What can CARET do for you?

Derek E. Smith

August 14, 2018

Table of contents

CARET - Classification And Regression Training

- An R package for training process for complex regression and classification problems
- Places multiple R packages into a common syntax for prediction modeling

Elements of CARET

- Pre-Processing
- Data Splitting
- Model Tuning and Training
  - 237 models available
  - Random hyperparameter search
- Parallel Processing
- Class Imbalance
  - Up-sampling
  - Down-sampling
  - SMOTE (Synthetic Minority Over-Sampling Technique)
- Variable Importance

Predictive Modeling Review

Predictive Modeling Jargon

- Features, Inputs = Predictors (i.e., Independent Variables)
- Responses, Outputs = Outcomes (i.e., Dependent Variables)
- Training Set = Subset of data used to construct the prediction model
- Tuning Parameters, Hyperparameters = Algorithmic constants that can't be estimated from the data and are used to maximize model performance (e.g., number of trees in a random forest)
- Testing Set = Subset of data used to assess the prediction model performance
- Classification = Prediction of qualitative outputs
- Regression = Prediction of quantitative outputs

Model Assessment and Selection

Training (train/validate)

Model selection (Training): Evaluate the performance across different models to choose the best one

- Tuning: Assessment of multiple models in the training dataset to select the tuning and hyperparameters (e.g., number of trees, interaction depth, etc.) that minimize prediction error
- Validation: Cross-validation or bootstrap resampling to estimate prediction error

Testing

Model Assessment (Testing): After choosing a final model evaluate its prediction error/classification error on a separate data set.

- Generalization - performance of the learning method on an independent dataset

Generalization error,

$$Err_{\tau} = E[L(Y, \hat{f}(X))|\tau]$$

Where: τ is the testing set data and $L(Y, \hat{f}(X))$ is loss function to measure errors between Y and $\hat{f}(X)$.

Performance Metrics

Performance Metrics for Model Selection/Evaluation

Classification

- Accuracy/Misclassification Error
  - Limited use with rare events
- Cohen's Kappa

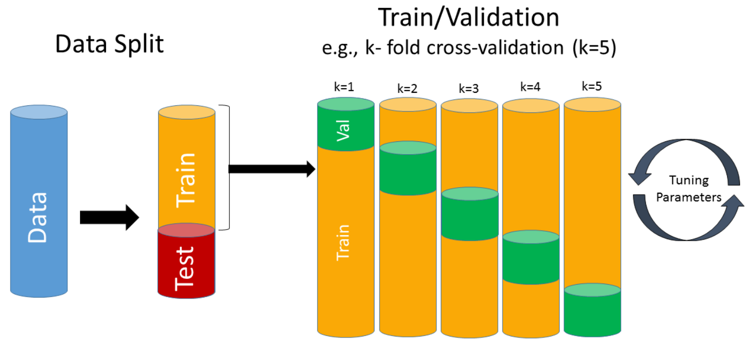
$$Kappa = \frac{A_{obs} - A_{exp}}{1 - A_{exp}}$$

- AUC

Regression

- Root Mean Square Error
- R^2
- Mean Absolute Error

Overview


Data Split
Train/Validation
e.g., k- fold cross-validation (k=5)
Data
Train
Test
k=1
k=2
k=3
k=4
k=5
Val
Train
Tuning
Parameters

Overfitting - Bias Variance Tradeoff

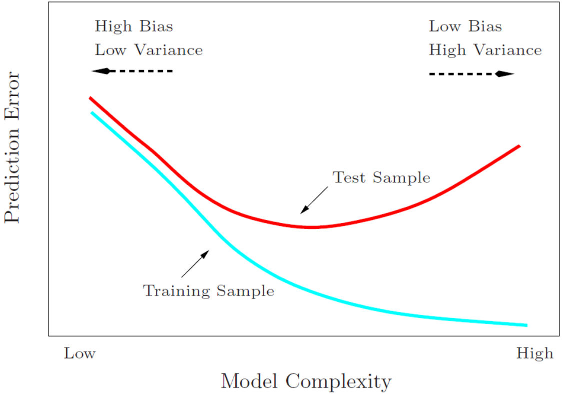


Image Source: Hastie et al. The Elements of Statistical Learning

Model Inference and Averaging

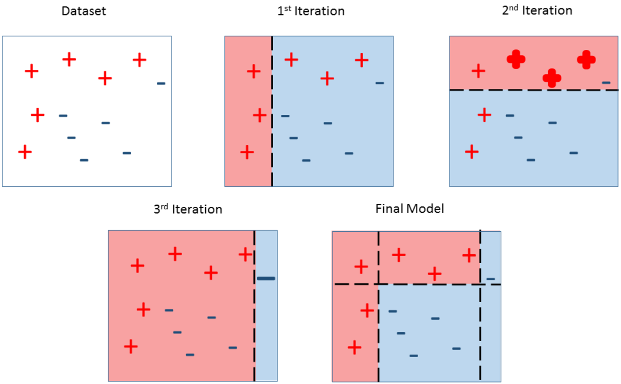
Ensemble learning methods

- Bagging (Bootstrap Aggregation)
  - Bootstrap sample the data to produce N datasets
  - Produce a model for N datasets
  - Take the average across the N models
- Random Forest
  - Similar to bagging except predictors are randomly sampled without replacement upon each iteration
- Boosting
  - An additive model where weak classifiers are combined to create a strong classifier

Ensemble learning methods

- Boosting cont.
  1. The best classifier is found
     - the sample space is partitioned such that the greatest number of samples are correctly classified
  2. Samples are weighted
     - Incorrectly classified samples are given a higher weight
     - Correctly classified samples are given a smaller weight
  3. Steps 1 and 2 are repeated a number of times
- Samples that are difficult to classify continuously receive higher weights increasing the ability for the model to classify them

