Support Vector Classifier

Natashia Benjamin

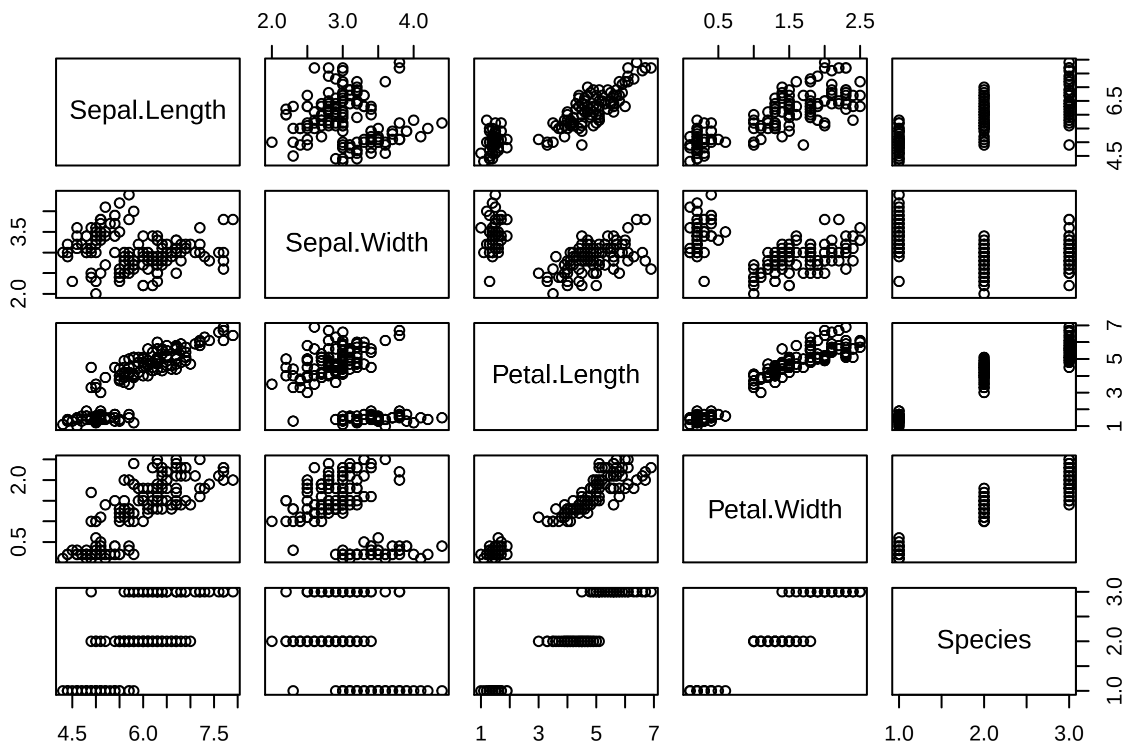
11/8/2021

This project aims at implementing a Support vector classifier on the iris data set in R. The data set contains 150 flowers from three different species, setosa, virginia, and versicolor. The code will use the dimension of the flowers petals and sepals (width and length), to predict which of the three specie they belong.

