HW 3

Sam Reid

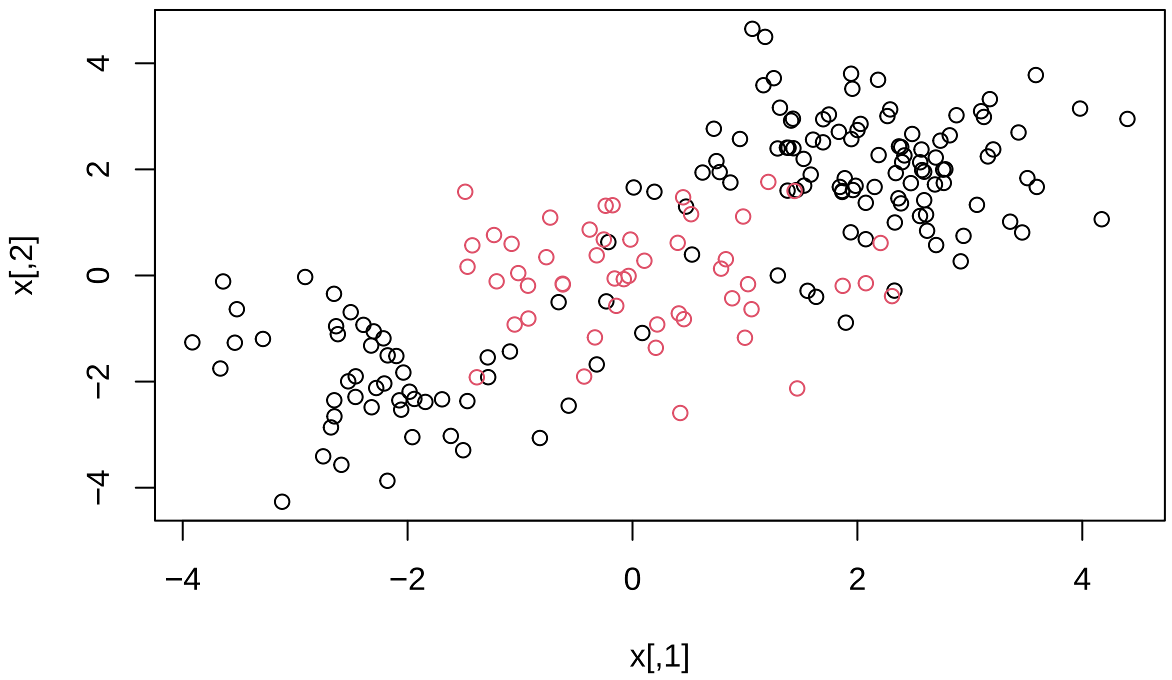
11/27/2023

In this homework, we will discuss support vector machines and tree-based methods. I will begin by simulating some data for you to use with SVM.

```r
library(e1071)
```

