Models for discussion in report

Timothy Hedspeth

```
# This Rmarkdown will be the primary file in terms of modeling with for our
# project, this file contains the model fitting for our 4 models of interest
# risk score models and classification trees for Salmonella and E. Coli


##~~~~~~~~~~~~~~~~~~~~~~~##
#### Data and packages ####
##~~~~~~~~~~~~~~~~~~~~~~~##


rm(list=ls())

library(kableExtra)
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v ggplot2 3.4.0     v purrr   0.3.4
## v tibble  3.1.6     v dplyr   1.0.8
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter()     masks stats::filter()
## x dplyr::group_rows() masks kableExtra::group_rows()
## x dplyr::lag()        masks stats::lag()
```

```
library(ggpubr)
library(mice)
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
##
## Attaching package: 'Matrix'
```

```
## The following objects are masked from 'package:tidyr':
```

```
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-4
```

```
library(bestglm)
```

```
## Loading required package: leaps
```

```
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(DescTools)
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'
```

```
## The following objects are masked from 'package:DescTools':
##
##     MAE, RMSE
```

```
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(kableExtra)
library(ggpubr)
library(rpart)
library(rpart.plot)

setwd("~/Desktop/Semester_3/Practical/Final") # Where all our data is
salmonella <- read.csv("final_salmonella.csv")
ecoli <- read.csv("final_ecoli.csv")

# Functions required for this analysis (available on our Github)
source("functions_outbreaks_and_lasso.R")
source("compress_levels.R")
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
##
## Attaching package: 'rlang'
```

```
## The following objects are masked from 'package:purrr':
##
```

```
##     %@%, as_function, flatten, flatten_chr, flatten_dbl, flatten_int,
##     flatten_lgl, flatten_raw, invoke, splice
```

Section 1; Salmonella

We decided to first examine the bacteria Salmonella. Recall that our interest lays in predicting if a case is part of an outbreak, in the case we define an outbreak to be **8 or more cases** from the same SNP.cluster in a given month and year. We first remove all NA values, as we discuss in our EDA that the degree of missingness does not allow for us to impute the data and trust the results. After defining an outbreak we compress the levels of the SNP.cluster to be the top 7 most observed across our time frame, and create our test train splits. In other versions of this work we included interaction terms though the results were found to not be as interpretable as our additive models, an important aspect as we do hope that our models could be used as tools for individuals in the public health sphere that do not have extensive statistical training.

```
#~~~~~~~~~~~~~~~~~~#
## Clean the data ##
#~~~~~~~~~~~~~~~~~~#


salmonella <- salmonella %>% filter(!is.na(month) & !is.na(year) &
                                      !is.na(Min.same)  & !is.na(Serovar) &
                                      !is.na(Isolation.type)  & !is.na(AMR.genotypes) &
                                      !is.na(Computed.types) & !is.na(SNP.cluster)) %>%
  dplyr::select(-c(Min.diff, Location))

# We will add an outbreak to the model
salmonella <- add_outbreak(salmonella, 8)
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
# Compress the levels of SNP clusters to be the top overall
salmonella$SNP.cluster <- compress_levels_factors(salmonella, 'SNP.cluster')


## Create a test and train split
salmonella_train <- salmonella %>% filter(year != 2021 & year !=  2019 &
                                            year != 2020 & year !=  2022)
salmonella_test <- salmonella  %>%  filter(year == 2021 | year ==  2019 )
# table(salmonella_test$outbreak)
table(salmonella_train$outbreak)
```

```
##
##     0     1
## 11559  2084
```

```
Percent_outbreak_train <- 2084/(2084+11559)
Percent_outbreak_test <- 332/(332+2074)

# Get rid of variables that we won't use in analysis
salmonella_train <- salmonella_train %>% dplyr::select(-c(year, Min.same))
salmonella_test <- salmonella_test %>% dplyr::select(-c(year, Min.same))

# Make the variables factors
salmonella_train[] <- lapply(salmonella_train, function(x){return(as.factor(x))})
```

```r
salmonella_test[] <- lapply(salmonella_test, function(x){return(as.factor(x))})
```

We will first fit our Lasso Regularized Logistic Regression as we include many factor variables that all have 8 or more levels to predict if a case is part of an outbreak using biological, spatial and time of the year. Lasso aided in our model selection procedure as coefficients are snapped to 0 when they are weakly associated with the outcome. These coefficients were divided by the median non-zero coefficients and rounded to get the corresponding risk score coefficients.

```r
##~~~~~~~~~~~~~~~~~~~~~~~~~~~~##
#### Salmonella, Risk Score ####
##~~~~~~~~~~~~~~~~~~~~~~~~~~~~##

# Perform Lasso
salmonella_coef <- lasso(salmonella_train, 12)
salmonella_coef <- as.data.frame(as.matrix(salmonella_coef))
sal_coef <- as.vector(salmonella_coef$s1)

# Divide the Lasso
salmonella_coefs  <- round(salmonella_coef$s1/median(sal_coef[sal_coef != 0]))
salmonella_coef$s1 <- salmonella_coefs


## Some coefficients that we trained the data on will not be observed in the
## test set so we will need to add these to be able to assess how the
## model perform on testing data


# We note that the factors relating to SNP and AMR are the issue as
# some of the levels are not observed, so we will add them now so the
# score can be calculated

# SNP cluster
levels_snp_train <- as.list(levels(salmonella_train$SNP.cluster))
salmonella_test$SNP.cluster <- factor(salmonella_test$SNP.cluster,
                                      levels = levels_snp_train)
salmonella_test$SNP.cluster <- relevel(salmonella_test$SNP.cluster, "Other")

# AMR genotypes
levels_AMR_train <- as.list(levels(salmonella_train$AMR.genotypes))
salmonella_test$AMR.genotypes <- factor(salmonella_test$AMR.genotypes,
                                        levels = levels_AMR_train)
salmonella_test$AMR.genotypes <- relevel(salmonella_test$AMR.genotypes, "aadA1=COMPLETE,gyrA_D87Y=POINT


# We need to make sure that all levels of the factor are accounted for
dim(model.matrix(outbreak~., salmonella_test))
```

```
## [1] 1406   50
```

```r
dim(model.matrix(outbreak~., salmonella_train))
```

```
## [1] 13643    50
```

```r
## Create and display a table of Non-Zero coefficients

sal_coefs_table <- salmonella_coef %>%
```

