HW7 - DATA 609

Thomas Hill

December 5, 2021

```
library(caret)
```

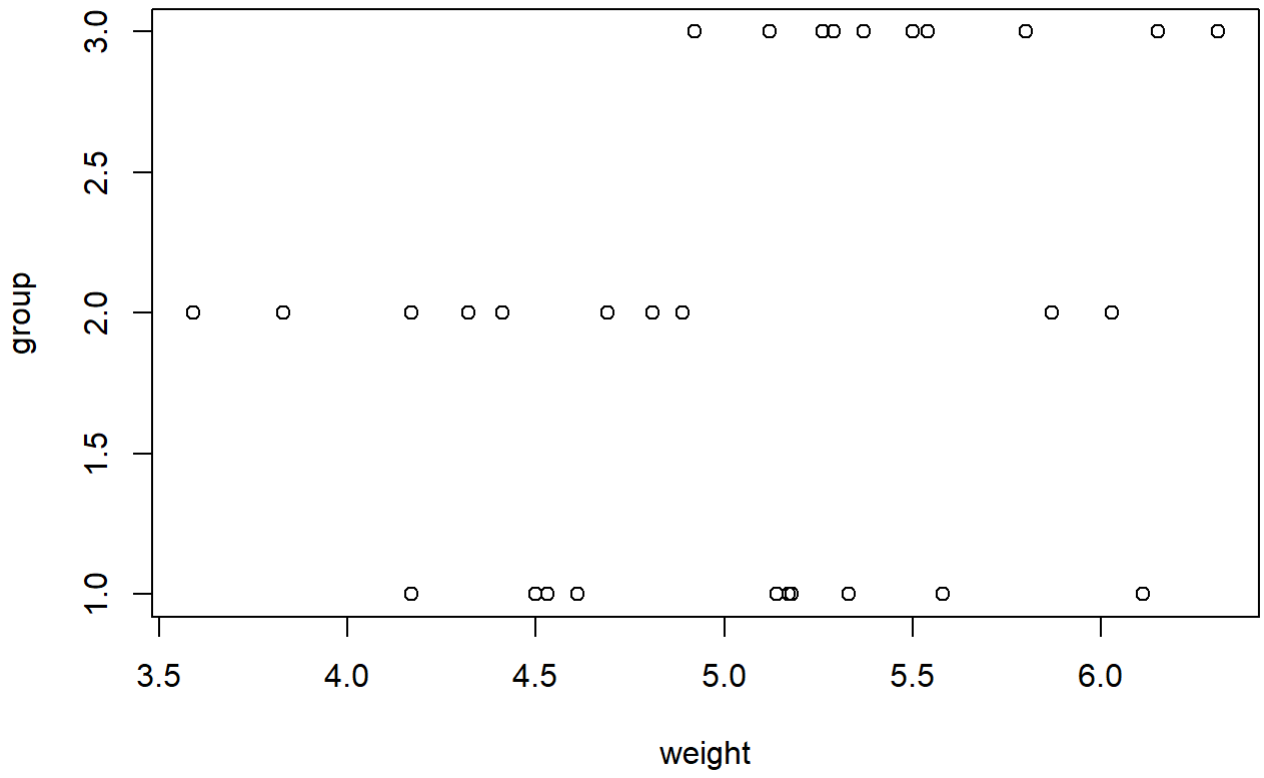
```
## Loading required package: ggplot2
```

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

```
library(e1071)
```

Ex. 1 – Use the *svm()* algorithm of the *e1071* package to carry out the support vector machine for the PlantGrowth dataset. Then, discuss the number of support vectors/samples [Install the *e1071* package in R if needed.]

```
plot(PlantGrowth)
```



PlantGrowth is a dataset that gives weight as the independent variable and labels each plant with one of three labels as levels of a factor. I'll consider two different kernels: the radial kernel, which is the default kernel used in *svm*, and the linear kernel.

```
pg_fit_rad <- svm(formula = group ~ weight, data = PlantGrowth, kenel = 'radial')
pg_fit_lin <- svm(formula = group ~ weight, data = PlantGrowth, kernel = 'linear')
```

Next, lets see how well each model predicts the categories in the original dataset.

```
rad_predict <- predict(pg_fit_rad, PlantGrowth)

100*round(table(rad_predict, PlantGrowth$group)/30,3)
```

```
##
## rad_predict ctrl trt1 trt2
##        ctrl  0.0  6.7  3.3
##        trt1 13.3 20.0  0.0
##        trt2 20.0  6.7 30.0
```