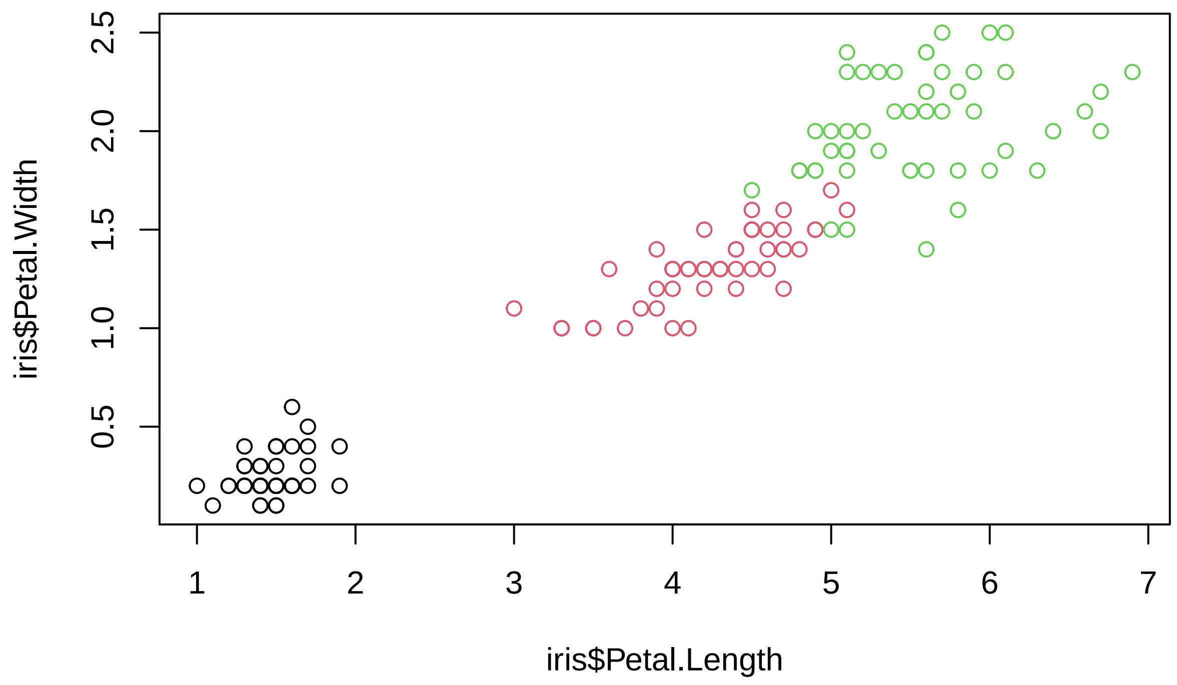
```r
library("e1071")
```

```
## Warning: package 'e1071' was built under R version 4.0.4
```

```r
plot(iris)
```



```r
plot(iris$Petal.Length, iris$Petal.Width, col=iris$Species)
```

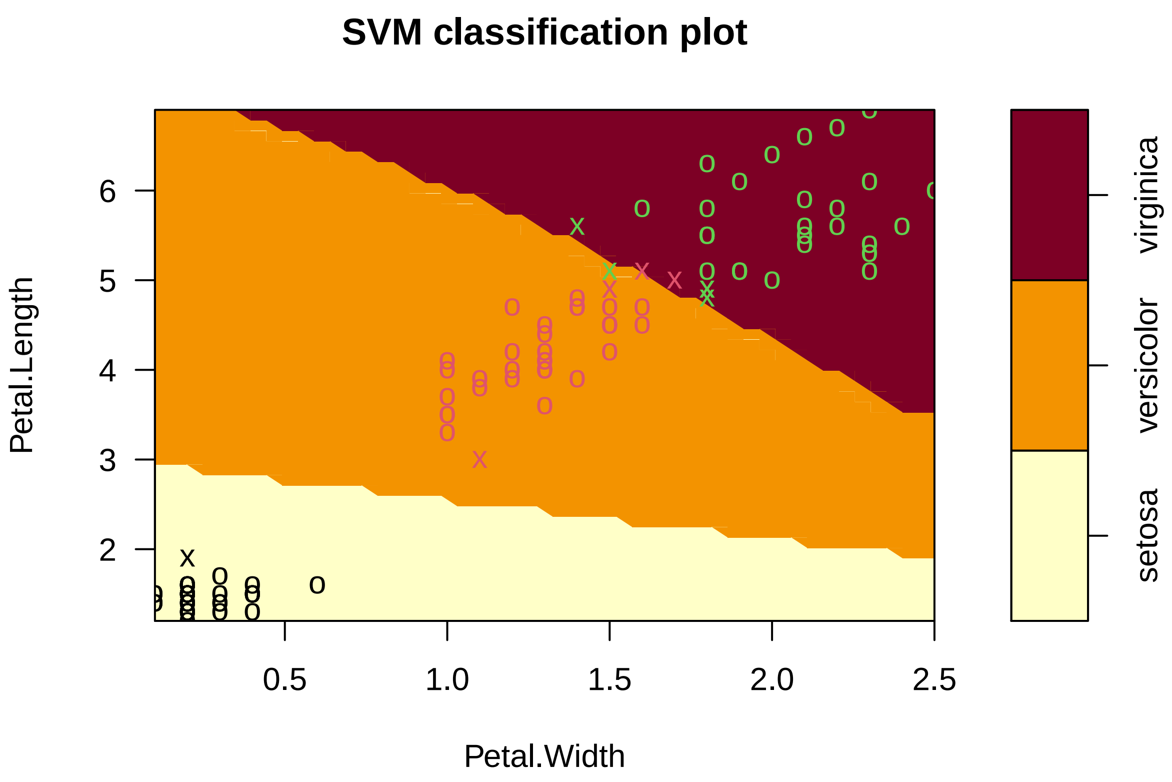


```r
s<-sample(150, 100)
col<-c("Petal.Length", "Petal.Width", "Species")
iris_train<-iris[s,col]
iris_test<-iris[-s,col]

svmfit <- svm(Species ~., data = iris_train, kernel = "polynomial", cost = 1, scale = FALSE)
print(svmfit)
```

```
##
## Call:
## svm(formula = Species ~ ., data = iris_train, kernel = "polynomial",
##     cost = 1, scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  1
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  11
```

