REPORT – Shiny Application for Assessment of CARET Methods on Medical Data

A shiny app has been designed to demonstrate 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.

Steps Performed

1. Learning the caret style of thinking about model optimisation and selection.
2. Choosing appropriate feature engineering (in the right order).
3. Performing hyper-parameter optimisation through resampling.
4. Using parallel processing for methods that support this.
5. Producing a set of candidate models appropriate to the data.
6. Assessing candidate models relative to a base (null) model.
7. Selecting the best model using an unbiased measure.

1. Data description

- The data consists of 1280 observations of 24 variables. The target variable is "Y".
- There are missing values in a portion of the numeric variables. The missing values visually appear at random. There are no excessively missing variables or observations.
- The numeric predictor data has some uni-variable outliers but these disappear when the IQR multiplier reaches 2.3 so these are of little significance.
- There is only one nominal variable i.e. "BloodType" which has a low cardinality of 4.
- There are two different sets of numeric variables having similar distributions and similar variations in the means and variances.
- The predictor correlation shows that a block of numeric variables are highly correlated. These all have similar variable names i.e. "Reagent*".
- Only one date formatted character variable exists i.e. "TreatmentDate". The format is YYYY-mm-dd. These have been converted to date variables.

2. Strategies

2.1. Missing data

- There are no excessively missing variables or observations to discard.
- Since most of the values are Missing at Random, partial deletion will not be feasible to avoid bias. The methods that implicitly handle missing values can be tried on the raw data.
- For the other methods, knn imputation with default k value of 5 is employed as the standard approach.
- Since there are no missing target values, we do not need to discard these observations.

2.2. Outliers

The uni-variable outliers are of little concern. All the observations are to be retained. It is worthwhile to consider some robust and ensemble methods just in case they offer some advantage.

3. Processing

- Centering and scaling are not necessary due to the similarity in means and variances of the numeric data. Once selected, the best model was improved slightly by splitting and converting the date variable "TreatmentDate" into Years, Months and Days to use as additional predictors.
- Low cardinality nominal variables were treated with dummy encoding.
- The large number of variables present after dummy encoding can be reduced by employing methods with implicit feature selection.
- A static test set of 25% is set aside to assess the best model using stratified sampling based on the target variable.
- The hyper-parameters will be tuned by using 15 fold cross-validation resampling. The distribution of metrics have been plotted.
- Parallel processing is done on top of the models considering the computational intensity of the models to give better results.

4. Methods

The following methods were tried in addition to the pre given methods (**Null, Glmnet, Pls and Rpart**).

