TJADASYN Experiments

Julia Fortna and Tejasvi Yadav

2024-12-19

```
library(tjadasyn)
library(tree)

## Warning: package 'tree' was built under R version 4.3.3

library(caret)

## Warning: package 'caret' was built under R version 4.3.3

## Loading required package: ggplot2

## Warning: package 'ggplot2' was built under R version 4.3.3

## Loading required package: lattice
```

Experiments

The following experiments were performed on each of the datasets listed above, which are each imbalanced with a binary response variable and '1' as the minority class. We are investigating ADASYN's performance with a single decision tree. The performance metrics are located in objects called conf followed by the first letter of the data set name and the experiment number (ex. confO1 is the performance metrics for oil in experiment 1)

1: Baseline ADASYN

```
### Oil
adaOil1 <- tjadasyn(oil,"response")
adaOil1[,49] <- as.factor(adaOil1[,49])
sO1 <- splitting(adaOil1,"V49.1")
trainO1 <- adaOil1[sO1,]
testO1 <- adaOil1[-sO1,]
oil_tree1 <- tree(V49.1~., trainO1)
predO1 <- predict(oil_tree1, newdata = testO1,type = "class")
#### model performance
confO1<-confusionMatrix(predO1,testO1$V49.1, mode = "everything")
### Madelon
adaMad1 <- tjadasyn(madelon, "response")
sM1 <- splitting(adaMad1, "V1")
trainM1 <- adaMad1[sM1,]
testM1 <- adaMad1[-sM1,]
trainM1[,1] <- as.factor(trainM1[,1])
testM1[,1] <- as.factor(testM1[,1])
mad_tree1 <- tree(V1~., data = trainM1)
```

```
predM1 <- predict(mad_tree1, newdata = testM1, type = "class")
#### Model Performance
confM1<-confusionMatrix(predM1,testM1$V1, mode = "everything")
### Arcene
adaArc1 <- tjadasyn(arcene,"response")
adaArc1[,1] <- as.factor(adaArc1[,1])
sA1 <- splitting(adaArc1,"V1")
trainA1 <- adaArc1[sA1,]
testA1 <- adaArc1[-sA1,]
arc_tree1 <- tree(V1~., trainA1)
predA1 <- predict(arc_tree1, newdata = testA1,type = "class")
#### model performance
confA1<-confusionMatrix(predA1,testA1$V1, mode = "everything")
### Gisette
adagd1 <- tjadasyn(gisette, var= "response")
sg1 <- splitting(adagd1, "V1")
traing1 <- adagd1[sg1,]
testg1 <- adagd1[-sg1,]
traing1[,1] <- as.factor(traing1[,1])
testg1[,1] <- as.factor(testg1[,1])
ag_tree1 <- tree(V1~., data = traing1)
predg1 <- predict(ag_tree1, newdata = testg1, type = "class")
#### model accuracy
confg1<-confusionMatrix(predg1,testg1$V1, mode = "everything")
### Caravan
adaCar1 <- tjadasyn(car,"response")
adaCar1[,86] <- as.factor(adaCar1[,86])
sC1 <- splitting(adaCar1,"V86")
trainC1 <- adaCar1[sC1,]
testC1 <- adaCar1[-sC1,]
car_tree1 <- tree(V86~., trainC1)
predC1 <- predict(car_tree1, newdata = testC1,type = "class")
#### model performance
confC1<-confusionMatrix(predC1,testC1$V86, mode = "everything")
```