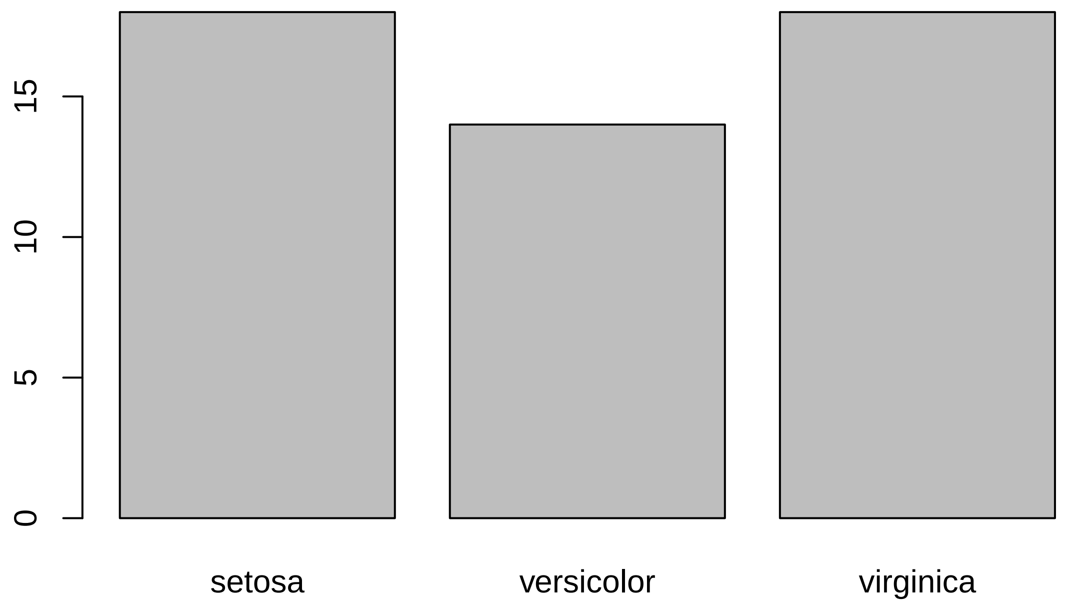
```
plot(svmfit, iris_train[,col])
```



```
tuned <- tune(svm, Species ~., data = iris_train, kernel = "polynomial", ranges = list(cost=c(0.001,0.0
# Will show the optimal cost parameter
summary(tuned)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##    10
##
## - best performance: 0.04
##
## - Detailed performance results:
##    cost error dispersion
## 1 1e-03  0.62 0.16865481
## 2 1e-02  0.26 0.12649111
## 3 1e-01  0.12 0.06324555
## 4 1e+00  0.05 0.07071068
## 5 1e+01  0.04 0.05163978
## 6 1e+02  0.04 0.05163978
```

```r
p <- predict(svmfit, iris_test[,col], type="class")
plot(p)
```