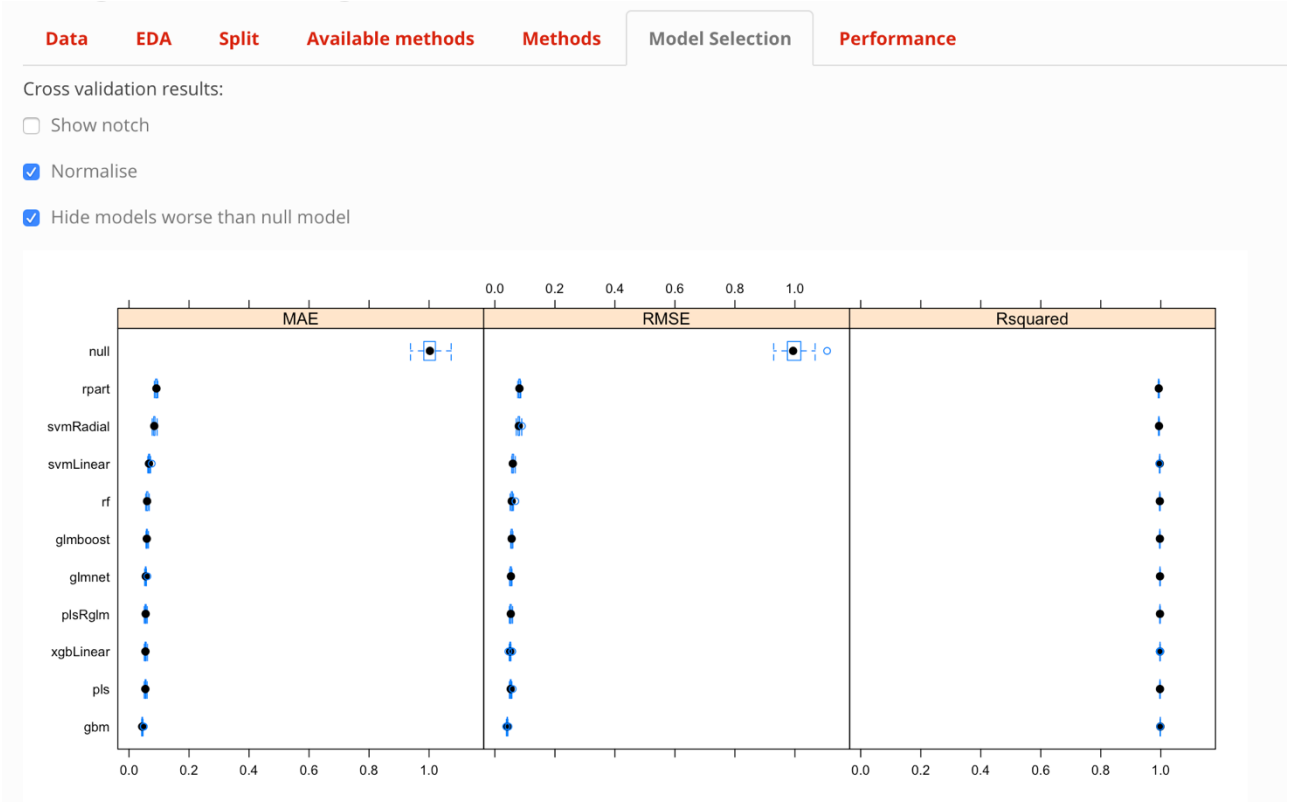
Method	Characteristics	Notes	Reason Chosen
Random Forest	Random Forest, Ensemble Model, Bagging, Implicit Feature Selection	Slow to train 1 hyperparameter	Random example of Tree based method is chosen that performs implicit feature selection.
PlsRglm	Generalized Linear Models, Partial Least Squares, Two Class Only	2 hyperparameters	Chosen to combine GLM and PLS models as both performed better individually
Boosted GLM	Generalized Linear Model, Ensemble Model, Boosting, Linear Classifier, Two Class Only, Accepts Case Weights	2 hyperparameters	Random example of generalised linear model that is an ensemble model and accepts case weights
GBM	Tree-Based Model, Boosting, Ensemble Model, Implicit Feature Selection, Accepts Case Weights	3 hyperparameters Slightly slow to train	Chosen as it includes features of previous models that performed well
XGB Linear	Linear Classifier Models, Linear Regression Models, L1 Regularization Models, L2 Regularization Models, Boosting, Ensemble Model, Implicit Feature Selection	4 hyperparameters Fast to train	Chosen as it uses regularization in addition to the features of well performed GBM model
XGB Tree	Tree-Based Model Boosting Ensemble Model Implicit Feature Selection Accepts Case Weights	7 hyperparameters Very slow to train and crashed later	Chosen to try similar to XGB linear and tree based.
ANN	Neural Network, L2 Regularization, Accepts Case Weights	2 hyperparameters Performed worse than null model	Totally random try to select a that features neural network. Also ANN features case weights
SVM Linear	Kernel Method, Support Vector Machines, Linear Regression, Linear Classifier, Robust Methods	2 hyperparameters Slow to train	Totally random try to select a random model that features kernel methods. Also it includes features of the previously successful models

SVM Radial	Kernel Method, Support Vector Machines, Radial Basis Function, Robust Methods	2 hyperparameters Slow to train	Similar to SVM Linear and features radial basis function

5. Models

The following models were successfully trained. A visual summary of the models is shown below.

Models that performed worse than the Null model are omitted.
Only ANN model performed worse than the Null model



Model	Processing steps	Resampled Performance
glmnet	knnimpute dummy	RMSE = 92.95 R2 = 1.00 MAE = 68.18
pls	date knnimpute dummy	RMSE = 92.49 R2 = 1.00 MAE = 67.25
rpart	date knnimpute dummy	RMSE = 143.28 R2 = 0.99 MAE = 110.90
rf	date knnimpute dummy	RMSE = 99.61 R2 = 1.00 MAE = 74.14

plsRglm	date knnimpute dummy	RMSE = 92.74 R2 = 1.00 MAE = 67.56
glmboost	knnimpute dummy	RMSE = 98.45 R2 = 1.00 MAE = 72.96
gbm	knnimpute dummy	RMSE = 72.43 R2 = 1.00 MAE = 54.40
xgbLinear	knnimpute dummy	RMSE = 89.7 R2 = 1.00 MAE = 67.62
nnet	knnimpute dummy	RMSE = 1800.16 R2 = 0.03 MAE = 75.27
svmLinear	scale center knnimpute dummy	RMSE = 105.89 R2 = 1.00 MAE = 82.25
svmRadial	scale center knnimpute dummy	RMSE = 140.85 R2 = 1.00 MAE = 103.80

