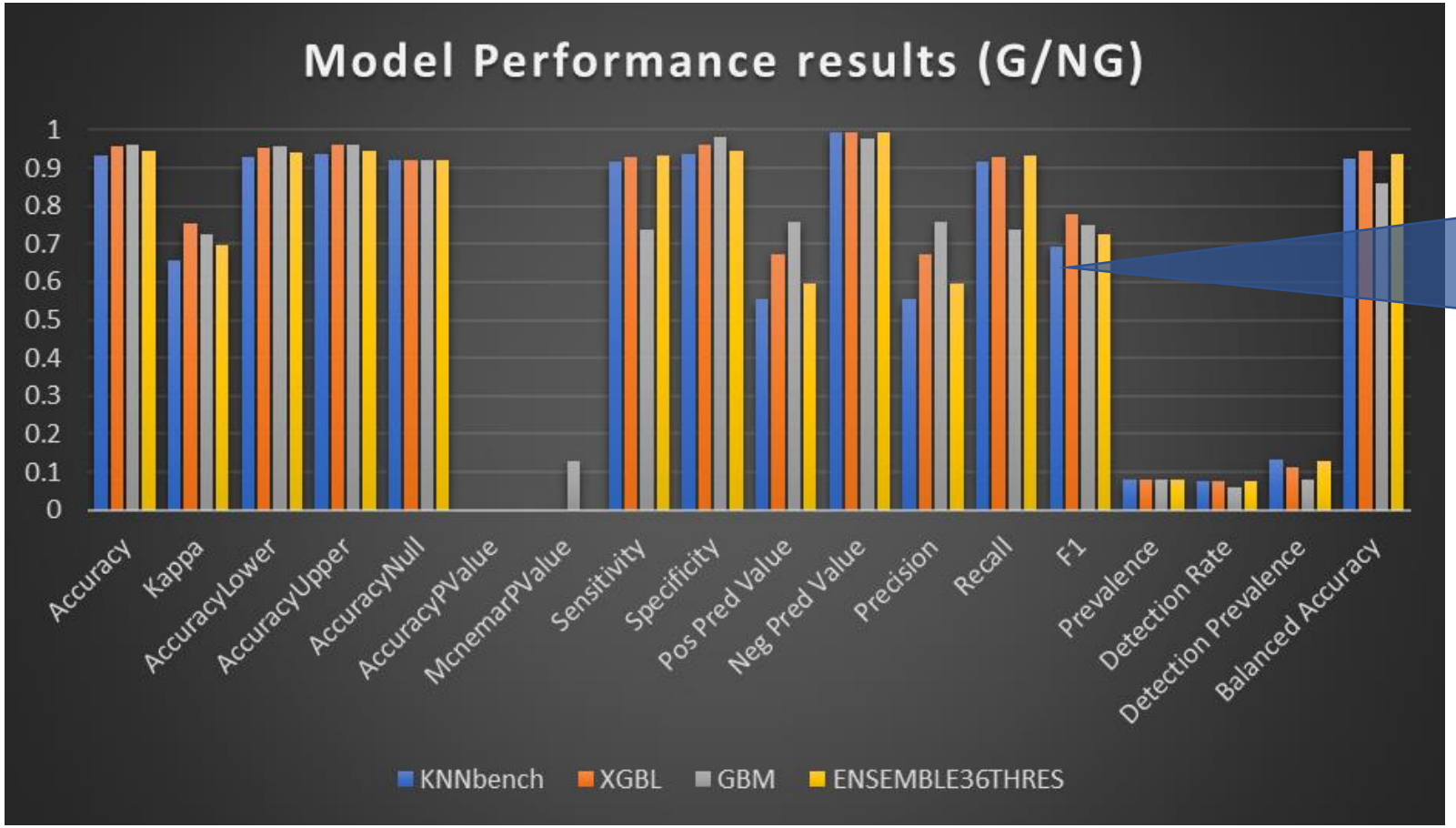
'MOD3' denotes which model run it is (1 = G/NG, 2 = MS/MC, 3 = combo), 'RAW' is the final prediction matrix – this is then followed by model type (e.g. ada), seed number (82) and percentage of training set used (1). 'Results' is for model performance results and 'switch' is for switch rate analysis, while 'dist' is for distribution analysis.

This set of code saves the raw output of the model to file, as well as combining it with the previous best model 1 (to get a final model) where the raw results are also saved. Combining with model 1 is obviously only applicable for this example using model 2.

Overall this setup should make it very to spot test new algorithms going forward as well as iterate over multiple seeds and across various input parameters. However if you want to add model specific tuning parameters you will need to edit the model function itself.