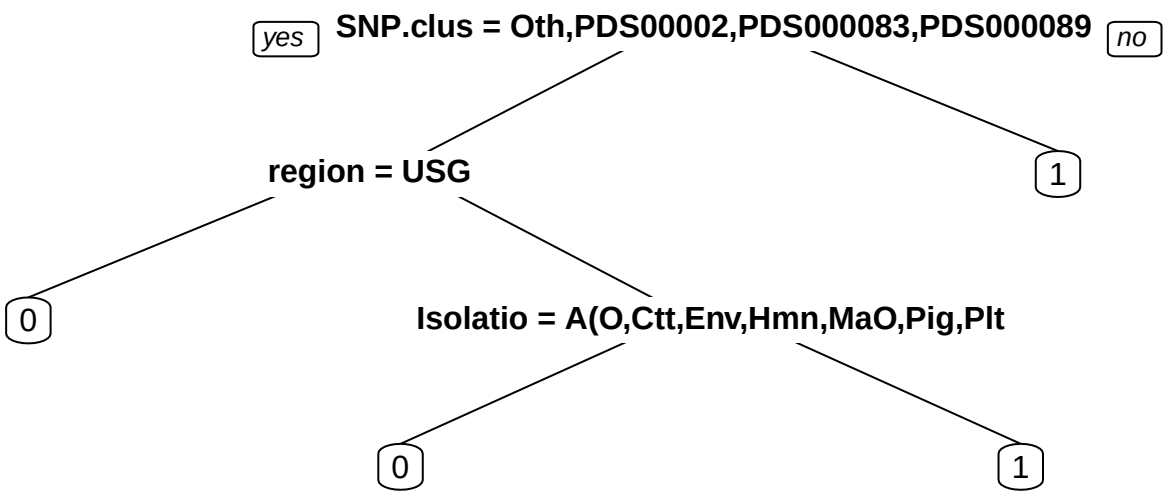
```
                    filter(s1 != 0) %>%
                    dplyr::rename(Score =  s1)

sal_coefs_table %>%  kbl(caption = "Score coefficents",booktabs=T, escape=T, align = "c") %>%
                      kable_styling(full_width = FALSE, latex_options = c('hold_position'))
```

Table 1: Score coefficents

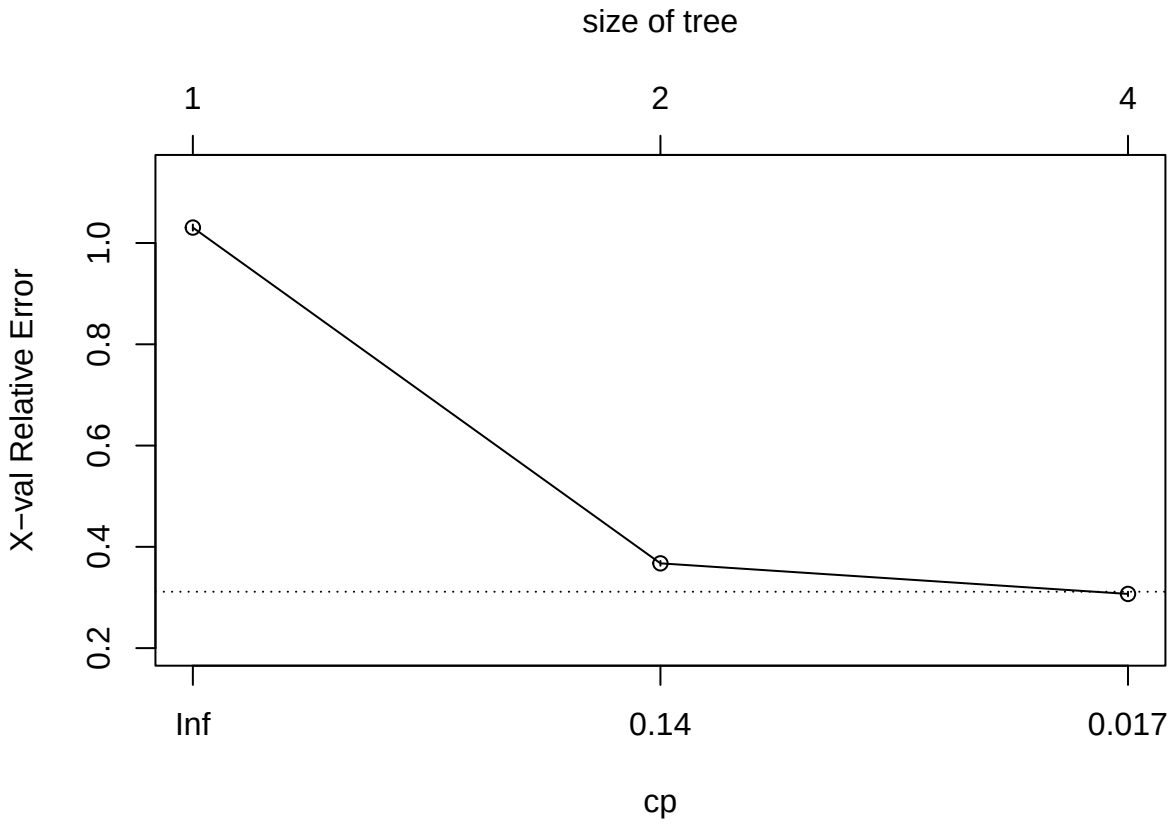
	Score
SNP.clusterPDS000030237.975	3
SNP.clusterPDS000032668.817	1
SNP.clusterPDS000120941.3	1
Computed.typesantigen_formula=9:g,m:-,serotype=Enteritidis	1
Isolation.source.categoryProduce and Food	1
regionUSA, General	-2

