Case Study 4

Shubh Shah

2024-11-05

This case will be going over both the Boston data set and the German Credit data set

Boston Housing Data

The goal for the Boston Housing data set was the analyze and model the housing data set using CART and linear regression. Starting off, I had loaded the data from the MASS library splitting the data into 90% training and 10% testing. Below will show the code.

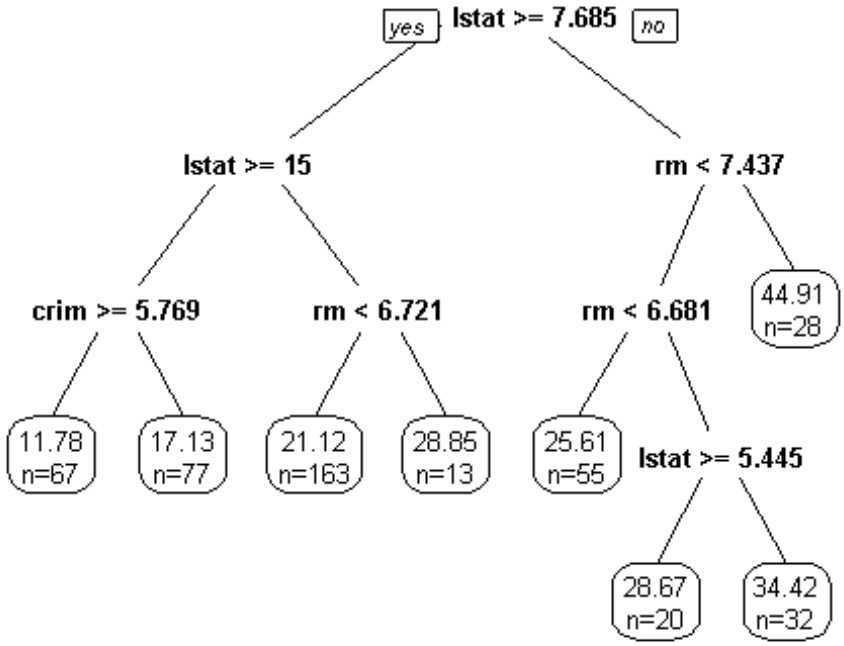
```
boston <- data(Boston)
sample_index <- sample(nrow(Boston), nrow(Boston)*0.90)
train <- Boston[sample_index, ]
test <- Boston[-sample_index, ]
```

After splitting the dataset, I had visualized the regression tree model using the training data set predicting the target median home value (medv). Afterwards, I had visualized the regression tree using the 'prp' function, which had displayed the splits at each node.

```
boston_rpart <- rpart(formula = medv ~., data = train)
boston_rpart
```

```
## n= 455
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 455 38486.3000 22.56615
##    2) lstat>=7.685 320 10243.3300 18.52000
##      4) lstat>=15 144  2818.8700 14.64167
##        8) crim>=5.76921 67   920.3364 11.77761 *
##        9) crim< 5.76921 77   870.7322 17.13377 *
##      5) lstat< 15 176  3486.3320 21.69318
##       10) rm< 6.7205 163  2383.1860 21.12270 *
##       11) rm>=6.7205 13   384.9523 28.84615 *
##    3) lstat< 7.685 135 10586.1500 32.15704
##      6) rm< 7.437 107  3766.2290 28.81963
##       12) rm< 6.6805 55  1037.6210 25.61273 *
##       13) rm>=6.6805 52  1564.7130 32.21154
##         26) lstat>=5.445 20   336.4775 28.67500 *
##         27) lstat< 5.445 32   821.7547 34.42188 *
##      7) rm>=7.437 28  1073.7470 44.91071 *
```

```
prp(boston_rpart, digits = 4, extra = 1)
```



