lab11

shichenh

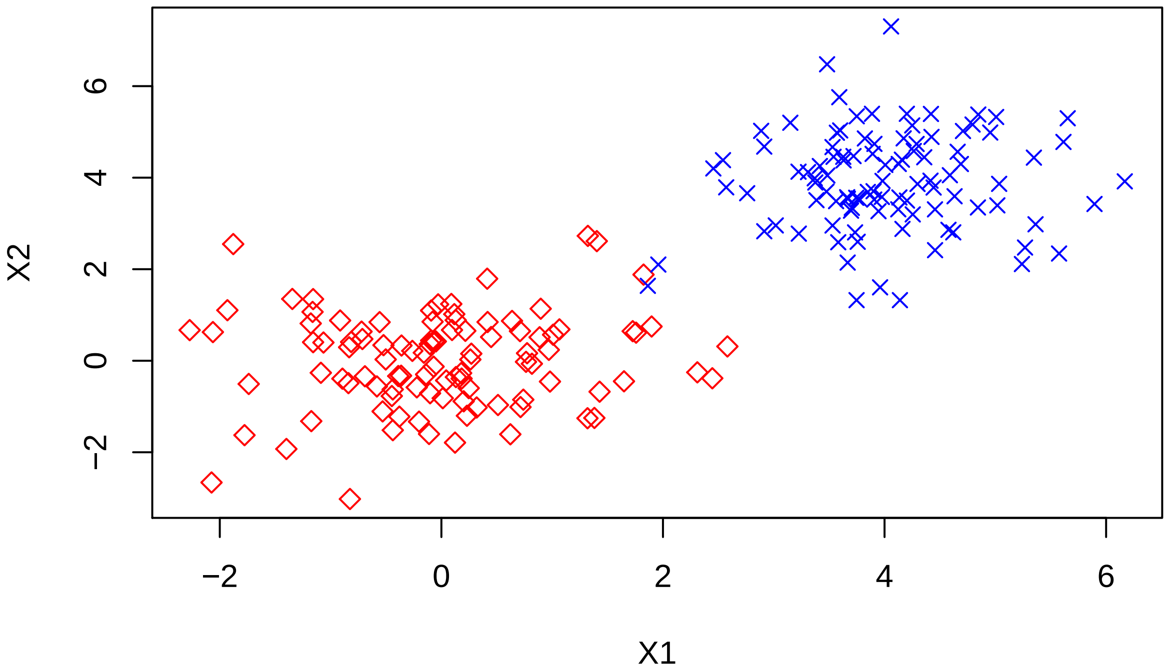
11/12/2017

```
library(kernlab)
library(ROCR)
```