We are able to see in table X that of the variable the only variable that has more than one level with non-zero scores are the SNP cluster, and the SNP cluster shown in the first line has a score of 3, the highest of any in magnitude regardless of sign. Following this we note that when we do not know the exact region of the country that the outbreak was identified this detracts from the score. The other two SNP clusters, a computed type, and having the sample derived from produce or food, all increase the risk of the case being in an outbreak by 1 point.

```
##~~~~~~~~~~~~~~~~~~~~~~##
#### Salmonella Tree ####
##~~~~~~~~~~~~~~~~~~~~~~##


# We need to weight the tree (its done automatically in lasso)
prop_zero <- 1/(sum(salmonella_train$outbreak == 0)/nrow(salmonella_train))
prop_one <- 1/(sum(salmonella_train$outbreak == 1)/nrow(salmonella_train))

sal_weights <- ifelse(salmonella_train$outbreak == 0, prop_zero, prop_one)


# Fit the tree
tree_salmonella <- rpart(outbreak ~ ., data = salmonella_train,
                         method = "class", weights = sal_weights,
                         control=rpart.control(minsplit = 20,
                                               minbucket = 30))

# Plot the tree
prp(tree_salmonella,
    main = "Classification Tree, E. Coli")
```




```
# Looking  at complexity
plotcp(tree_salmonella)
```




```
# The tree already is meeting the standards of minimizing complexity
# we will not further prune it
```

We will now look at model diagnostics

```
## Diagnostics; salmonella  ##


# Get the risk scores in the testing data
variables_sal <- model.matrix(outbreak~., salmonella_test)

salmonella_test$score <- variables_sal[,-1] %*% salmonella_coefs[-1]


mod_salmonella <- glm(outbreak ~ score, data=salmonella_test, family = quasibinomial())

# For confusion matrices
salmonella_test$predicted <- predict(mod_salmonella, type = "response")

risk_perfomance_salmonella <- roc(outbreak ~ predicted, data=salmonella_test)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
sal_risk_AUC <-auc(risk_perfomance_salmonella)
sal_risk_Brier <- BrierScore(mod_salmonella)

#ggroc(risk_perfomance_salmonella) +
 # theme_minimal() +
  #geom_abline(intercept = 1, slope = 1,
   #           color = "darkgrey", linetype = "dashed")
threshhold <- .5
confusionMatrix(as.factor(ifelse(salmonella_test$predicted > threshhold,
                                 1,0)), salmonella_test$outbreak)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1068  332
##          1    6    0
##
##                Accuracy : 0.7596
##                  95% CI : (0.7364, 0.7817)
##     No Information Rate : 0.7639
##     P-Value [Acc > NIR] : 0.6601
##
##                   Kappa : -0.0085
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 0.9944
##             Specificity : 0.0000
##          Pos Pred Value : 0.7629
##          Neg Pred Value : 0.0000
##              Prevalence : 0.7639
##          Detection Rate : 0.7596
##    Detection Prevalence : 0.9957
##       Balanced Accuracy : 0.4972
```

```
##
##        'Positive' Class : 0
##
```

```
salmonella_test$score <- as.numeric(salmonella_test[,10])
salmonella_test$outbreak_1 <- case_when(salmonella_test$outbreak == "0" ~ 0,
                                        salmonella_test$outbreak == "1" ~ 1)
salmonella_test  %>% group_by(score) %>%
                     dplyr::summarize(pct = sum(outbreak_1)/n()) %>%
                     dplyr::rename("Percent outbreaks" = pct) %>%
                     kbl(caption = "Percent of cases at each score that belong to an outbreak",
                         booktabs=T, escape=T, align = "c") %>%
                     kable_styling(full_width = FALSE, latex_options = c('hold_position'))
```

Table 2: Percent of cases at each score that belong to an outbreak

score	Percent outbreaks
-2	0.0000000
0	0.2641209
1	0.0000000
2	0.0000000
3	0.0000000
4	0.0000000

```
sal_train <- model.matrix(outbreak~., salmonella_train)
salmonella_train$scores <- sal_train[,-1] %*% salmonella_coefs[-1]
salmonella_train$score <- as.numeric(salmonella_train[,10])
salmonella_train$outbreak_1 <- case_when(salmonella_train$outbreak == "0" ~ 0,
                                         salmonella_train$outbreak == "1" ~ 1)

salmonella_train %>% group_by(score) %>%
                     dplyr::summarize(pct = sum(outbreak_1)/n()) %>%
                     dplyr::rename("Percent outbreaks" = pct) %>%
                     kbl(caption = "Percent of cases at each score that belong to an outbreak",
                         booktabs=T, escape=T, align = "c") %>%
                     kable_styling(full_width = FALSE, latex_options = c('hold_position'))
```

Table 3: Percent of cases at each score that belong to an outbreak

score	Percent outbreaks
-2	0.0091848
-1	0.0918891
0	0.1514212
1	0.3162544
2	0.5801839
3	0.8000000
4	0.9142857
5	1.0000000

```
### Tree ###

preds = predict(tree_salmonella, salmonella_train, type = "class")
sensitivity(preds, salmonella_train$outbreak)
```

```
## [1] 0.86331
```

```
specificity(preds, salmonella_train$outbreak)
```

```
## [1] 0.8296545
```

```
print("predicted")
```