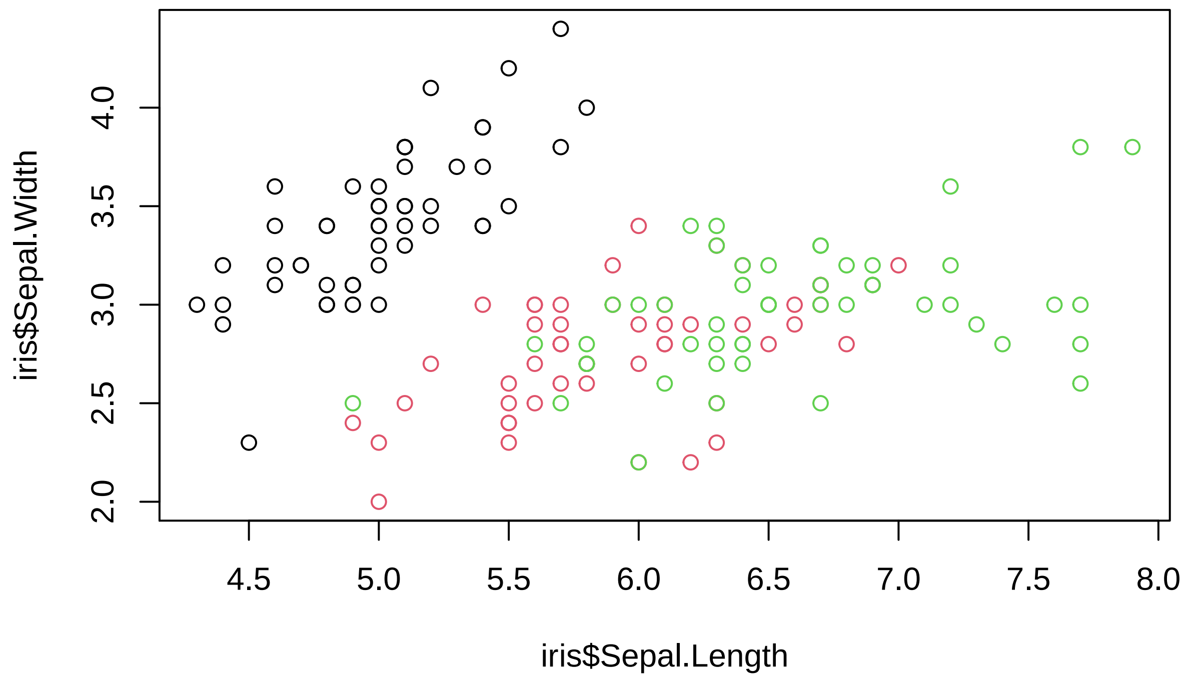
```r
table(p, iris_test[,3])
```

```
##
## p            setosa versicolor virginica
##   setosa         18          0         0
##   versicolor      0         12         2
##   virginica       0          1        17
```

```r
#mean(p== iris_test[,3])


#Classify based on Sepal


plot(iris$Sepal.Length, iris$Sepal.Width, col=iris$Species)
```

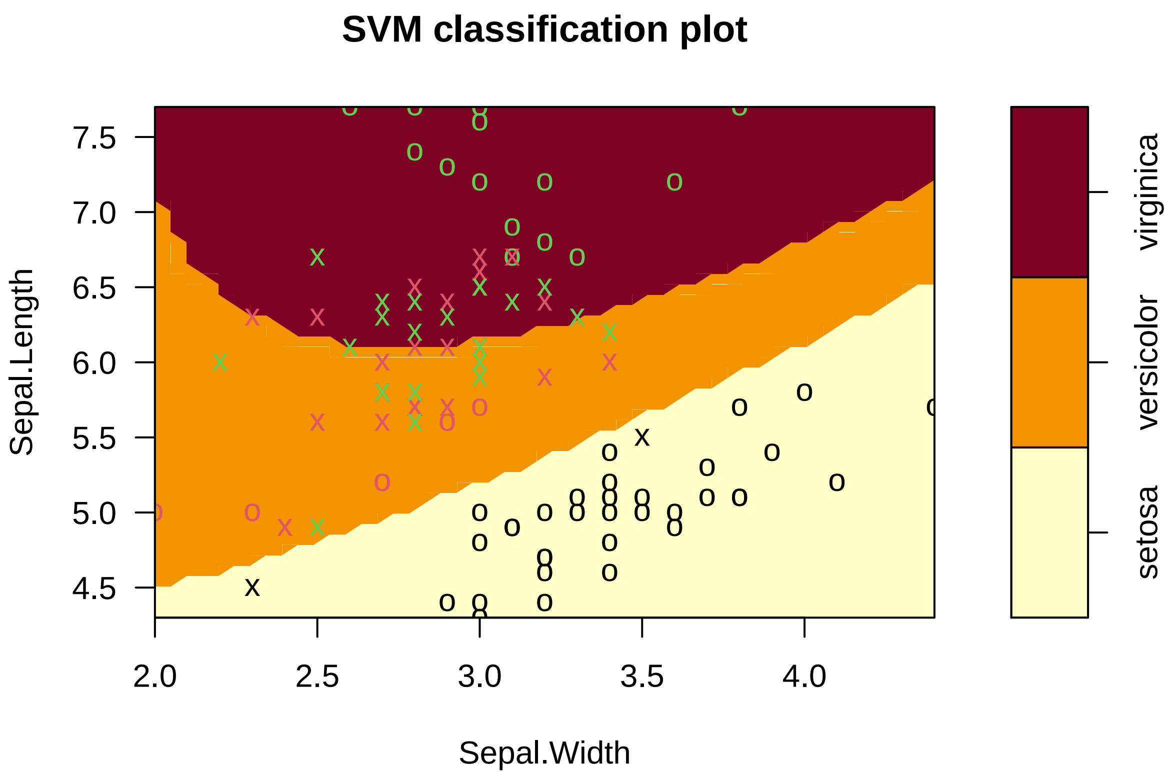


```
s<-sample(150, 100)
sep<-c("Sepal.Length", "Sepal.Width", "Species")
sep_train<-iris[s,sep]
sep_test<-iris[-s,sep]

svm_sepfit <- svm(Species ~., data = sep_train, kernel = "polynomial", cost = 10, scale = FALSE)
print(svm_sepfit)
```

```
##
## Call:
## svm(formula = Species ~ ., data = sep_train, kernel = "polynomial",
##     cost = 10, scale = FALSE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  48
```