After visualization, I had then started to work on finding the in-sample and out-of-sample predictions. Below will be the in-sample-predictions. When running this model, I had got low values, before finalizing this project, I had received a MSE of 15.72. However, a low value would represent a better fit. But its important to understand overfitting

```
# In Sample Prediction
boston_train_pred_tree = predict(boston_rpart)
mse.tree <- mean((boston_train_pred_tree - train$medv)^2)
print(mse.tree)

## [1] 17.20617
```

I had moved on to finding the out of sample predictions. I had found it by finding the averaged squared difference betweeen the predicted and actual values of medv in the test set. Its important that I find this as it would measure how it generalize on new, unforseen data. A higher out of sample MSE. Before submission, I had received a value of 29.98356. Since my out of sample is higher than my in sample, this can be an example of overfitting. However after running this model multiple times, it would seem that the in sample would be lower than out of sample at certain points.

```
# Out of Sample Prediction
boston_test_pred_tree = predict(boston_rpart, test)
mpse.tree <- mean((boston_test_pred_tree - test$medv)^2)
print(mpse.tree)

## [1] 27.39011
```

Moving on, I had created a linear regression model that was fitted on the trainng dataset without 'indus' and 'age'. I had then calculated the in sample and out of sample mse. I had found that the in sample was smaller than the out of sample. Likewise, the out of sample was higher than the in sample CART model and out of sample MSE. This suggests that the CART model would be able to fit non linear relationships in comparison to a linear regression model.

```
boston.reg = lm(medv~. -indus -age, data = train)

boston_train_pred_reg = predict(boston.reg, train)
mse.reg <- mean((boston_train_pred_reg - train$medv)^2)
print(mse.reg)

## [1] 21.52628

boston_test_pred_reg = predict(boston.reg, test)
mpse.reg <- mean((boston_test_pred_reg - test$medv)^2)
print(mpse.reg)

## [1] 25.54215
```

Running this test, I had received many different inputs. I had an out of sample lower than in sample, and sometimes different numbers that had made me update this markdown. I do apologize for the incorrections within it. However from my result, I had found that the CART model generalized well for this relationship.

German Credit Data

```
library(dplyr)
library(ROCR)
credit_data <- read.csv(file = "https://xiaoruizhu.github.io/Data-Mining-
R/lecture/data/credit_default.csv", header=T)
```

Before I had done the split, I had certain categorical data converted to factors and then had performed the split. This would be 80% training and 20% testing.

```
credit_data<- rename(credit_data, default=default.payment.next.month)
# convert categorical data to factor
credit_data$SEX<- as.factor(credit_data$SEX)
credit_data$EDUCATION<- as.factor(credit_data$EDUCATION)
credit_data$MARRIAGE<- as.factor(credit_data$MARRIAGE)
index <- sample(nrow(credit_data),nrow(credit_data)*0.80)
credit_train = credit_data[index,]
credit_test = credit_data[-index,]
```

I had then built the classification tree on the training model. Its going to be confusing, so I will do my best to label everything properly. The credit_rpart0 would represent an rpart model with no specific matrix applied.

```
credit_rpart0 <- rpart(formula = default ~ ., data = credit_train, method =
"class")
credit_rpart0
```