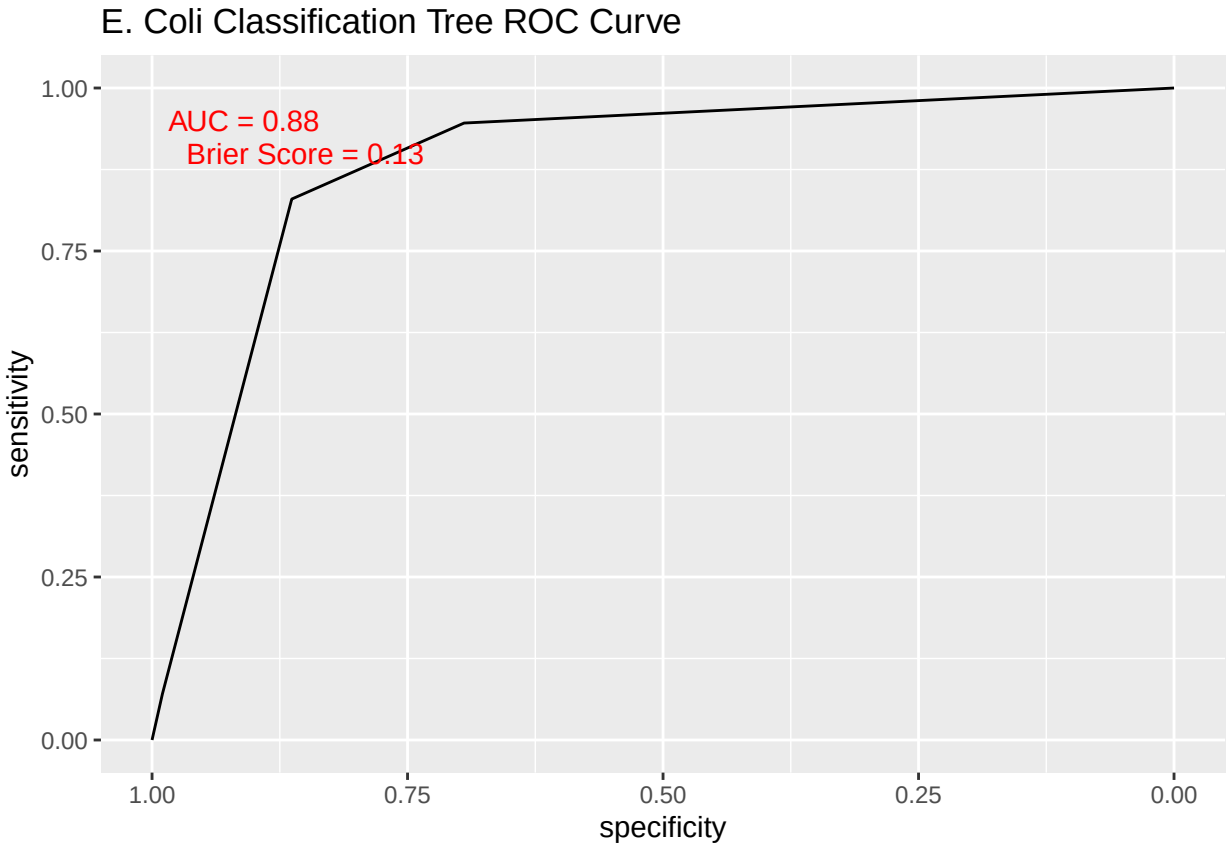
```
## [1] "predicted"
```

```
# Predict with Tree
predict_salmonella_tree <- as.data.frame(predict(tree_salmonella, salmonella_train))
names(predict_salmonella_tree) <- c("prob_tree")
salmonella_append <- cbind(salmonella_train, predict_salmonella_tree)

# ROC Analysis
salmonella_acc <- roc(outbreak ~ prob_tree, data =salmonella_append)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls > cases
```

```
#AUC
salmonella_AUC = round(auc(salmonella_acc), 2)

# Brier Score
salmonella_Brscr = round(sum((predict_salmonella_tree[,2] - (as.numeric(salmonella_train$outbreak) - 1))

ggroc(salmonella_acc) + ggtitle('E. Coli Classification Tree ROC Curve') +
  annotate("text", x = .91, y = .95, label = paste0('AUC = ', salmonella_AUC), color = 'red') +
  annotate("text", x = .85, y = .9, label = paste0('Brier Score = ', salmonella_Brscr), color = 'red')
```




```
# Predict on Test Set
pred_test = predict(tree_salmonella, newdata = salmonella_test, type = "class")
sensitivity(pred_test, salmonella_test$outbreak)
```

```
## [1] 0.9134078
```

```
specificity(pred_test, salmonella_test$outbreak)
```

```
## [1] 0
```

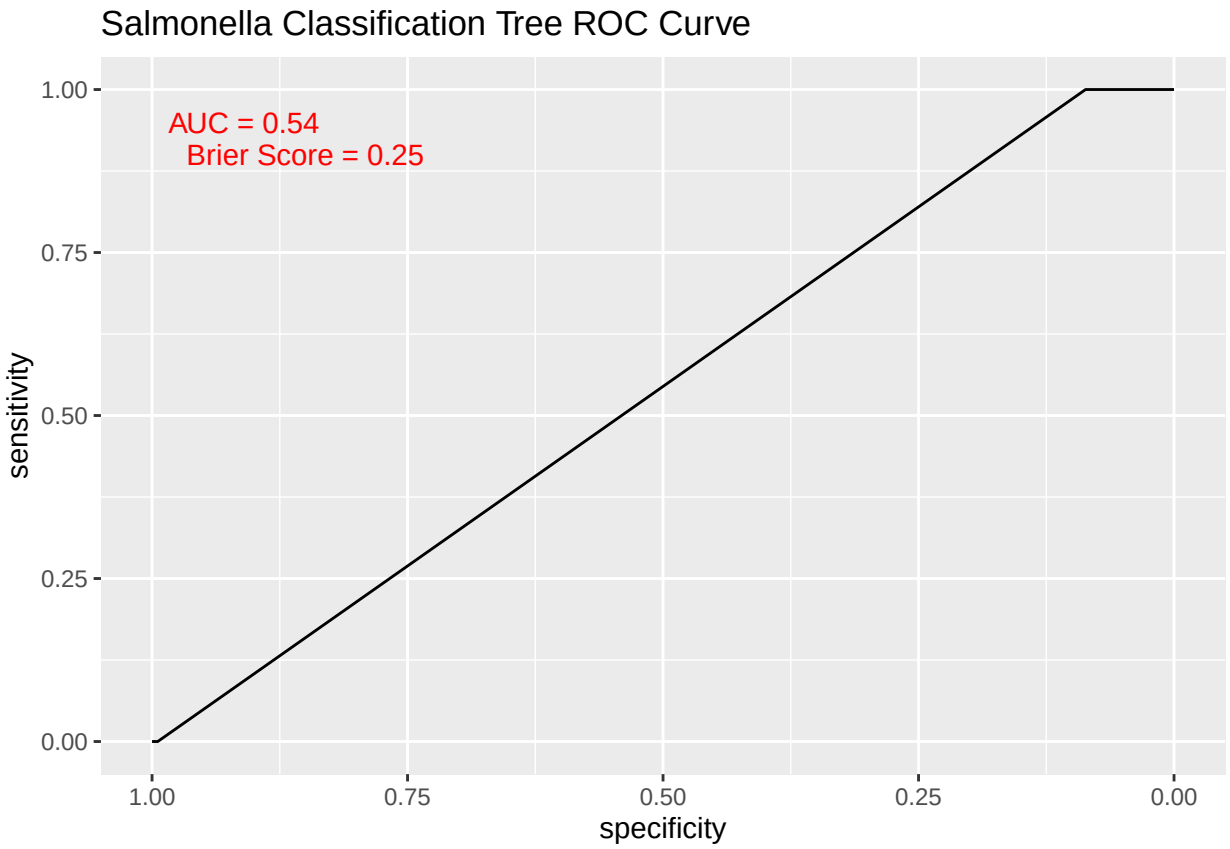
```
# Predict with Tree
predict_test_tree <- as.data.frame(predict(tree_salmonella, salmonella_test))
names(predict_test_tree) <- c("prob_tree")
salmonella_test_append <- cbind(salmonella_test, predict_test_tree)

# ROC Analysis
salmonella_test_acc <- roc(outbreak ~ prob_tree, data =salmonella_test_append)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