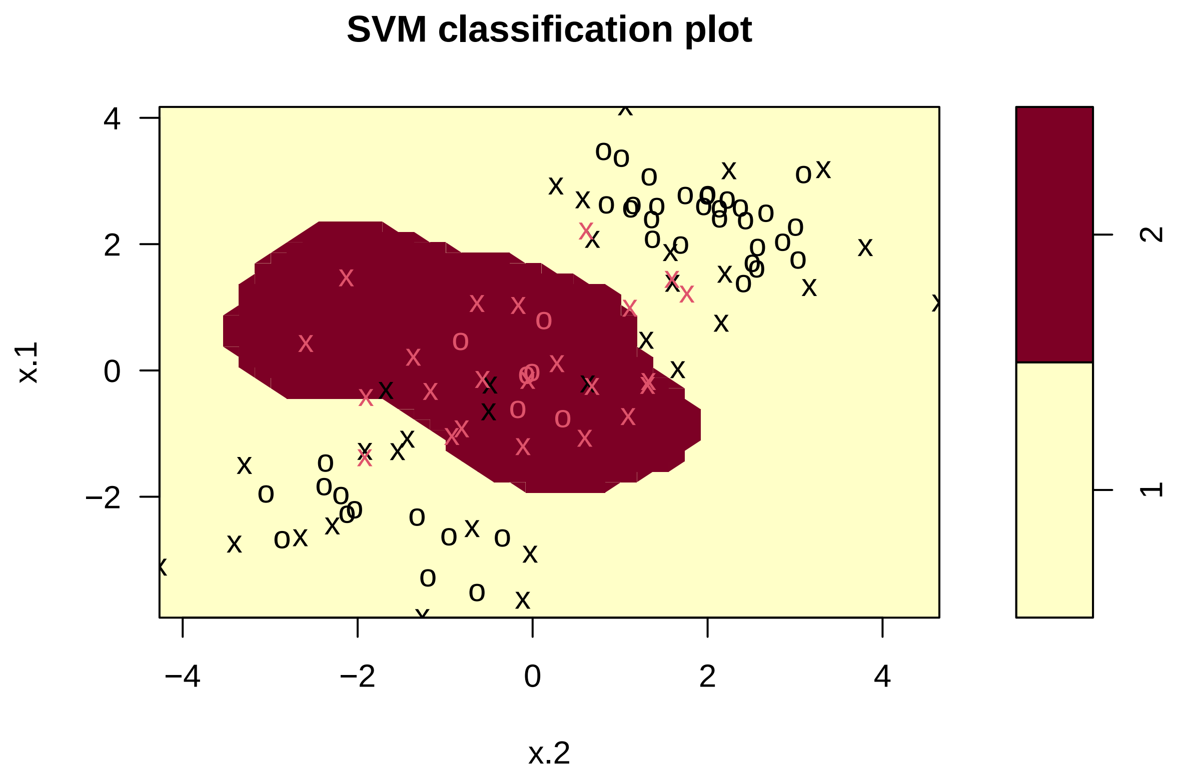
```
## Warning: package 'e1071' was built under R version 4.3.2
```

```r
set.seed(1)
x=matrix(rnorm(200*2),ncol=2)
x[1:100,]=x[1:100,]+2
x[101:150,]=x[101:150,]-2
y=c(rep(1,150),rep(2,50))
dat=data.frame(x=x,y=as.factor(y))
plot(x, col=y)
```



Quite clearly, the above data is not linearly separable. Create a training-testing partition with 100 random observations in the training partition. Fit an svm on this training data using the radial kernel, and tuning parameters $\gamma = 1$, cost $= 1$. Plot the svm on the training data.

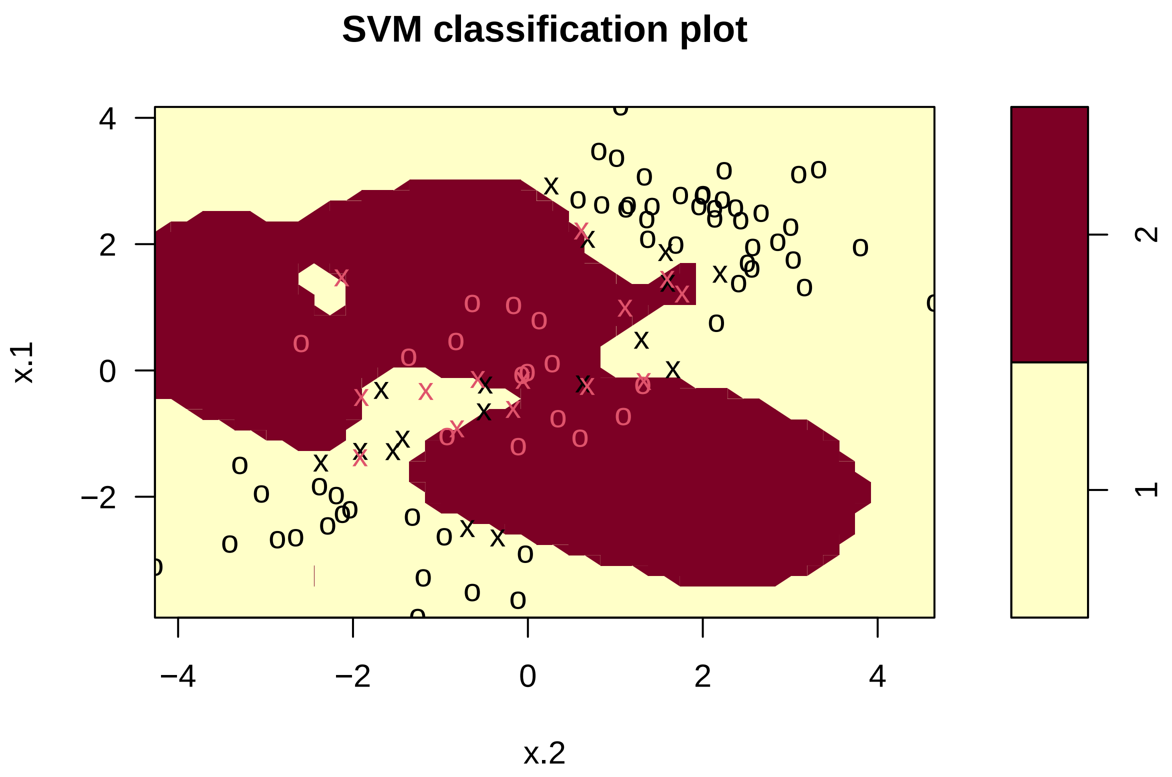
```
set.seed(1)
sample1 <- sample(200,100)
TRAIN <- dat[sample1,]
TEST <- dat[-sample1,]
svmfit <- svm(y~., data=TRAIN, kernel="radial", gamma=1, cost=1, scale="False")
plot(svmfit, TRAIN)
```