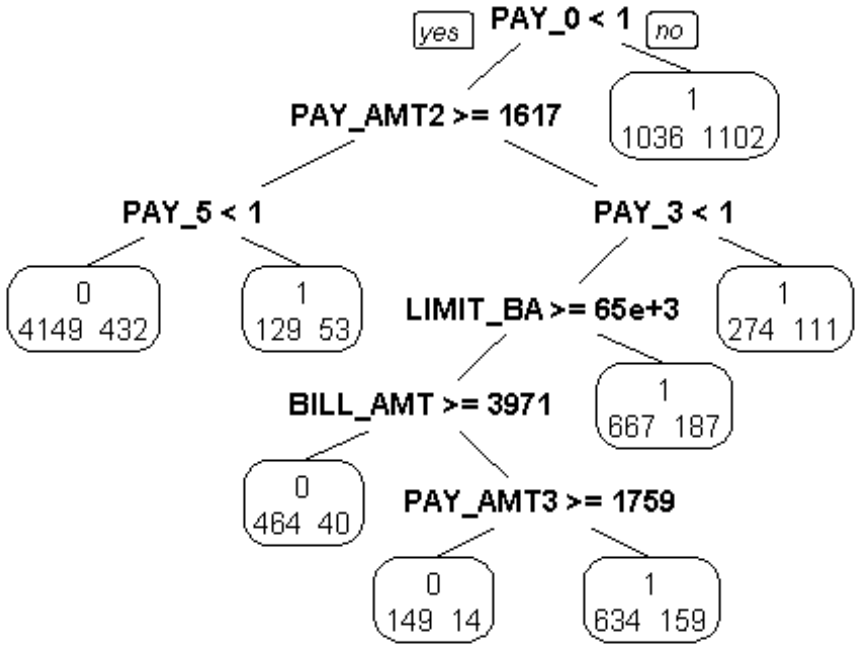
```
## n= 9600
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
## 1) root 9600 2098 0 (0.7814583 0.2185417)
##   2) PAY_0< 1.5 8591 1404 0 (0.8365732 0.1634268) *
##   3) PAY_0>=1.5 1009  315 1 (0.3121903 0.6878097) *
```

The second one would represent a model with the assumption that the false negatives would cost 5 times of false positives. This would be from within the Classification Tree PDF. From my understanding, I believe that it assigns 5 for false negatives and 1 for false positives, I had used the lecture to help.

```
credit_rpart <- rpart(formula = default ~ . , data = credit_train, method =
"class", parms = list(loss=matrix(c(0,5,1,0), nrow = 2)))
credit_rpart
```

```
## n= 9600
##
## node), split, n, loss, yval, (yprob)
##       * denotes terminal node
##
##  1) root 9600 7502 1 (0.78145833 0.21854167)
##    2) PAY_0< 0.5 7462 4980 0 (0.86652372 0.13347628)
##      4) PAY_AMT2>=1616.5 4763 2425 0 (0.89817342 0.10182658)
##        8) PAY_5< 1 4581 2160 0 (0.90569745 0.09430255) *
##        9) PAY_5>=1 182  129 1 (0.70879121 0.29120879) *
##      5) PAY_AMT2< 1616.5 2699 2188 1 (0.81067062 0.18932938)
##       10) PAY_3< 1 2314 1914 1 (0.82713915 0.17286085)
##         20) LIMIT_BAL>=65000 1460 1065 0 (0.85410959 0.14589041)
##           40) BILL_AMT2>=3970.5 504  200 0 (0.92063492 0.07936508) *
##           41) BILL_AMT2< 3970.5 956  783 1 (0.81903766 0.18096234)
##             82) PAY_AMT3>=1759 163   70 0 (0.91411043 0.08588957) *
##             83) PAY_AMT3< 1759 793  634 1 (0.79949559 0.20050441) *
##         21) LIMIT_BAL< 65000 854  667 1 (0.78103044 0.21896956) *
##       11) PAY_3>=1 385  274 1 (0.71168831 0.28831169) *
##    3) PAY_0>=0.5 2138 1036 1 (0.48456501 0.51543499) *

prp(credit_rpart, extra = 1)
```



In Sample prediction. Predicted 'default' values on traing data

```
pred0 <- predict(credit_rpart0, type="class")
table(credit_train$default, pred0, dnn = c("True", "Pred"))
```

```
##     Pred
## True    0    1
##    0 7187  315
##    1 1404  694
```

```
credit_train.pred.tree1<- predict(credit_rpart, credit_train, type="class")
table(credit_train$default, credit_train.pred.tree1,
dnn=c("Truth","Predicted"))
```

```
##      Predicted
## Truth    0    1
##     0 4762 2740
##     1  486 1612
```

Out of Sample

```
credit_test.pred.tree1<-predict(credit_rpart, credit_test, type="class")
table(credit_test$default, credit_test.pred.tree1, dnn = c("Truth",
"Predicted"))
```

```
##      Predicted
## Truth    0    1
```

```
##     0 1180  686
##     1  145  389
```