```
#AUC
salmonella_test_AUC = round(auc(salmonella_test_acc), 2)

# Brier Score
salmonella_test_Brscr = round(sum((predict_test_tree[,2] - (as.numeric(salmonella_test$outbreak) - 1))^

ggroc(salmonella_test_acc) + ggtitle('Salmonella Classification Tree ROC Curve') +
  annotate("text", x = .91, y = .95, label = paste0('AUC = ', salmonella_test_AUC), color = 'red') +
```

```
  annotate("text", x = .85, y = .9, label = paste0('Brier Score = ', salmonella_test_Brscr), color = 'r
```



E. coli

```
##~~~~~~~~~~~~~~~~~~~~~~~~~##
#### E. Coli Risk Scores ####
##~~~~~~~~~~~~~~~~~~~~~~~~~##

# Read in the ecoli data
ecoli <- read.csv("final_ecoli.csv")



## We will do a complete case analysis ##
ecoli <- ecoli %>% filter(!is.na(month) & !is.na(year) &
                            !is.na(SNP.cluster)  & !is.na(Strain) &
                            !is.na(AMR.genotypes) & !is.na(Location)&
                            !is.na(Isolation.type) & !is.na(region) &
                            !is.na(Isolation.source.category)) %>%
  dplyr::select(-c(day, Min.same, Location))



# We will add an outbreak to the model
ecoli <- add_outbreak(ecoli, 10)
```

```
## `summarise()` has grouped output by 'year', 'month'. You can override using the
## `.groups` argument.
```

```
# Compress the SNP Cluster levels
ecoli$SNP.cluster <- compress_levels_factors(ecoli, 'SNP.cluster')

## Get test and train splits ##
ecoli_train <- ecoli %>% filter(year != 2021 & year !=  2019 )
ecoli_test <- ecoli  %>%  filter(year == 2021 | year ==  2019 |
                                   year == 2020 | year ==  2022)
table(ecoli_test$outbreak)
```

```
##
##    0    1
## 1811  175
```

```
ecoli_train <- ecoli_train %>% dplyr::select(-c(year))
ecoli_test <- ecoli_test %>% dplyr::select(-c(year))


ecoli_train[] <- lapply(ecoli_train, function(x){return(as.factor(x))})
ecoli_test[] <- lapply(ecoli_test, function(x){return(as.factor(x))})
```

```
# Lasso
ecoli_coef <- lasso(ecoli_train, 12)
ecoli_coef <- as.data.frame(as.matrix(ecoli_coef))
eco_coef <- as.vector(ecoli_coef$s1)
ecoli_coefs  <- round(ecoli_coef$s1/median(eco_coef[eco_coef != 0]))
ecoli_coef$s1 <- ecoli_coefs

# Need  to check strain, SNP cluster,


# Make sure all the factors are interpe

#levels(ecoli_test$month_source) == levels(ecoli_train$month_source)
strain_levels_train <- as.list(levels(ecoli_train$Strain))
ecoli_test$Strain <- factor(ecoli_test$Strain,
                            levels = strain_levels_train)
snp_levels_train <- as.list(levels(ecoli_train$SNP.cluster))
ecoli_test$SNP.cluster <- factor(ecoli_test$SNP.cluster,
                                 levels = snp_levels_train)



# Make sure all
dim(model.matrix(outbreak~., ecoli_test))
```

```
## [1] 1986   42
```

```
dim(model.matrix(outbreak~., ecoli_train))
```

```
## [1] 5552   42
```

```
# Coefficents
eco_coefs_table <- ecoli_coef %>%
                   filter(s1 != 0) %>%
```

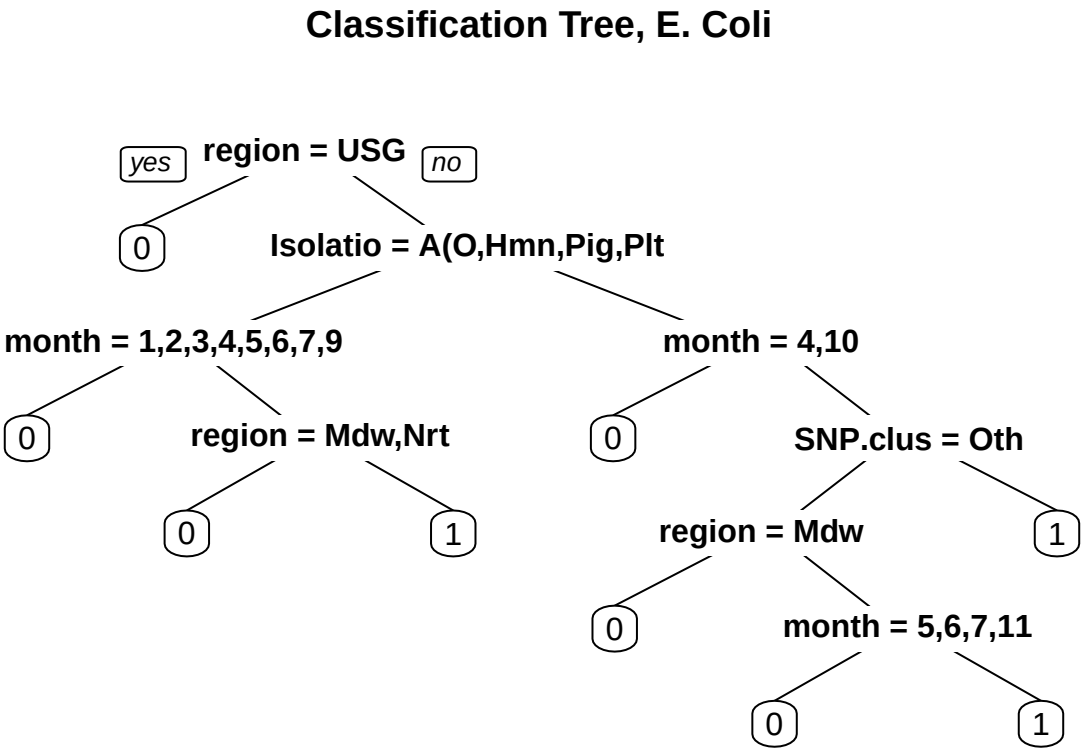
```
                    dplyr::rename(Score =  s1)