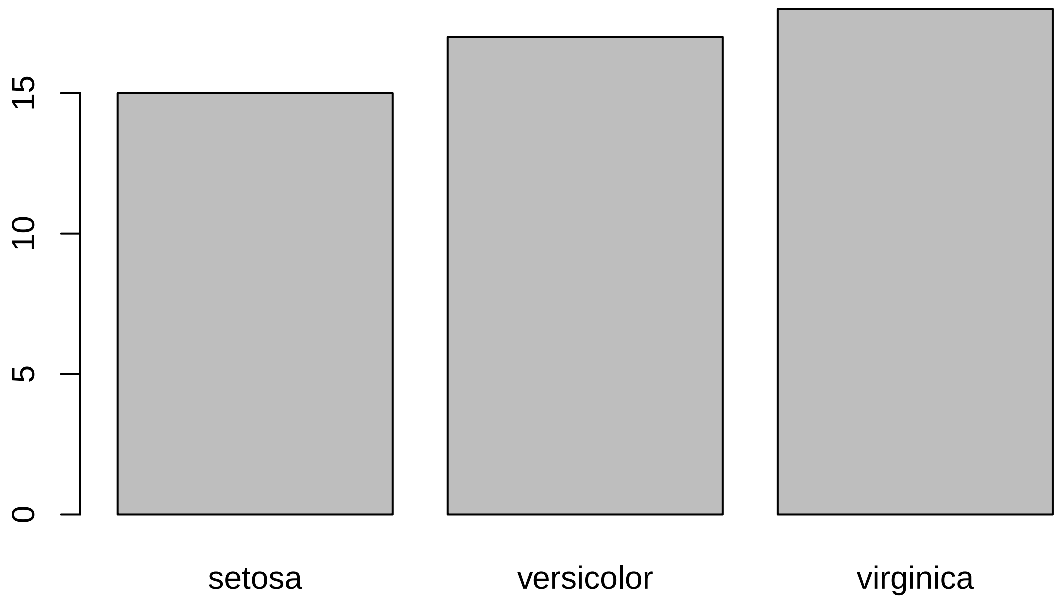
```
plot(svm_sepfit, sep_train[,sep])
```



```
tuned_sep <- tune(svm, Species ~., data = sep_train, kernel = "polynomial", ranges = list(cost=c(0.001,(
# Will show the optimal cost parameter
summary(tuned_sep)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##   100
##
## - best performance: 0.32
##
## - Detailed performance results:
##    cost error dispersion
## 1 1e-03  0.73 0.11595018
## 2 1e-02  0.50 0.13333333
## 3 1e-01  0.42 0.09189366
## 4 1e+00  0.40 0.14142136
## 5 1e+01  0.34 0.08432740
## 6 1e+02  0.32 0.09189366
```

```r
p <- predict(svm_sepfit, sep_test[,sep], type="class")
plot(p)
```



```r
table(p, sep_test[,3])
```

```
##
## p            setosa versicolor virginica
##   setosa         15          0         0
##   versicolor      0         15         2
##   virginica       0          7        11
```

Based on how the the sepal dimensions are, we observe several misclassification using the SVM method. We had to use a higher score for the cost parameter, to allow for greater misclassification compared to prediction via pwtal dimension.