2. Reduce feature space to only top 50% most informative features based on correlation with response variable

```
### Oil
adaOil2 <- tjadasyn(oil,"response", subsetFeat = TRUE,completeData = FALSE)
adaOil2[,17] <- as.factor(adaOil2[,17])
sO2 <- splitting(adaOil2,"V17")
trainO2 <- adaOil2[sO2,]
testO2 <- adaOil2[-sO2,]
oil_tree2 <- tree(V17~., trainO2)
predO2 <- predict(oil_tree2, newdata = testO2,type="class")
#### model performance
confO2<-confusionMatrix(predO2,testO2$V17, mode = "everything")
### Madelon
adaMad2 <- tjadasyn(madelon, subsetFeat = TRUE, var= "response")
```

```r
sM2 <- splitting(adaMad2, "V6")
trainM2 <- adaMad2[sM2,]
testM2 <- adaMad2[-sM2,]
trainM2[,1] <- as.factor(trainM2[,1])
testM2[,1] <- as.factor(testM2[,1])
mad_tree2 <- tree(V6~., data = trainM2)
predM2 <- predict(mad_tree2, newdata = testM2, type = "class")
## Model Performance
confM2<-confusionMatrix(predM2,testM2$V6, mode = "everything")
### Arcene
# remove columns with 0 variance
zero_var_cols_arc <- which(apply(arcene, 2, var) == 0)
arcdata_clean <- arcene[, apply(arcene, 2, var) != 0]
adarc2 <- tjadasyn(arcdata_clean, subsetFeat = TRUE, var= "response")
ac2 <- splitting(adarc2, "V7")
trainarc2 <- adarc2[ac2,]
testarc2 <- adarc2[-ac2,]
trainarc2[,1] <- as.factor(trainarc2[,1])
testarc2[,1] <- as.factor(testarc2[,1])
arc_tree2 <- tree(V7~., data = trainarc2)
predarc2 <- predict(arc_tree2, newdata = testarc2, type = "class")
## Model Performance
confarc2<-confusionMatrix(predarc2,testarc2$V7, mode = "everything")
### Gisette
#removing columns with zero varaince before calculating the correlation
matrix
zero_var_cols <- which(apply(gisette, 2, var) == 0)
gdataIMB_clean <- gisette[, apply(gisette, 2, var) != 0]
adagd2 <- tjadasyn(gdataIMB_clean, subsetFeat = TRUE, var= "response")
sg2 <- splitting(adagd2, "V7")
traing2 <- adagd2[sg2,]
testg2 <- adagd2[-sg2,]
traing2[,1] <- as.factor(traing2[,1])
testg2[,1] <- as.factor(testg2[,1])
ag_tree2 <- tree(V7~., data = traing2)
predg2 <- predict(ag_tree2, newdata = testg2, type = "class")
## Model Performance
confg2<-confusionMatrix(predg2,testg2$V7, mode = "everything")
### Caravan
adaCar2 <- tjadasyn(car,"response", subsetFeat = TRUE,completeData = FALSE)
adaCar2[,26] <- as.factor(adaCar2[,26])
sC2 <- splitting(adaCar2,"V26")
trainC2 <- adaCar2[sC2,]
testC2 <- adaCar2[-sC2,]
car_tree2 <- tree(V26~., trainC2)
predC2 <- predict(car_tree2, newdata = testC2,type="class")
#### model performance
confC2<-confusionMatrix(predC2,testC2$V26, mode = "everything")
```

3. Reduce feature space only when calculating nearest neighbors

```
### Oil
adaOil3 <- tjadasyn(oil,"response", subsetFeat = TRUE,completeData = TRUE)
adaOil3[,49] <- as.factor(adaOil3[,49])
sO3 <- splitting(adaOil3,"V49.1")
trainO3 <- adaOil3[sO3,]
testO3 <- adaOil3[-sO3,]
oil_tree3 <- tree(V49.1~., trainO3)
predO3 <- predict(oil_tree3, newdata = testO3,type="class")
## model performace
confO3<-confusionMatrix(predO3,testO3$V49.1,mode = "everything")
### Madelon
adaMad3 <- tjadasyn(madelon,"response", subsetFeat = TRUE,completeData =
TRUE)
adaMad3[,1] <- as.factor(adaMad3[,1])
sM3 <- splitting(adaMad3,"V1")
trainM3 <- adaMad3[sM3,]
testM3 <- adaMad3[-sM3,]
mad_tree3 <- tree(V1~., trainM3)
predM3 <- predict(mad_tree3, newdata = testM3,type="class")
## model performance
confM3<-confusionMatrix(predM3,testM3$V1,mode = "everything")
### Arcene
adaArc3 <- tjadasyn(arcdata_clean,"response", subsetFeat = TRUE,completeData
= TRUE)
adaArc3[,1] <- as.factor(adaArc3[,1])
sA3 <- splitting(adaArc3,"V1")
trainA3 <- adaArc3[sA3,]
testA3 <- adaArc3[-sA3,]
arc_tree3 <- tree(V1~., trainA3)
predA3 <- predict(arc_tree3, newdata = testA3,type = "class")
## model performance
confA3<-confusionMatrix(predA3,testA3$V1,mode = "everything")
### Gisette
adaG3 <- tjadasyn(gdataIMB_clean,"response", subsetFeat = TRUE,completeData =
TRUE)
adaG3[,1] <- as.factor(adaG3[,1])
sG3 <- splitting(adaG3,"V1")
trainG3 <- adaG3[sG3,]
testG3 <- adaG3[-sG3,]
g_tree3 <- tree(V1~., trainG3)
predg3 <- predict(g_tree3, newdata = testG3,type = "class")
## model performance
confG3<-confusionMatrix(predg3,testG3$V1,mode = "everything")
### Caravan
adaCar3 <- tjadasyn(car,"response", subsetFeat = TRUE,completeData = TRUE)
adaCar3[,86] <- as.factor(adaCar3[,86])
sC3 <- splitting(adaCar3,"V86")
trainC3 <- adaCar3[sC3,]
```

```
testC3 <- adaCar3[-sC3,]
car_tree3 <- tree(V86~., trainC3)
predC3 <- predict(car_tree3, newdata = testC3,type="class")
#### model performance
confC3<-confusionMatrix(predC3,testC3$V86, mode = "everything")
```