Per the initial data, each class is one third (33% of the data). Each row represents the predicted classification, while the columns are the true designation. For the radial fit, only 10% are predicted to be part of the control, while treatment 2 is overrepresented. This kernel only predicts the correct class 50% of the time.

```
lin_predict <- predict(pg_fit_lin, PlantGrowth)

100*round(table(lin_predict, PlantGrowth$group)/30,3)
```

```
##
## lin_predict ctrl trt1 trt2
##        ctrl  3.3  6.7  6.7
##        trt1 13.3 20.0  0.0
##        trt2 16.7  6.7 26.7
```

The linear kernel performs no better, with again 50% of the true classes predicted accurately. It does appear more sensitive to identifying the control group, while sacrificing performance with respect to identifying treatment 2. Treatment 1 appears unchanged.

```
print(pg_fit_rad$tot.nSV)
```

```
## [1] 29
```

```
print(pg_fit_lin$tot.nSV)
```

```
## [1] 27
```

Finally, lets consider the number of support vectors generated using each kernel. Both models in this case offer over two dozen support vectors, nearly as many as the number of samples in the original dataset. This indicates that each model is complex to accommodate the variation amongst the classes. Both numbers are high, so it's not clear that the linear kernel is significantly less complex than the radial.

Ex. 2 – Do a similar SVM analysis as that in the previous section using the *iris* dataset. Discuss the number of support vectors/samples.

```
summary(iris)
```

```
##   Sepal.Length    Sepal.Width     Petal.Length    Petal.Width
##  Min.   :4.300   Min.   :2.000   Min.   :1.000   Min.   :0.100
##  1st Qu.:5.100   1st Qu.:2.800   1st Qu.:1.600   1st Qu.:0.300
##  Median :5.800   Median :3.000   Median :4.350   Median :1.300
##  Mean   :5.843   Mean   :3.057   Mean   :3.758   Mean   :1.199
##  3rd Qu.:6.400   3rd Qu.:3.300   3rd Qu.:5.100   3rd Qu.:1.800
##  Max.   :7.900   Max.   :4.400   Max.   :6.900   Max.   :2.500
##        Species
##  setosa    :50
##  versicolor:50
##  virginica :50
##
##
##
```

The *iris* dataset has four variables being considered, with three possible categories. Again, lets consider two different kernels for the svm and how well they predict the classes of the original data

```
iris_fit_rad <- svm(formula = Species ~., data = iris, kenel = 'radial')
iris_fit_lin <- svm(formula = Species ~., data = iris, kenel = 'linear')
```

```
rad_predict_iris <- predict(iris_fit_rad, iris)

100*round(table(rad_predict_iris, iris$Species)/150,3)
```

```
##
## rad_predict_iris setosa versicolor virginica
##       setosa       33.3        0.0       0.0
##       versicolor    0.0       32.0       1.3
##       virginica     0.0        1.3      32.0
```

Again, each species comprises 33% of the original data (columns). The predicted species in each row is significantly better than the PlantGrowth example. All *setosa* species are correctly classified, and nearly all of the other two species are.

```
lin_predict_iris <- predict(iris_fit_lin, iris)

100*round(table(lin_predict_iris, iris$Species)/150,3)
```

```
##
## lin_predict_iris setosa versicolor virginica
##       setosa       33.3        0.0       0.0
##       versicolor    0.0       32.0       1.3
##       virginica     0.0        1.3      32.0
```

The linear kernel offers similar performance.

```
print(iris_fit_rad$tot.nSV)
```

```
## [1] 51
```

```
print(iris_fit_lin$tot.nSV)
```

```
## [1] 51
```

Finally, both kernels have the same number of support vectors: 51. This is significantly smaller than the initial sample of 150. This indicates that the svm was able to distinguish between the three species using a relatively simple model.

Ex. 3 – Use the *iris* dataset (or any other dataset) to select 80% of the samples for training *svm()*, then use the remaining 20% for validation. Discuss your results.

```
set.seed(1205)

train_index <- sample(seq_len(150), size =120) #generate random index separating iris into speci
fied samples
train <- iris[train_index,] #training data
test <- iris[-train_index,] #test data

test_cat <- iris[-train_index, 5]  #true designations for test


m_iris <- svm(formula = Species ~., data =train, kernel = 'radial') #iris model
iris_predict <- predict(m_iris, test) #find predicted categories

100*round(table(iris_predict, test_cat)/30,3) #contingency table
```

```
##             test_cat
## iris_predict setosa versicolor virginica
##   setosa       30.0        0.0       0.0
##   versicolor    0.0       33.3       3.3
##   virginica     0.0        0.0      33.3
```

```
print(m_iris$tot.nSV)
```

```
## [1] 47
```

Using a 20% cross-validation technique, a similar model is generated compared to example 2. The number of support vectors is slightly lower than the non-validated model, with n = 47.