rectangular_class_domain

Ye

7/28/2017

Load library

```
suppressWarnings(library(caret))
```

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

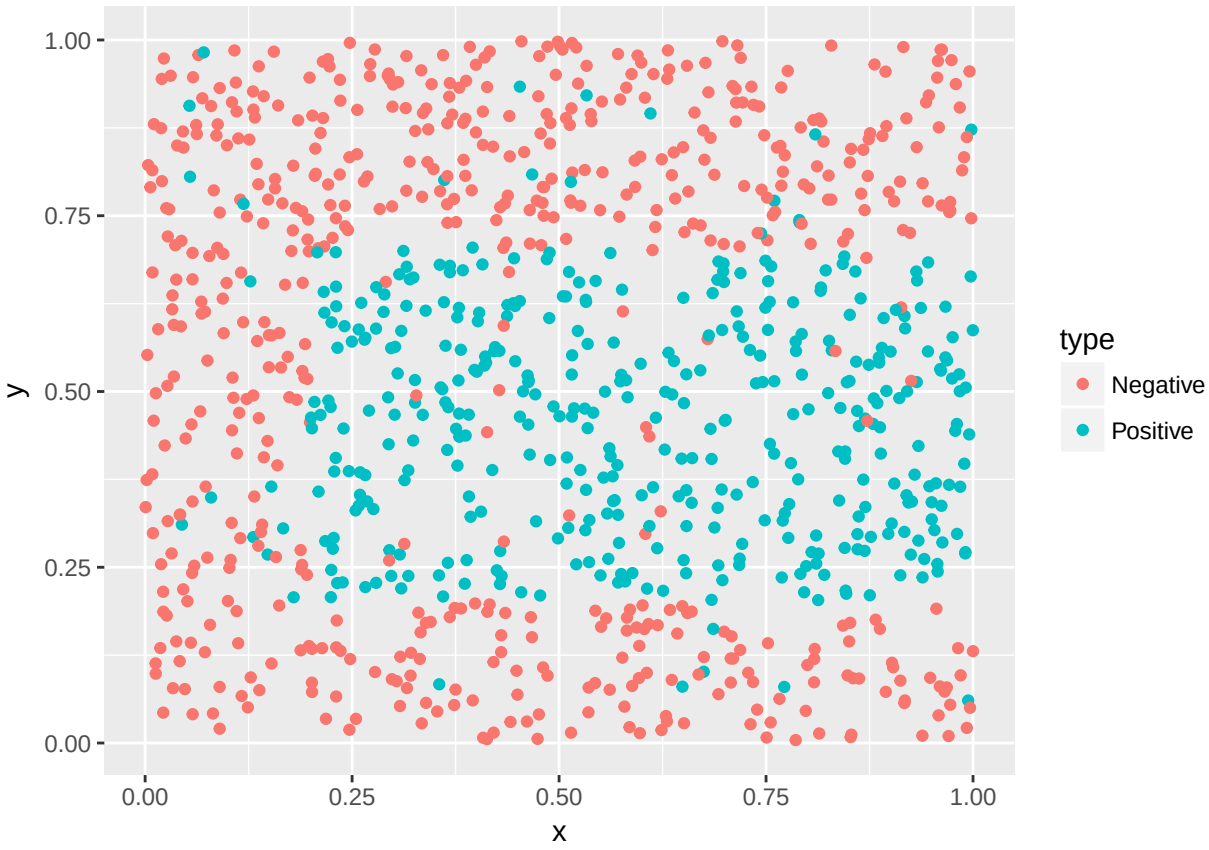
```
## Loading required package: ggplot2
```

```
suppressWarnings(library(rpart))
suppressWarnings(library(e1071))
```

Create rectangular class domain with certain randomness

```
N = 1000
xPos = 0.2
yMinPos = 0.2
yMaxPos = 0.7
newData = data.frame(x=runif(N),y=runif(N))
newData$type = with(newData,ifelse(x>xPos & y>yMinPos & y<yMaxPos,
                                   'Positive', 'Negative'))

n = N/10
newData$type[1:n] = c('Positive', 'Negative')[1+rbinom(n, 1, 0.5)]
newData$type = factor(newData$type)
newData = newData[sample(nrow(newData)),]
qplot(x=x,y=y,data=newData, color=type)
```



Logistic regression and perform cross validation (caret) to check the predictive quality

```
modelFormula = formula('type ~ x + y')
logrFit <- glm(modelFormula, family=binomial("logit"),data=newData)
print(summary(logrFit))
```

```
##
## Call:
## glm(formula = modelFormula, family = binomial("logit"), data = newData)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7026  -0.9746  -0.6837   1.1540   1.9829
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.7225     0.1860  -3.884 0.000103 ***
## x             1.9459     0.2416   8.053 8.09e-16 ***
## y            -1.2524     0.2392  -5.236 1.64e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 1353.7  on 999  degrees of freedom
## Residual deviance: 1252.8  on 997  degrees of freedom
## AIC: 1258.8
##
## Number of Fisher Scoring iterations: 4
```

```
ctrl <- trainControl(method = "cv", number = 10)
logrTrain <- train(modelFormula, data=newData,
                   method = 'glm', trControl = ctrl)
summary(logrTrain)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7026  -0.9746  -0.6837   1.1540   1.9829
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.7225     0.1860  -3.884 0.000103 ***
## x             1.9459     0.2416   8.053 8.09e-16 ***
## y            -1.2524     0.2392  -5.236 1.64e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1353.7  on 999  degrees of freedom
## Residual deviance: 1252.8  on 997  degrees of freedom
## AIC: 1258.8
##
## Number of Fisher Scoring iterations: 4
```