Results analysis (example using model 1)

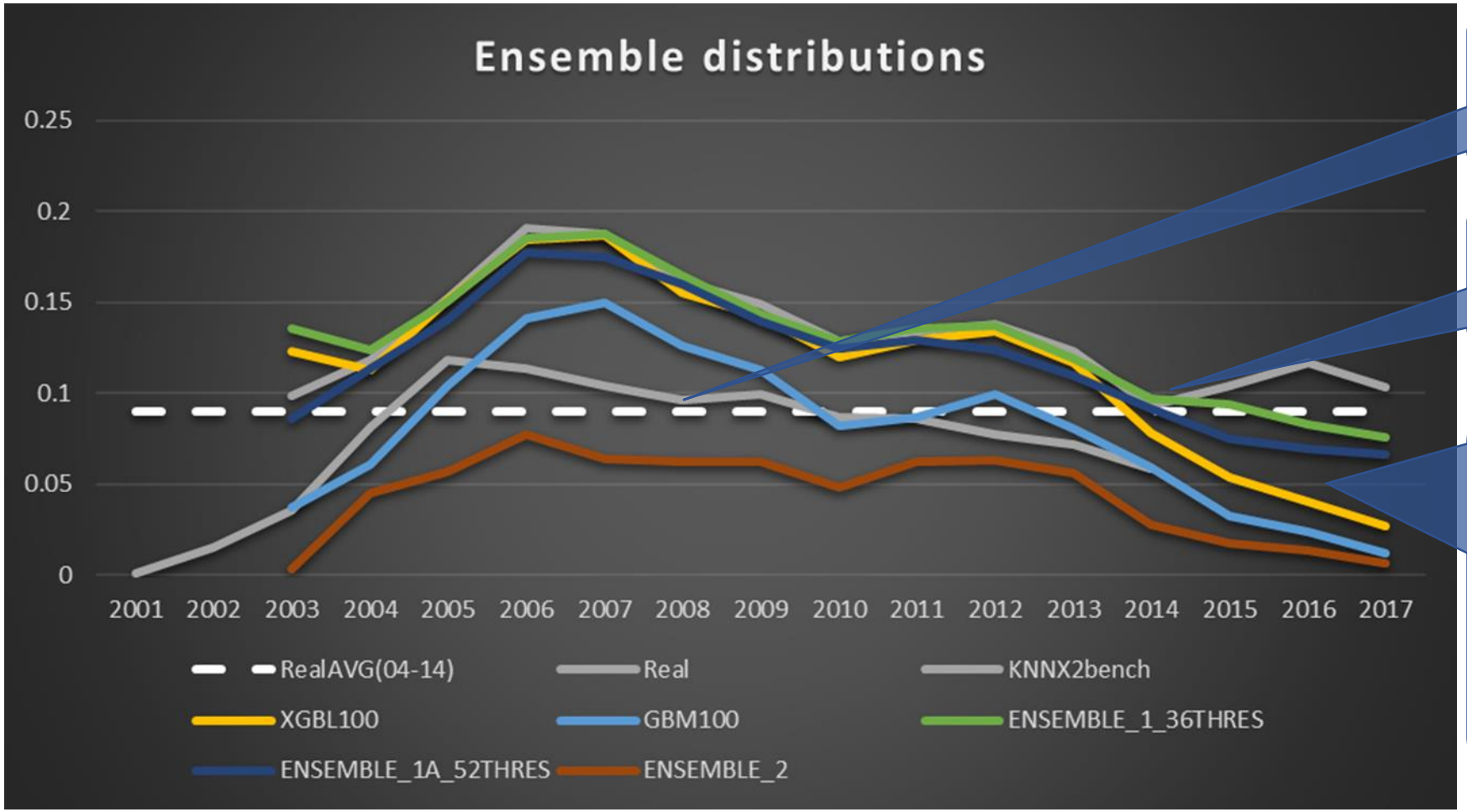
- Important to spot check performance of models against the benchmark



Import for uneven classification problems like this not just to look at accuracy but also at other pure metrics (like Recall) or blended metrics like F1 scores

Results analysis (example using model 1)

- However also important to ‘sense-check’ with distribution analysis



Before 2015 we want to see it as close to the real line as possible (without overfitting)