Best model

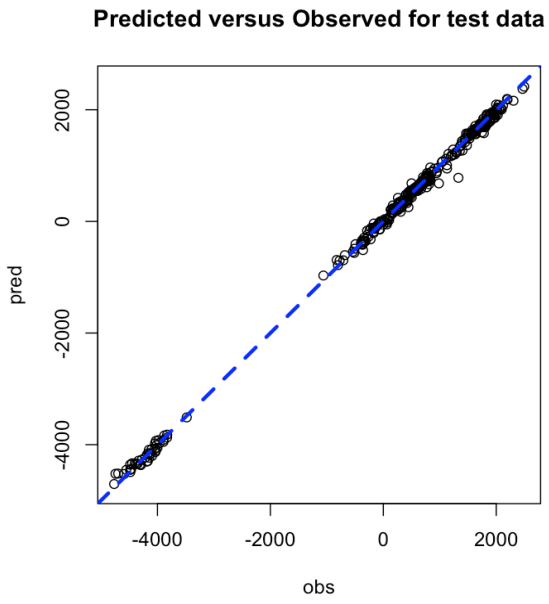
Based on the RMSE value, the best model is GBM. And the other three models PLS, PlsRglm and XGB Linear also have performed well with slightly lower RMSE values.

The models PlsRglm, PLS and XGB linear can be ensembled to obtain a better performing model.

Best performing transparent models: GBM and Glmnet models.

Performance on unseen data

The test data was predicted using the best model - GBM. It generated the following chart.

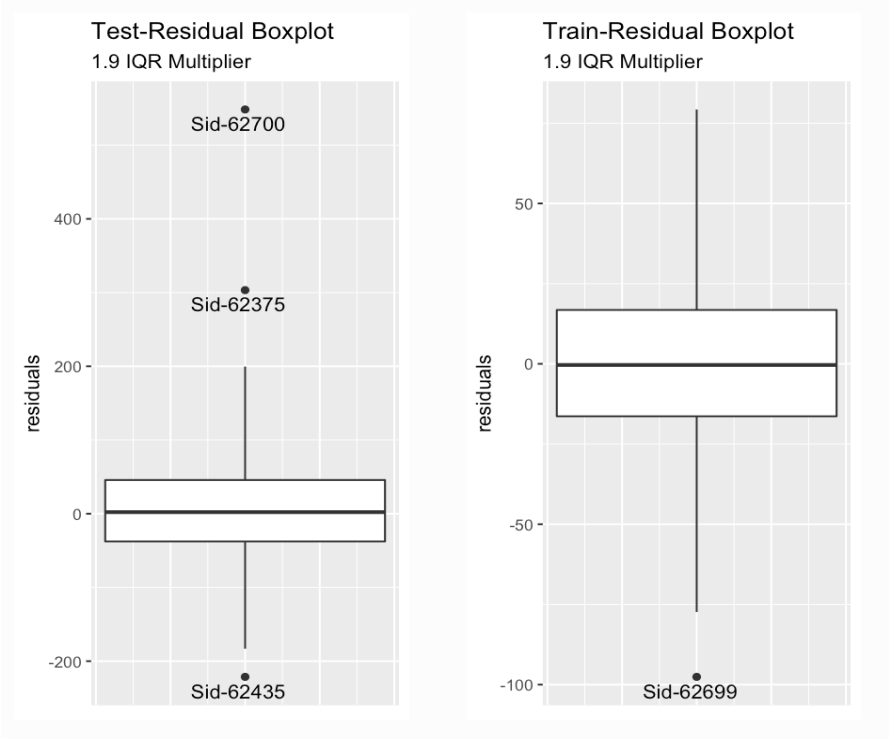


The metrics for the best model are

Test Metric	Value
RMSE	73.9055682
MAE	53.6144135
R2	0.9985287

Observation that do not fit the model

The model-based outliers (at IQR Multiplier of 1.9) are shown below:



Best Method description

The best model uses GBM – Gradient Boosting Machine method. This method seems to work well with this data because the data has a non-linear relationship with the predictors. There are sufficient observations for this method to be a good fit.

GBM is designed to be configured in a wide variety of ways. It is used to predict a continuous value in regression. In GBM, learning or boosting happens by optimising the loss functions. It improves the performance over the base algorithm with various regularization schemes. GBM contains 4 hyperparameters

- **Number of trees**: The total number of trees in the sequence or ensemble.
- **Learning rate**: Contribution of each tree on the final outcome and algorithm processing rate.
- **Tree depth**: Controls the depth of the individual trees.
- **Minimum number of observations in terminal nodes**: Controls the complexity of each tree.

Gradient boost starts by making a single leaf. It makes an initial guess for the weights of all the samples. The first guess is the average value. Then the gradient boost builds a tree. This fixed tree is based on the errors made by the previous tree. Gradient boost tries to restrict the size of the tree with a maximum number of leaves between 8 and 32. It also scales all the trees by the same amount. It continues until additional trees fail to improve the fit.