Notice that the above decision boundary is decidedly non-linear. It seems to perform reasonably well, but there are indeed some misclassifications. Let's see if increasing the cost [1] helps our classification error rate. Refit the svm with the radial kernel, $\gamma = 1$, and a cost of 10000. Plot this svm on the training data.

```
svmfit2 <- svm(y~., data=TRAIN, kernel="radial", gamma=1, cost=10000)
plot(svmfit2, TRAIN)
```

[1]Remember this is a parameter that decides how smooth your decision boundary should be



It would appear that we are better capturing the training data, but comment on the dangers (if any exist), of such a model.

Student Answer

The model that was generated is highly irregular, including enclaves and peninsulas that we know (because we made the data) do not represent the underlying data structure. This means testing error may be much higher than with a simpler more spherical model, as the areas that the irregular model tries to interpolate are more likely to be incorrect on new data. ##

Create a confusion matrix by using this svm to predict on the current testing partition. Comment on the confusion matrix. Is there any disparity in our classification results?

```r
#remove eval = FALSE in above
table(true=TEST[,"y"], pred=predict(svmfit2, newdata=TEST))
```

```
##     pred
## true  1  2
##    1 67 12
##    2  2 19
```

The model misclassifies type 1 variables as type 2 more often than the other way around. Visually this seems to be because the overfitted model extended out into the type 1 ‘region’ to try and catch the stray type

2s, but since these regions have a higher proportion of type 1s in the testing data, it misclassified more. A 'smaller' model would have the opposite effect. ##

Is this disparity because of imbalance in the training/testing partition? Find the proportion of class 2 in your training partition and see if it is broadly representative of the underlying 25% of class 2 in the data as a whole.

```
sum(TRAIN$y == 2)/nrow(TRAIN)
```

```
## [1] 0.29
```

Student Response There is a slight difference in proportion, but not likely enough to cause the disparity seen in the table. This likely means the problem was with the overcomplexity of the model itself.

Let's try and balance the above to solutions via cross-validation. Using the `tune` function, pass in the training data, and a list of the following cost and γ values: $\{0.1, 1, 10, 100, 1000\}$ and $\{0.5, 1,2,3,4\}$. Save the output of this function in a variable called `tune.out`.

```
set.seed(1)
tune.out <- tune(svm, y~., data=TRAIN, ranges=list(gamma=c(.1,1,10,100,1000), cost=c(.5,1,2,3,4)))
```

I will take `tune.out` and use the best model according to error rate to test on our data. I will report a confusion matrix corresponding to the 100 predictions.

```
table(true=TEST[,"y"], pred=predict(tune.out$best.model, newdata=TEST))
```

```
##     pred
## true  1  2
##    1 70  9
##    2  0 21
```

Comment on the confusion matrix. How have we improved upon the model in question 2 and what qualifications are still necessary for this improved model.

Student Response Our test misclassification rate is down to 9%, which means the model is much more accurately fitted to the underlying pattern. However, our data is still more likely to misclassify type 1 objects than type 2. In fact all of the misclassified objects in this testing set were type 1, which means any interpretation of the data on a specific unknown point should factor in the difference, especially if the real-world consequence of misclassifications are different based on direction. # Let's turn now to decision trees.

```
library(kmed)
```

```
## Warning: package 'kmed' was built under R version 4.3.2
```

```r
data(heart)
library(tree)
```