Use classification tree to fit the data

```
treeFit <- rpart(modelFormula, data=newData)
printcp(treeFit)
```

```
##
## Classification tree:
## rpart(formula = modelFormula, data = newData)
##
## Variables actually used in tree construction:
## [1] x y
##
## Root node error: 410/1000 = 0.41
##
## n= 1000
##
##        CP nsplit rel error  xerror     xstd
```

```
## 1 0.33902      0   1.00000 1.00000 0.037934
## 2 0.19512      2   0.32195 0.32683 0.026274
## 3 0.01000      3   0.12683 0.13415 0.017584
```

```
treeTrain <- train(modelFormula, method="rpart",data=newData,
                   trControl = ctrl)
```

Use SVM to fit the data

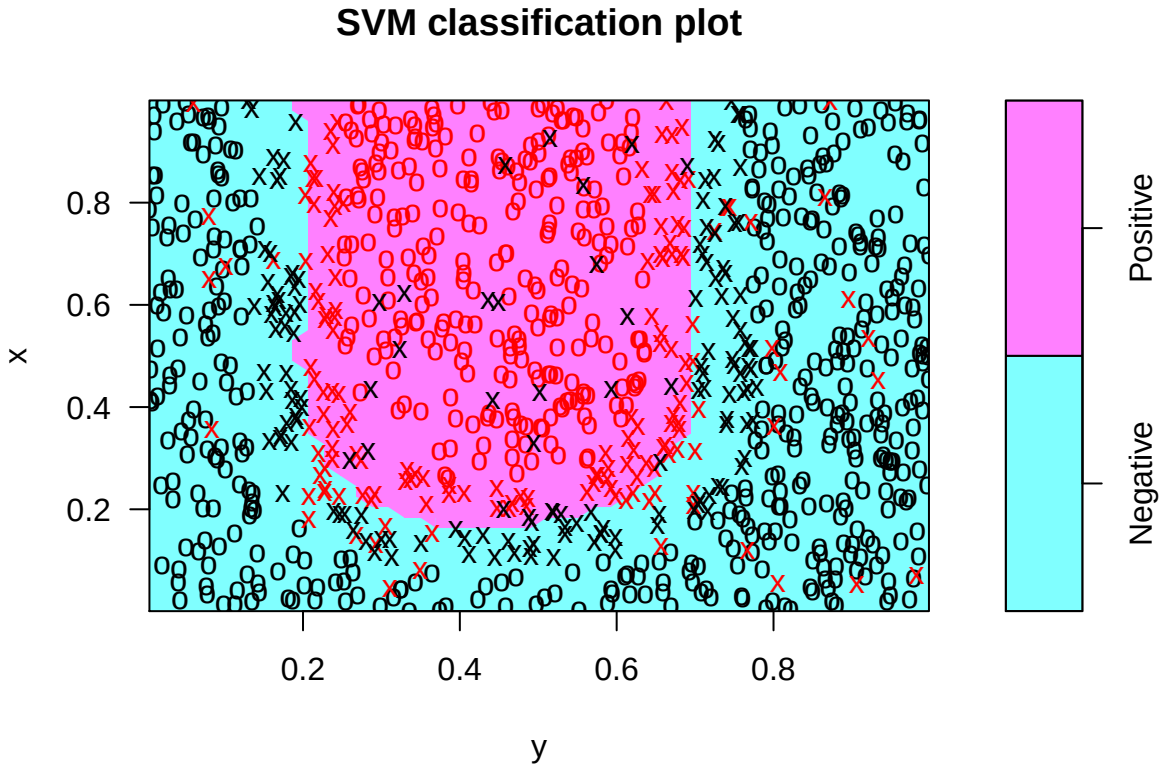
```
svmTuned <- tune.svm(type~., data = newData, gamma = 10^(-4:-1), cost = 5*(1:4))
summary(svmTuned)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  gamma cost
##    0.1   20
##
## - best performance: 0.087
##
## - Detailed performance results:
##    gamma cost error dispersion
## 1  1e-04    5 0.410 0.03800585
## 2  1e-03    5 0.388 0.06051630
## 3  1e-02    5 0.245 0.04453463
## 4  1e-01    5 0.096 0.02503331
## 5  1e-04   10 0.410 0.03800585
## 6  1e-03   10 0.371 0.03754997
## 7  1e-02   10 0.184 0.03835507
## 8  1e-01   10 0.092 0.02347576
## 9  1e-04   15 0.410 0.03800585
## 10 1e-03   15 0.366 0.03949684
## 11 1e-02   15 0.149 0.02846050
## 12 1e-01   15 0.090 0.02494438
## 13 1e-04   20 0.410 0.03800585
## 14 1e-03   20 0.363 0.03368151
## 15 1e-02   20 0.136 0.02913570
## 16 1e-01   20 0.087 0.02451757
```

```
svmTuned$best.parameters
```

```
##    gamma cost
## 16   0.1   20
```

```
plot(svmTuned$best.model, newData)
```




```
print(list(tree = treeTrain$results, logit = logrTrain$results))
```

```
## $tree
##          cp Accuracy     Kappa AccuracySD    KappaSD
## 1 0.0000000    0.943 0.8819222 0.02162817 0.04461735
## 2 0.1951220    0.887 0.7733622 0.05207900 0.10140037
## 3 0.3390244    0.689 0.2709764 0.12801476 0.35019431
##
## $logit
##   parameter Accuracy     Kappa AccuracySD    KappaSD
## 1      none    0.633 0.2093917 0.04473378 0.09975101
```