Missclassification

```
cost <- function(r, phat){
  weight1 <- 5
  weight0 <- 1
  pcut <- weight0/(weight1+weight0)
  c1 <- (r==1)&(phat<pcut) #logical vector - true if actual 1 but predict 0
  c0 <-(r==0)&(phat>pcut) #logical vector - true if actual 0 but predict 1
  return(mean(weight1*c1+weight0*c0))
}

cost(credit_train$default, predict(credit_rpart, credit_train, type="prob"))
```

```
## [1] 0.66
```

Out of Sample Missclassification

```
cost(credit_test$default, predict(credit_rpart, credit_test, type = "prob"))
```

```
## [1] 0.6827083
```

Logisitic Regression Model for Comparison

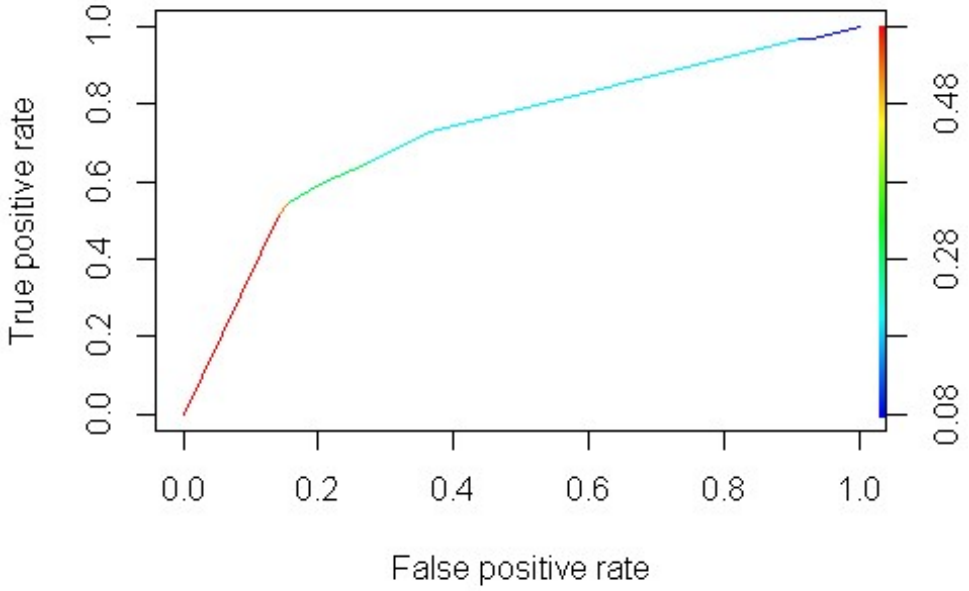
```
#Fit logistic regression model
credit_glm <- glm(default~.,data = credit_train,family=binomial)
#Get binary prediction
credit_test_pred_glm <- predict(credit_glm, credit_test, type="response")
#Calculate cost using test set
cost(credit_test$default, credit_test_pred_glm)
```

```
## [1] 0.6470833
```

AUC Calculation: the value of 0.7214 (when I had ran it) indicated that it can correctly identify defaults. I do beleive that there is room for improvement within this model itself. However, it is effective in finding high risk customers.

```
credit_test_prob_rpart = predict(credit_rpart, credit_test, type="prob")

pred = prediction(credit_test_prob_rpart[,2], credit_test$default)
perf = performance(pred, "tpr", "fpr")
plot(perf, colorize=TRUE)
```




```
slot(performance(pred, "auc"), "y.values")[[1]]
```

```
## [1] 0.7270715
```

Overall: When I had ran it, I had received a value of 0.6879167 from the Logistic Regression, whereas the CART model had an out of sample cost of 0.694375. Since they are somewhat similar, I would prefarably go with the CART model, as it would prioritze of reducing false negatives.