BIOST 546 Final Project Report

My-Anh Doan

2023-03-08

Abstract

Summarize the major aspects of the report as well as mention the following:

- The statistical question being answered
- The data analysis methods employed
- General findings or trends as a result of the analysis

Introduction

This report applies machine learning tools to discriminate between patients with Alzheimer's Outcome (AD) from healthy patients (C) by analyzing thickness measurements of their cerebral cortex.

The training data set contains cerebral cortex thickness measurements at $p = 360$ different brain regions of interest from $n = 400$ patients. The data set aims to answer the question of **whether the cerebral cortex thickness measurements can be used to predict AD diagnosis** in the unseen test data set (which also contains cerebral cortex thickness measurements at $p = 360$ brain regions of interest from $n = 400$ different patients). **Table 1** shows the number of observations in each outcome class in the provided training data set.

Table 1: Outcome observation counts (training set)

Outcome	n
C	97
AD	303

Data analysis

The original provided training data set was split 80:20 to generate training and test data sets (`train_set` and `test_set`, respectively) which are used for model fitting and model performance evaluation.

```r
# split data into train/test set ----
set.seed(2)
index <- sample(1:n, size = 0.8*n, replace = FALSE)

train_set <- full[index, ]
test_set <- full[-index, ]
```

The following models were then fitted to the `train_set` subset as shown below:

- Simple glm Logistic Regression
- Ridge Logistic Regression
- Lasso Logistic Regression
- Decision Tree (pre- and post-pruning)
- Bagged Tree
- Random Forest
- Boosted Tree

The data analysis began with fitting a simple glm logistic regression model to the training data. Ridge and lasso logistic regression models were then used to reduce the number of coefficients in the resulting model to reduce variance in the model and improve prediction accuracy.

Decision tree algorithms were looked at given the high-dimensional data this data set. A simple classification tree was fitted without pruning first and then pruned. Bagging, Random Forest, and boosting tree algorithms were also looked at as methods for tree models that might be overfitted to the training data.

```r
set.seed(2)
# fit a simple glm model ----
glm_model <- glm(formula = Outcome ~ ., data = train_set,
                 family = binomial(link = "logit"))

# fit a ridge logistic model ----
# obtain optimal lambda value for ridge model
ridge_cv <- cv.glmnet(train_x_scaled, train_y, lambda = lambda_grid,
                      alpha = 0, nfolds = 10, family = "binomial",
                      type.measure = "class")
ridge_lambda <- ridge_cv$lambda.min

ridge_model <- glmnet(train_x_scaled, train_y,
                      lambda = ridge_lambda,
                      alpha = 0, family = "binomial")

# fit a lasso logistic model ----
# obtain optimal lambda value for lasso model
lasso_cv <- cv.glmnet(train_x_scaled, train_y, lambda = lambda_grid,
                      alpha = 1, nfolds = 10, family = "binomial",
```

```r
                      type.measure = "class")
lasso_lambda <- lasso_cv$lambda.min

lasso_model <- glmnet(train_x_scaled, train_y,
                      lambda = lasso_lambda,
                      alpha = 1, family = "binomial")

# fit a simple classification tree without pruning ----
overgrown_tree <- tree(Outcome ~ ., train_set)

# fit a simple classification tree with pruning ----
# obtain subtree size that minimizes the CV misclassification error
cv_tree <- cv.tree(overgrown_tree, FUN = prune.misclass)
subtree_size <- cv_tree$size[which(cv_tree$dev == min(cv_tree$dev))]

pruned_tree <- prune.tree(overgrown_tree, best = subtree_size)

# fit a bagged tree model ----
bagged_model <- randomForest(Outcome ~ ., data = train_set,
                             mtry = p, importance = TRUE)

# fit a random forest model ----
rf_model <- randomForest(Outcome ~ ., data = train_set,
                         mtry = p/3, importance = TRUE)

# fit a boosted trees model ----
boosted_model <- gbm(Outcome ~ ., data = train_set_num,
                     distribution = "bernoulli",
                     n.trees = 500, interaction.depth = 2, shrinkage = 0.1)
```

After fitting the previously mentioned models to the training data `train_set`, the models were used to predict `Outcome` classes on the test data `test_set`.

The training and test misclassification errors were calculated for each model (**Table 2**).

Table 2: Model Performance

Model	Training MSE	Test MSE
Simple glm	0.000	0.500
Ridge	0.028	0.050
Lasso	0.028	0.100
Overgrown Tree	0.022	0.312
Pruned Tree	0.156	0.262
Bagged Tree	0.000	0.188
Random Forest	0.000	0.150
Boosted Tree	0.000	0.064

Results and Conclusions

Table 2 shows that of the eight fitted models, the simple glm model, bagged tree model, random forest model, and boosted tree model had the lowest training misclassification error values (0) while the prune decision tree had the greatest training misclassification error (0.156). This is not a surprising result given that training misclassification error decreases as model complexity increases.

From the four models with the lowest training MSE, the boosted tree model had the smallest test misclassification error (0.064) while the simple glm model had the greatest test misclassification error (0.5). Here, we can conclude that the boosted tree model is the best performing model out of the eight models.

The fitted boosted tree model was applied to the blinded test set `X_test`, which contains contains cerebral cortex thickness measurements at $p = 360$ brain regions from $n = 400$ patients. The blinded test set does not contain Outcome class labels like the original provided training set. The results are summarized in **Table 3** below.

Table 3: Boosted Tree Blinded Test Outcomes

Outcome	n
AD	315
C	85

At four different time points of this analysis, blinded predictions were submitted to evaluate different model accuracy. Results are summarized in **Table 4** below.

Table 4: Blinded Predictions Models and Accuracy

Model	Blinded Pred. Accuracy (%)
Simple glm	71.25
Ridge	68.00
Random Forest	66.50
Boosted Tree	NA