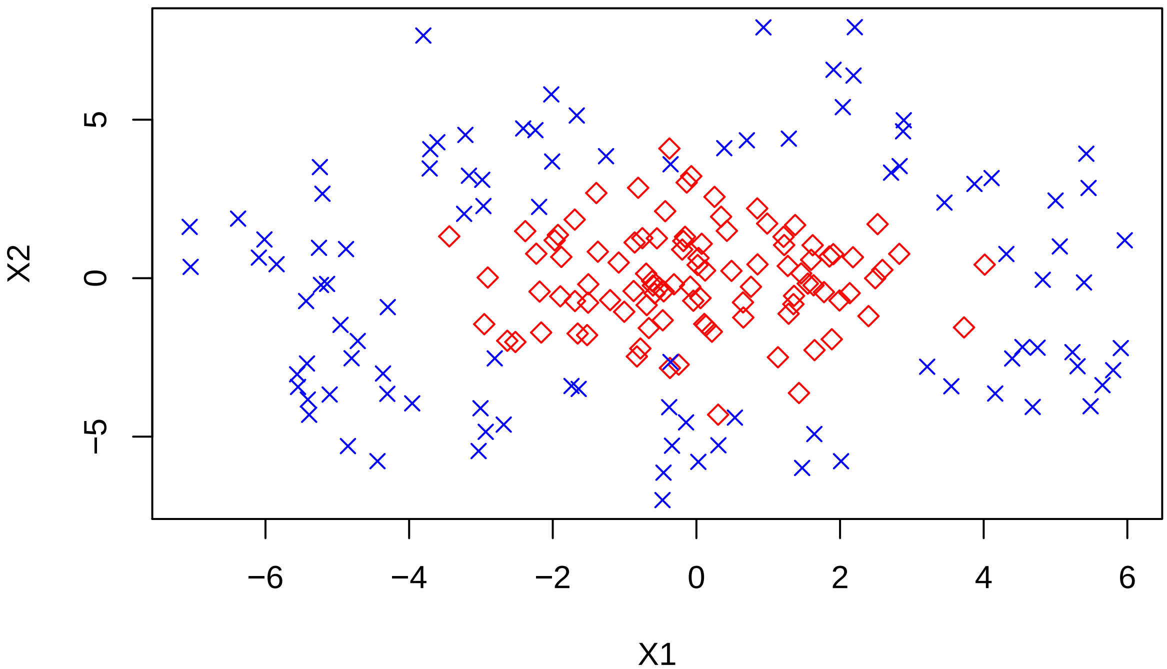
```
set.seed(100)
X1 <- c(rnorm(100), rnorm(100, mean = 4))
X2 <- c(rnorm(100), rnorm(100, mean = 4))
y <- factor(c(rep(0, 100), rep(1, 100)))
df1 <- data.frame(X1, X2, y)

set.seed(200)
r <- c(runif(100, 1, 2), runif(100, 5, 6))
theta <- runif(200, 0, 2 * pi)
X1 <- r * cos(theta) + rnorm(200)
X2 <- r * sin(theta) + rnorm(200)
y <- factor(c(rep(0, 100), rep(1, 100)))
df2 <- data.frame(X1, X2, y)
```

```
pchs <- c(5, 4)
colors <- c("red", "blue")
with(df1, plot(X1, X2, pch = pchs[y], col=colors[y]))
```




```
pchs <- c(5, 4)
colors <- c("red", "blue")
with(df2, plot(X1, X2, pch = pchs[y], col=colors[y]))
```