eco_coefs_table %>%  kbl(caption = "Score coefficents",booktabs=T, escape=T, align = "c") %>%
                     kable_styling(full_width = FALSE, latex_options = c('hold_position'))
```

Table 4: Score coefficents

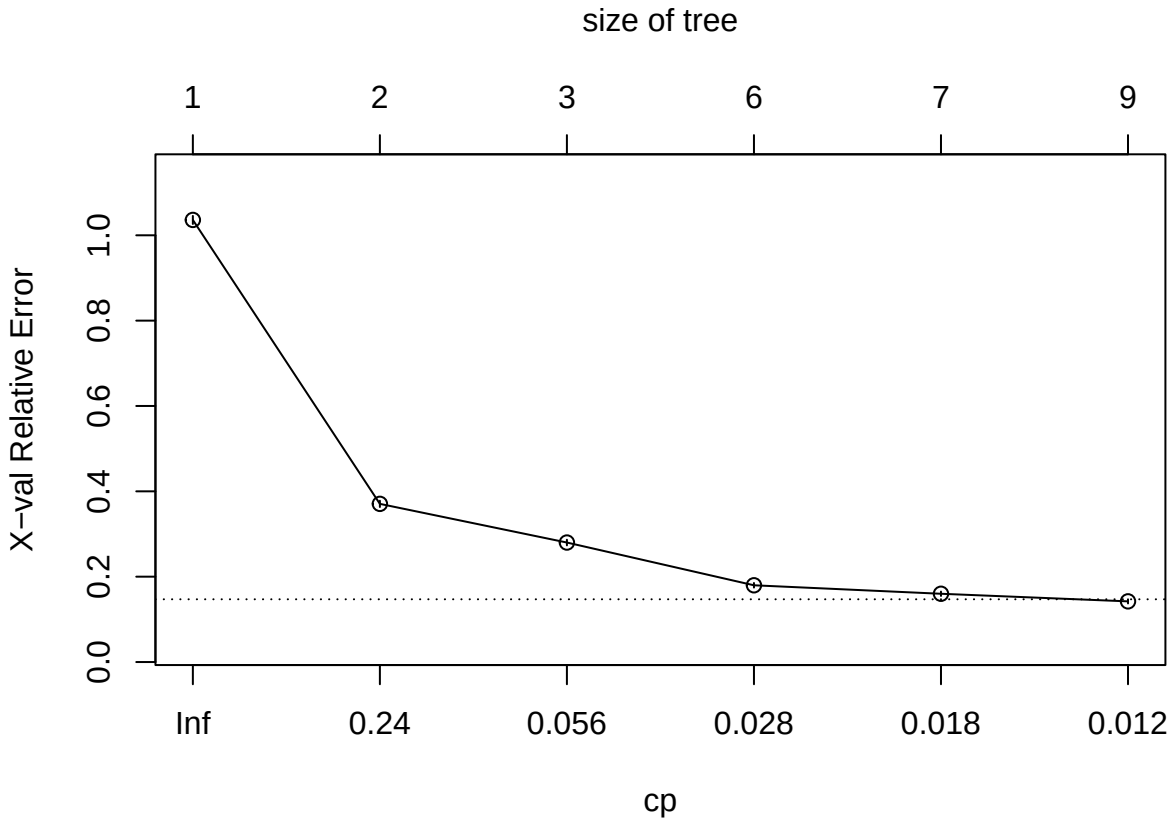
	Score
(Intercept)	1
SNP.clusterPDS000076681.93	2
AMR.genotypesacrF=MISTRANSLATION,blaEC=COMPLETE,mdtM=COMPLETE	2
Isolation.source.categoryHuman	-1
Isolation.source.categoryPoultry	-3
regionNortheast	3
regionUSA, General	-7

Tree

```
# We need to weight the tree (its done automatically in lasso)
prop_zero <- 1/(sum(ecoli_train$outbreak == 0)/nrow(ecoli_train))
prop_one <- 1/(sum(ecoli_train$outbreak == 1)/nrow(ecoli_train))

eco_weights <- ifelse(ecoli_train$outbreak == 0, prop_zero, prop_one)


# Fit the tree
tree_ecoli <- rpart(outbreak ~ ., data = ecoli_train,
                        method = "class", weights = eco_weights,
                        control=rpart.control(minsplit = 20,
                                              minbucket = 30))

# Plot the tree
prp(tree_ecoli,
    main = "Classification Tree, E. Coli")
```




```
# Looking  at complexity
plotcp(tree_ecoli)
```



```
# The tree already is meeting the standards of minimizing complexity
# we will not further prune it
```

Measures

```
variables3 <- model.matrix(outbreak~., ecoli_test)


ecoli_test$score <- variables3[,-1] %*% ecoli_coefs[-1]

mod_ecoli <- glm(outbreak ~ score, data=ecoli_test, family = quasibinomial())

# For confusion matrices
ecoli_test$predicted <- predict(mod_ecoli, type = "response")

risk_perfomance_ecoli <- roc(outbreak ~ predicted, data=ecoli_test)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```