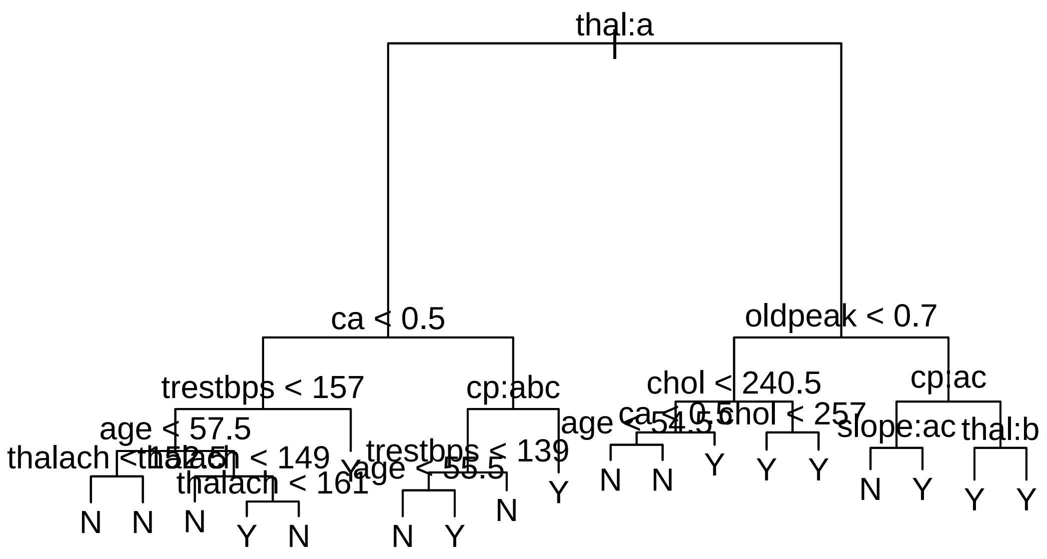
```
## Warning: package 'tree' was built under R version 4.3.2
```

The response variable is currently a categorical variable with four levels. Convert heart disease into binary categorical variable. Then, ensure that it is properly stored as a factor.

```r
disease_factor <- as.factor(ifelse(heart$class>0, "Y", "N"))
heart$class <- disease_factor
```

Train a classification tree on a 240 observation training subset (using the seed I have set for you). Plot the tree.

```r
library(tree)
set.seed(101)
sample2 <- sample(297,240)
train.x <- heart[sample2,]
test.x <- heart[-sample2,]
heart.tree <- tree(class~., data = train.x)
plot(heart.tree)
text(heart.tree)
```



Use the trained model to classify the remaining testing points. Create a confusion matrix to evaluate performance. Report the classification error rate.

```r
heart.pred = predict(heart.tree, test.x, type="class")
with(test.x, table(heart.pred, class))
```

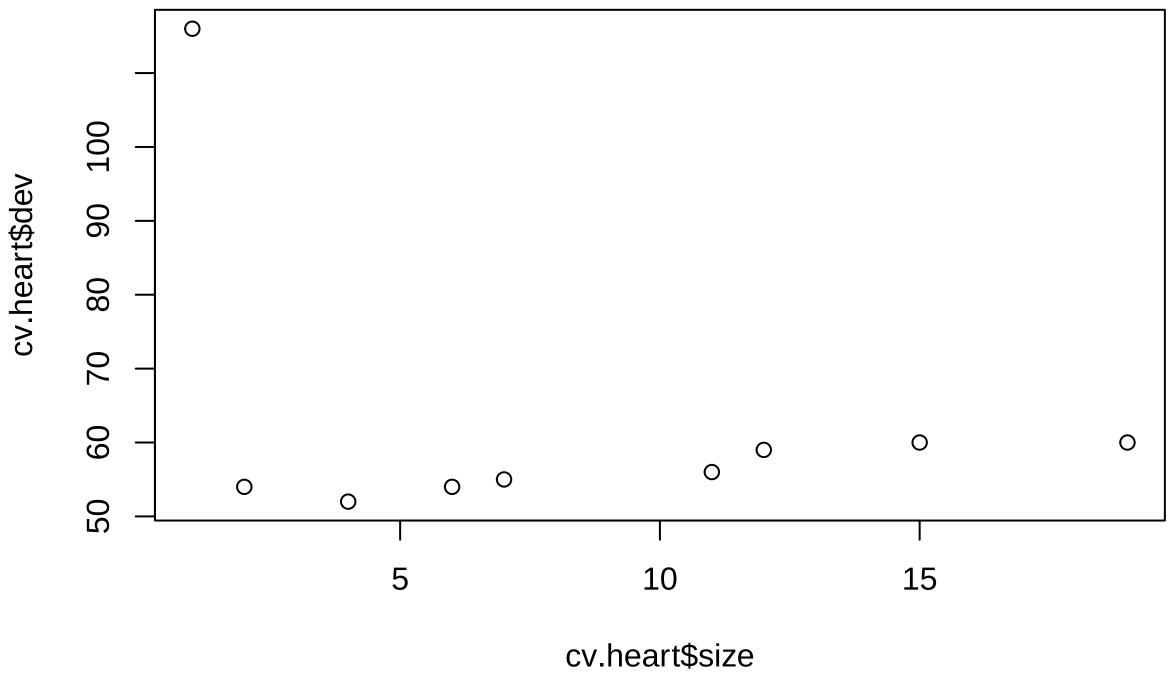
```
##           class
## heart.pred  N  Y
##          N 28  3
##          Y  8 18
```

```r
print(paste("Misclassification rate: ", (3+8)/57))
```