4. Use corruption parameter with informative feature selection

```
### Oil
adaOil4 <- tjadasyn(oil,"response", subsetFeat = TRUE,completeData = TRUE,
corrupt = 0.01)
adaOil4[,49] <- as.factor(adaOil4[,49])
sO4 <- splitting(adaOil4,"V49.1")
trainO4 <- adaOil4[sO4,]
testO4 <- adaOil4[-sO4,]
oil_tree4 <- tree(V49.1~., trainO4)
predO4 <- predict(oil_tree4, newdata = testO4,type="class")
## model performace
confO4<-confusionMatrix(predO4,testO4$V49.1,mode = "everything")

### Madelon
adaMad4 <- tjadasyn(madelon,"response", subsetFeat = TRUE,completeData =
TRUE, corrupt = 0.02)
adaMad4[["V1"]] <- as.factor(adaMad4[["V1"]])
sM4 <- splitting(adaMad4,"V1")
trainM4 <- adaMad4[sM4,]
testM4 <- adaMad4[-sM4,]
mad_tree4 <- tree(V1~., trainM4)
predM4 <- predict(mad_tree4, newdata = testM4,type="class")
## model performace
confM4<-confusionMatrix(predM4,testM4$V1,mode = "everything")

### Arcene
adaArc4 <- tjadasyn(arcdata_clean,"response", subsetFeat = TRUE,completeData
= TRUE, corrupt = 0.05)
adaArc4[,1] <- as.factor(adaArc4[,1])
sA4 <- splitting(adaArc4,"V1")
trainA4 <- adaArc4[sA4,]
testA4 <- adaArc4[-sA4,]
arc_tree4 <- tree(V1~., trainA4)
predA4 <- predict(arc_tree4, newdata = testA4,type = "class")
## model performance
confA4<-confusionMatrix(predA4,testA4$V1,mode = "everything")

### Gisette
adaG4 <- tjadasyn(gdataIMB_clean,"response", subsetFeat = TRUE,completeData =
TRUE, corrupt = 0.02)
adaG4[,1] <- as.factor(adaG4[,1])
sG4 <- splitting(adaG4,"V1")
trainG4 <- adaG4[sG4,]
```

```
testG4 <- adaG4[-sG4,]
g_tree4 <- tree(V1~., trainG4)
predg4 <- predict(g_tree4, newdata = testG4,type = "class")
## model performance
confG4<-confusionMatrix(predg4,testG4$V1,mode = "everything")

### Caravan
adaCar4 <- tjadasyn(car,"response", subsetFeat = TRUE,completeData =
TRUE,corrupt = 0.01)
adaCar4[,86] <- as.factor(adaCar4[,86])
sC4 <- splitting(adaCar4,"V86")
trainC4 <- adaCar4[sC4,]
testC4 <- adaCar4[-sC4,]
car_tree4 <- tree(V86~., trainC4)
predC4 <- predict(car_tree4, newdata = testC4,type="class")
#### model performance
confC4<-confusionMatrix(predC4,testC4$V86, mode = "everything")
```