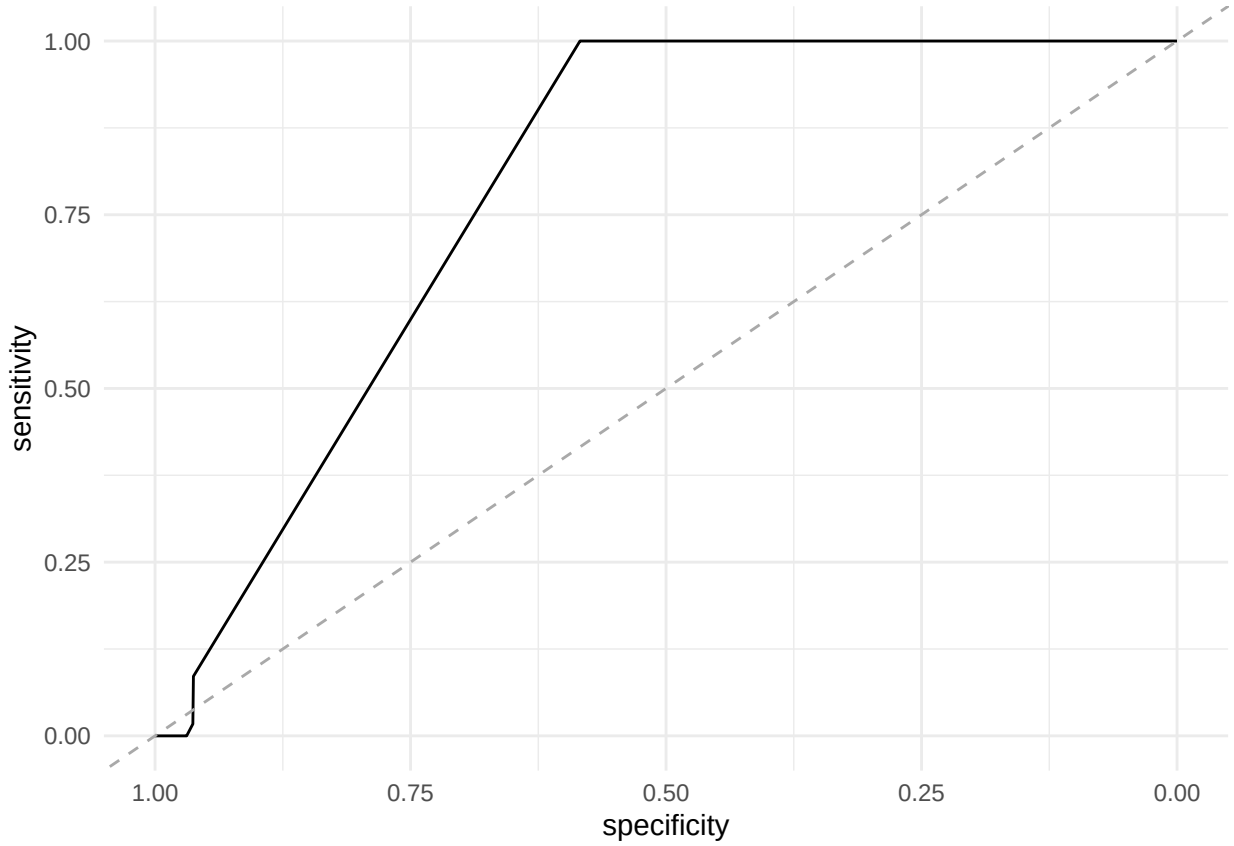
```
auc(risk_perfomance_ecoli)
```

```
## Area under the curve: 0.7896
```

```
BrierScore(mod_ecoli)
```

```
## [1] 0.07868401
```

```
ggroc(risk_perfomance_ecoli) +
  theme_minimal() +
  geom_abline(intercept = 1, slope = 1,
              color = "darkgrey", linetype = "dashed")
```




```
threshhold <- .5
confusionMatrix(as.factor(ifelse(ecoli_test$predicted > threshhold,
                                 1,0)), ecoli_test$outbreak)
```

```
## Warning in confusionMatrix.default(as.factor(ifelse(ecoli_test$predicted > :
## Levels are not in the same order for reference and data. Refactoring data to
## match.
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1811  175
##          1    0    0
##
##                Accuracy : 0.9119
##                  95% CI : (0.8985, 0.924)
##     No Information Rate : 0.9119
##     P-Value [Acc > NIR] : 0.5201
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.9119
##          Neg Pred Value :    NaN
```

```
##              Prevalence : 0.9119
##          Detection Rate : 0.9119
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

```
##  Test
ecoli_test$score <- as.numeric(ecoli_test[,9])
ecoli_test$outbreak_1 <- case_when(ecoli_test$outbreak == "0" ~ 0,
                                   ecoli_test$outbreak == "1" ~ 1)
ecoli_test %>% group_by(score) %>%
                     dplyr::summarize(pct = sum(outbreak_1)/n()) %>%
                     dplyr::rename("Percent outbreaks" = pct) %>%
                     kbl(caption = "Percent of cases at each score that belong to an outbreak",
                         booktabs=T, escape=T, align = "c") %>%
                     kable_styling(full_width = FALSE, latex_options = c('hold_position'))
```

Table 5: Percent of cases at each score that belong to an outbreak

score	Percent outbreaks
-8	0.0000000
-7	0.0000000
-6	0.0000000
-3	0.0000000
-1	0.0000000
0	0.1893491
1	0.9230769
2	0.2142857
3	0.0000000

```
variables4 <- model.matrix(outbreak~., ecoli_train)
ecoli_train$scores <- variables4[,-1] %*% ecoli_coefs[-1]
ecoli_train$score <- as.numeric(ecoli_train[,9])
ecoli_train$outbreak_1 <- case_when(ecoli_train$outbreak == "0" ~ 0,
                                    ecoli_train$outbreak == "1" ~ 1)

ecoli_train  %>% group_by(score) %>%
                     dplyr::summarize(pct = sum(outbreak_1)/n()) %>%
                     dplyr::rename("Percent outbreaks" = pct) %>%
                     kbl(caption = "Percent of cases at each score that belong to an outbreak",
                         booktabs=T, escape=T, align = "c") %>%
                     kable_styling(full_width = FALSE, latex_options = c('hold_position'))
```

```
print("tree")
```

```
## [1] "tree"
```

```
preds = predict(tree_ecoli, ecoli_train, type = "class")
sensitivity(preds, ecoli_train$outbreak)
```