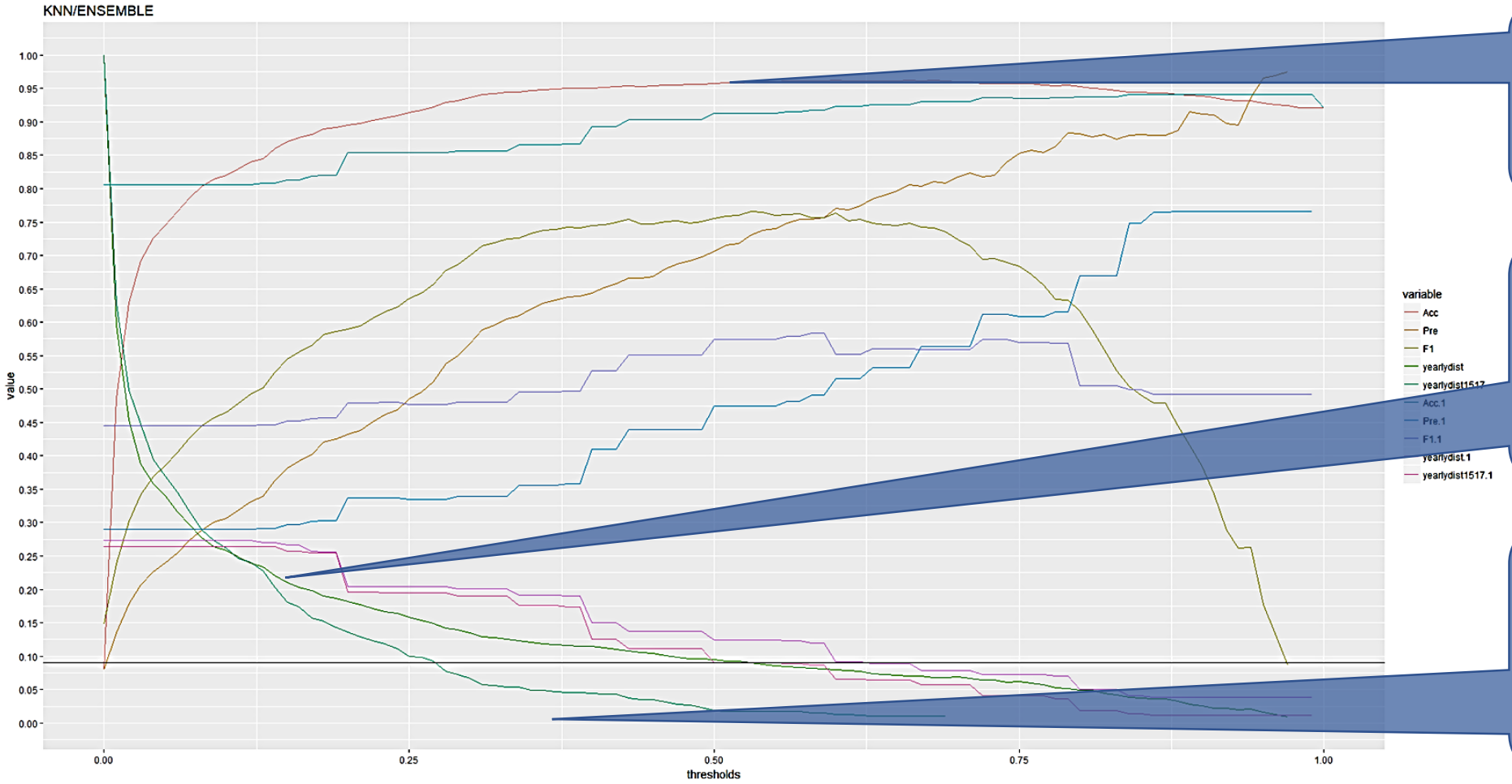
But after 2015 (out of sample, i.e. where we have no more Growth classes recorded), we want to see it not totally drop off / spike compares to before.

Though some research on water investment does suggest we should see a marginal decline in this period – though this should be counter-balanced to some extent by the growth of developing country stocks. See: https://www.cleantech.com/whats-ailing-water/

For Model 2 we may will also want to look at switch rate analysis with and without smoothing

Results analysis (example using model 1)

- As well as perform threshold analysis to optimise cut-off



Accuracy curve (want to maximise this)

Distribution curves (% of Growths per year) for both insample and out of sample data. Want this to be close to the black horizontal line (mean real distribution

Someone around 0.36 maximises performance metrics while keeping distribution within the ranges we want.

Idea for further improvement

- Model spot checking – More models could be spot checked to see if current benchmarks can be improved upon, this is especially true for model 2 (MC/MS) where
- Parameter tuning – All models used in the model 1 ensemble currently use default tuning parameters, results could be improved by experimenting/iterating these parameters for optimal results
- Ensemble modelling options – Thus far only basic ensemble methods have been used, going forward more sophisticated methods (e.g. voted stacking) could be experimented with to improve results
- Seed iteration – Before going into full production (and once AWS makes this easy) it would be useful to retest outputs over many seeds (available as input into model function) the average of these outputs should give a more precise idea of actual performance results and reduce spurious results.
- Cycle analysis – Analysis of cyclical patterns in the data (e.g. number of sales growth up/down periods) may be used to construct new variables to pull apart MC/MS.