Boosting Example



CARET Workflow

CARET Workflow

CARET Preprocessing

Categorical Variables must be set to factors and be syntactically valid in R.

```
## set categorical vars to factors
catVars<-c() #vector of categorical variable names

df[,catVars]<-lapply(df[,catVars],factor)

## check if the classes are correct
sapply(df,class)

## convert factor labels to valid R syntax
df[,catVars]<-lapply(df[,catVars],make.names)
```

CARET Preprocessing

findCorrelation - Identify correlated variables

```
# create a correlation matrix
descrCor<-cor(df)

# identify correlation > cutpoint
highCorVars<-findCorrelation(descrCor, cutoff = .75)

#Remove highly correlated variables from dataset
filteredDf <- df[,-highCorVars]
```

Note:

- Considers the absolute values of pair-wise correlations (i.e., 0.6 = -0.6)
- For two correlated variables it removes the one that has the highest correlation with all the other variables

CARET Preprocessing

nearZeroVar - Identify predictors that have only a handful of unique values that occur with very low frequencies

- Zero variance predictors (i.e. one unique value)
- Predictors that are have both of the following characteristics:
  1. they have very few unique values relative to the number of samples
  2. $\frac{\textit{Freq most common value}}{\textit{Freq second most common value}} > \text{threshold}$

```
-nzv<-nearZeroVar(df,
# cutoff for the ratio of the most common value to the second most common value
freqCut = 95/5,
# cutoff for the percentage of distinct values out of the number of total samples
uniqueCut = 10)

filteredDf <- df[, -nzv]
```

CARET Preprocessing

Other Preprocessing

- Centering
- Scaling
- Imputation (KNN, Bagging)
- Transformations (PCA, Box-Cox, Yeo-Johnson, Exponential)

```
preProcess(df,method = c("center", "scale","knnImpute","pca"))
```

CARET Preprocessing

createDataPartition - Create Training and Testing Datasets

```
set.seed(142)

# create a vector of row numbers to split
# the dataset into a training and test set

inTraining <- createDataPartition(df$outcome,
p = 2/3, # percentage of data the training set will use
list = FALSE) # return a list or matrix


training <- df[ inTraining,] # create the training set
testing  <- df[-inTraining,] # create the test set
```

CARET Training

Primary Functions

trainControl - Control parameters for model training

train - Algorithm selection and reporting metrics

Primary Arguments

```
fitControl <- trainControl(
method = "repeatedcv", # resampling method
number = 10, # resampling iterations or folds for CV
repeats=3, # number of CV repeats
classProbs = TRUE, # Return class probabilities
summaryFunction = twoClassSummary, # Evaluate model perfromance
allowParallel=TRUE, # allow parallel processing
seeds=seeds) # provide a list of seeds for parallel processing
```

Note: Full Argument is Algorithm Specific (e.g. ntree = number of trees in a random forest)

CARET Training

```
modFit<- train(outcome ~ ., data = training, ## training dataset
method = "rf", ## random forest method
trControl = fitControl, ## pass the tuning parameters
importance = TRUE, ## include variable importance
na.action = na.pass, ## How to treat missing values
ntree=150, ## set number of trees to be grown in a RF
metric = "ROC",## Select model based on optimizing the AUC
verbose = FALSE)
```

CARET Training

expand.grid - Expand the tuning parameters

```
gbmGrid <-  expand.grid(interaction.depth = c(1, 5, 9),
n.trees = (1:30)*50,
shrinkage = 0.1,
n.minobsinnode = 20)
```

```
set.seed(142)
modFit <- train(outcome ~ ., data = training,
method = "gbm",
trControl = fitControl,
verbose = FALSE,
tuneGrid = gbmGrid)
```

CARET Testing

predict - Predict the outcome for a given model

```
## Obtain the predictions usign the test set
predictors<-names(testing)[names(testing) !="outcome"]

## Predicting probabilities
# type="raw" for classes and "prob" for probabilities
pred <- predict(modFit, newdata=testing[,predictors],type="prob")

## determine the AUC from the predictions
out<-auc(response=testing$outcome,predictor=pred$Yes)
```

CARET Other Features

Class Imbalance

- Up-sampling
- Down-sampling
- SMOTE

Variable importance in CARET

Resources

- CARET Overview in Markdown
- Max Kuhn and Kjell Johnson (2013) Applied Predictive Modeling. Springer-Verlag New York. DOI: 10.1007/978-1-4614-6849-3
- Hastie T. Tibshirani R. and Friedman J. (2008) The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer-Verlag New York. DOI: 10.1007/978-0-387-84858-7
- Ewout Steyerberg (2009) Clinical Prediction Models A Practical Approach to Development, Validation, and Updating. Springer-Verlag New York. DOI: 10.1007/978-0-387-77244-8

Example

Aim: Can we predict whether an individual diagnosed with Diffuse large B-cell lymphoma (DLBCL) will develop a secondary primary malignancy (SPM)?

Data: SEER (Surveillance, Epidemiology, and End Results Program) 1973 to 2010.

- Models:
  - Random Forest
  - Boosted Trees
  - Linear Discriminant Analysis
  - Bagged Flexible Discriminant Analysis
- Training Parameters:
  - Default tuning parameters used
  - AUC used for model evaluation
- Data: N = 26,038, Split (2/3), 13% event rate
  - Train N = 17,359 (Cases = 2,252; Controls = 15,107)
  - Test N = 8,679 (Cases = 1,126; Controls = 7,553)