5. Baseline ADASYN w/ corruption

```
### Oil
adaOil5 <- tjadasyn(oil,"response", corrupt = 0.02)
adaOil5[,49] <- as.factor(adaOil5[,49])
sO5 <- splitting(adaOil5,"V49.1")
trainO5 <- adaOil5[sO5,]
testO5 <- adaOil5[-sO5,]
oil_tree5 <- tree(V49.1~., trainO5)
predO5 <- predict(oil_tree5, newdata = testO5,type="class")
## model performace
confO5<-confusionMatrix(predO5,testO5$V49.1,mode = "everything")

### Madelon
adaMad5 <- tjadasyn(madelon,"response", corrupt = 0.01)
adaMad5[["V1"]] <- as.factor(adaMad5[["V1"]])
sM5 <- splitting(adaMad5,"V1")
trainM5 <- adaMad5[sM5,]
testM5 <- adaMad5[-sM5,]
mad_tree5 <- tree(V1~., trainM5)
predM5 <- predict(mad_tree5, newdata = testM5,type="class")
## model performace
confM5<-confusionMatrix(predM5,testM5$V1,mode = "everything")

### Arcene
adaArc5 <- tjadasyn(arcene,"response", corrupt = 0.01)
adaArc5[,1] <- as.factor(adaArc5[,1])
sA5 <- splitting(adaArc5,"V1")
trainA5 <- adaArc5[sA5,]
testA5 <- adaArc5[-sA5,]
arc_tree5 <- tree(V1~., trainA5)
predA5 <- predict(arc_tree5, newdata = testA5,type = "class")
```

```
## model performance
confA5<-confusionMatrix(predA5,testA5$V1,mode = "everything")

### Gisette
adaG5 <- tjadasyn(gisette,"response", corrupt = 0.02)
adaG5[,1] <- as.factor(adaG5[,1])
sG5 <- splitting(adaG5,"V1")
trainG5 <- adaG5[sG5,]
testG5 <- adaG5[-sG5,]
g_tree5 <- tree(V1~., trainG5)
predg5 <- predict(g_tree5, newdata = testG5,type = "class")
## model performance
confG5<-confusionMatrix(predg5,testG5$V1,mode = "everything")

### Caravan
adaCar5 <- tjadasyn(car,"response",corrupt = 0.01)
adaCar5[,86] <- as.factor(adaCar5[,86])
sC5 <- splitting(adaCar5,"V86")
trainC5 <- adaCar5[sC5,]
testC5 <- adaCar4[-sC5,]
car_tree5 <- tree(V86~., trainC5)
predC5 <- predict(car_tree5, newdata = testC5,type="class")
#### model performance
confC5<-confusionMatrix(predC5,testC5$V86, mode = "everything")
```

6. Diminish tuning parameter – reduces feature space to a specified proportion of informative features designated by correlation with response variable

```
### Oil
adaOil6 <- tjadasyn(oil,"response", subsetFeat = TRUE,completeData = TRUE,
diminish = 0.65)
adaOil6[,49] <- as.factor(adaOil6[,49])
sO6 <- splitting(adaOil6,"V49.1")
trainO6 <- adaOil6[sO6,]
testO6 <- adaOil6[-sO6,]
oil_tree6 <- tree(V49.1~., trainO6)
predO6 <- predict(oil_tree6, newdata = testO6,type="class")
## model performace
confO6<-confusionMatrix(predO6,testO6$V49.1,mode = "everything")

### Madelon
adaMad6 <- tjadasyn(madelon,"response", subsetFeat = TRUE,completeData =
TRUE, diminish = .6)
adaMad6[["V1"]] <- as.factor(adaMad6[["V1"]])
sM6 <- splitting(adaMad6,"V1")
trainM6 <- adaMad6[sM6,]
testM6 <- adaMad6[-sM6,]
mad_tree6 <- tree(V1~., trainM6)
```

```r
predM6 <- predict(mad_tree6, newdata = testM6,type="class")
## model performace
confM6<-confusionMatrix(predM6,testM6$V1,mode = "everything")
### Arcene
adaArc6 <- tjadasyn(arcdata_clean,"response", subsetFeat = TRUE,completeData
= TRUE, diminish = 0.4)
adaArc6[,1] <- as.factor(adaArc6[,1])
sA6 <- splitting(adaArc6,"V1")
trainA6 <- adaArc6[sA6,]
testA6 <- adaArc6[-sA6,]
arc_tree6 <- tree(V1~., trainA6)
predA6 <- predict(arc_tree6, newdata = testA6,type = "class")
## model performance
confA6<-confusionMatrix(predA6,testA6$V1,mode = "everything")
### Gisette
adaG6 <- tjadasyn(gdataIMB_clean,"response", subsetFeat = TRUE,completeData =
TRUE, diminish = 0.2)
adaG6[,1] <- as.factor(adaG6[,1])
sG6 <- splitting(adaG6,"V1")
trainG6 <- adaG6[sG6,]
testG6 <- adaG6[-sG6,]
g_tree6 <- tree(V1~., trainG6)
predg6 <- predict(g_tree6, newdata = testG6,type = "class")
## model performance
confG6<-confusionMatrix(predg6,testG6$V1,mode = "everything")

### Caravan
adaCar6 <- tjadasyn(car,"response", subsetFeat = TRUE,completeData = TRUE,
diminish = 0.4)
adaCar6[,86] <- as.factor(adaCar6[,86])
sC6 <- splitting(adaCar6,"V86")
trainC6 <- adaCar6[sC6,]
testC6 <- adaCar6[-sC6,]
car_tree6 <- tree(V86~., trainC6)
predC6 <- predict(car_tree6, newdata = testC6,type="class")
#### model performance
confC6<-confusionMatrix(predC6,testC6$V86, mode = "everything")
```