```
## [1] 0.911465
```

Table 6: Percent of cases at each score that belong to an outbreak

score	Percent outbreaks
-8	0.0000000
-7	0.0035842
-6	0.0000000
-5	0.0000000
-4	0.0000000
-3	0.0000000
-1	0.0723982
0	0.1357430
1	0.4736842
2	0.8018648
3	0.7690355
5	0.0000000

```
specificity(preds, ecoli_train$outbreak)
```

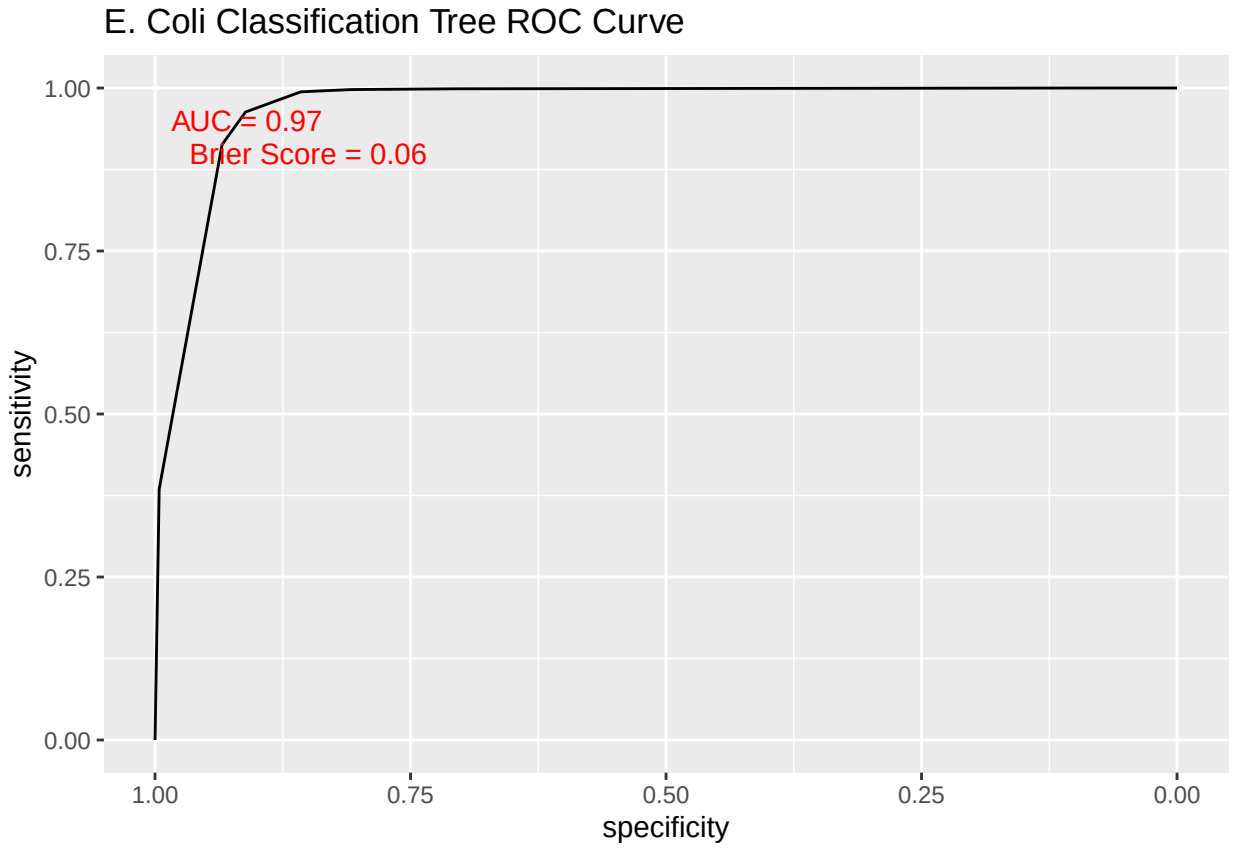
```
## [1] 0.9631829
```

```
# Predict with Tree
predict_ecoli_tree <- as.data.frame(predict(tree_ecoli, ecoli_train))
names(predict_ecoli_tree) <- c("prob_tree")
ecoli_append <- cbind(ecoli_train, predict_ecoli_tree)

# ROC Analysis
ecoli_acc <- roc(outbreak ~ prob_tree, data =ecoli_append)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls > cases
```

```
#AUC
ecoli_AUC = round(auc(ecoli_acc), 2)

# Brier Score
ecoli_Brscr = round(sum((predict_ecoli_tree[,2] - (as.numeric(ecoli_train$outbreak) - 1))^2)/length(ecol

ggroc(ecoli_acc) + ggtitle('E. Coli Classification Tree ROC Curve') +
  annotate("text", x = .91, y = .95, label = paste0('AUC = ', ecoli_AUC), color = 'red') +
  annotate("text", x = .85, y = .9, label = paste0('Brier Score = ', ecoli_Brscr), color = 'red')
```




```
# Predict on Test Set
pred_test = predict(tree_ecoli, newdata = ecoli_test, type = "class")
sensitivity(pred_test, ecoli_test$outbreak)
```

```
## [1] 0.759249
```

```
specificity(pred_test, ecoli_test$outbreak)
```

```
## [1] 0.2685714
```

```
# Predict with Tree
predict_test_tree <- as.data.frame(predict(tree_ecoli, ecoli_test))
names(predict_test_tree) <- c("prob_tree")
ecoli_test_append <- cbind(ecoli_test, predict_test_tree)

# ROC Analysis
ecoli_test_acc <- roc(outbreak ~ prob_tree, data =ecoli_test_append)
```

```
## Setting levels: control = 0, case = 1
## Setting direction: controls > cases
```

```
#AUC
ecoli_test_AUC = round(auc(ecoli_test_acc), 2)

# Brier Score
ecoli_test_Brscr = round(sum((predict_test_tree[,2] - (as.numeric(ecoli_test$outbreak) - 1))^2)/length(

ggroc(ecoli_test_acc) + ggtitle('E. Coli Classification Tree ROC Curve') +
  annotate("text", x = .91, y = .95, label = paste0('AUC = ', ecoli_test_AUC), color = 'red') +
```

```
  annotate("text", x = .85, y = .9, label = paste0('Brier Score = ', ecoli_test_Brscr), color = 'red')
```