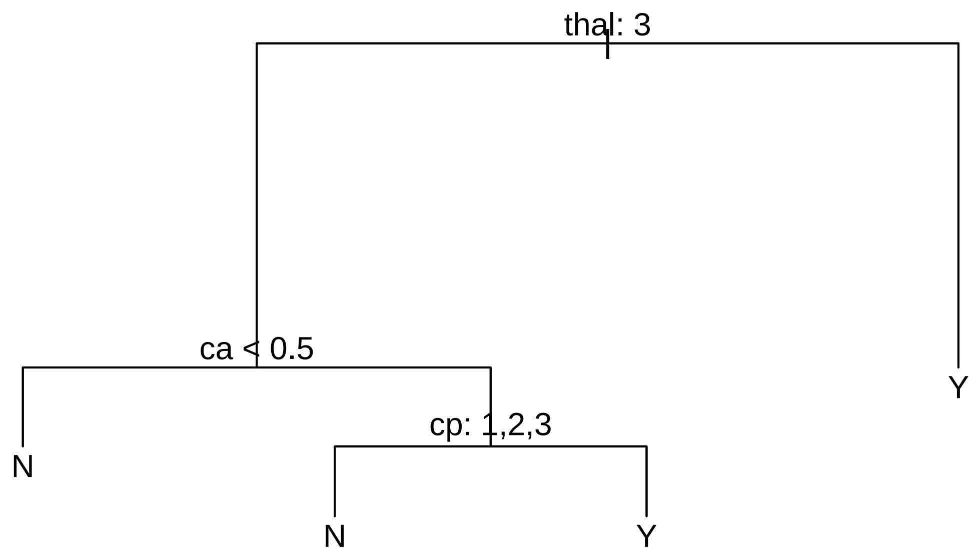
```
## [1] "Misclassification rate:  0.192982456140351"
```

Above we have a fully grown (bushy) tree. Now, cross validate it using the `cv.tree` command. Specify cross validation to be done according to the misclassification rate. Choose an ideal number of splits, and plot this tree. Finally, use this pruned tree to test on the testing set. Report a confusion matrix and the misclassification rate.

```r
set.seed(1)
cv.heart <- cv.tree(heart.tree, FUN = prune.misclass)
plot(cv.heart$size, cv.heart$dev)
```



```r
prune.heart <- prune.misclass(heart.tree, best=4)
plot(prune.heart)
text(prune.heart, pretty=0)
```



```
heart.pred2 = predict(prune.heart, test.x, type="class")
with(test.x, table(heart.pred2, class))
```

```
##            class
## heart.pred2  N  Y
##           N 26  4
##           Y 10 17
```

```
print(paste("Misclassification rate: ", (4+10)/57))
```

```
## [1] "Misclassification rate:  0.245614035087719"
```

Discuss the trade-off in accuracy and interpretability in pruning the above tree.

Student Input

While our model is slightly less accurate (24% compared to 19%), it is far more interpretable as we have narrowed in on the three most important factors in determining heart disease, and it is much easier to make a decision about a given patient using the simplified chart without having to input their data into an obscure function. However in this case we may want to prioritize accuracy above all else.

Discuss the ways a decision tree could manifest algorithmic bias.

Student Answer

Like any other model, if training data is substantially different from testing data, or from the whole set, there is the risk that the model will not be able to accurately describe patterns in the new data, as it will be trying to fit a structure that may not exist in some cases. Additionally, if you are trying to select for a response variable that is very unbalanced (say one class is <1%) there may simply not be enough data to accurately predict one variable as well as the other, no matter how complex the model is. A decision tree specifically may have issues in interpretation, in regards to some predictors being much more visibily 'important' due to their placement in the tree, when they may have a similar effect to other predictors further down, which could mislead readers into thinking there is more more direct correllation than there really is.