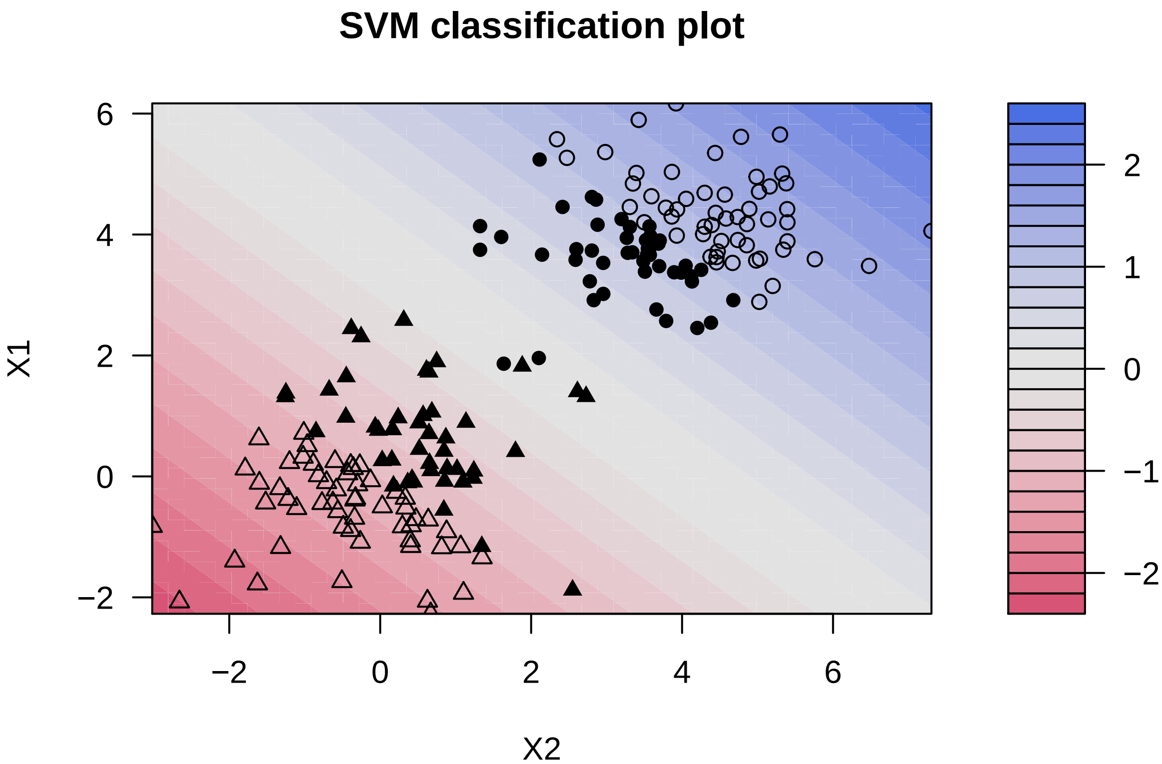


Bulding SVM

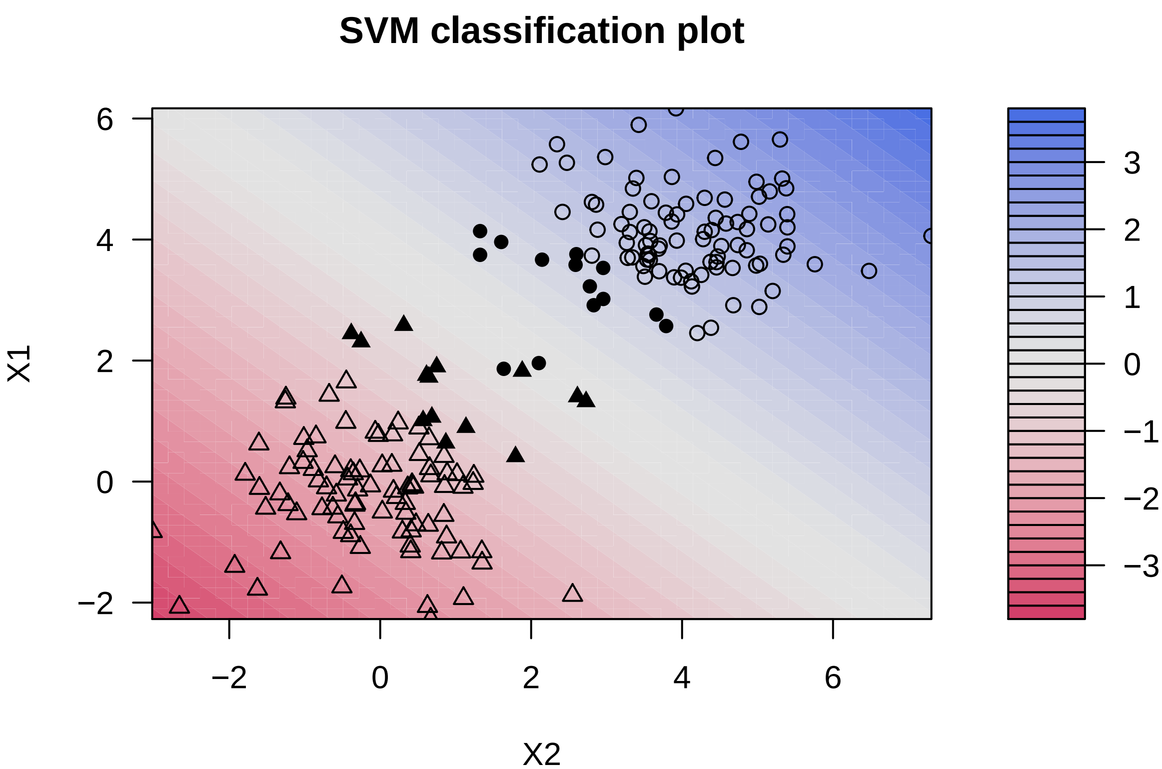
```
C_vector <- c(0.01, 0.1, 1, 10, 100, 1000, 10000)
dataset_list <- list(df1, df2)

for (df in dataset_list) {
  for (C in C_vector) {
    fit <- ksvm(y~X1+X2, data=df, kernel = "vanilladot", C=C)
    plot(fit, data=df)
  }
}
```

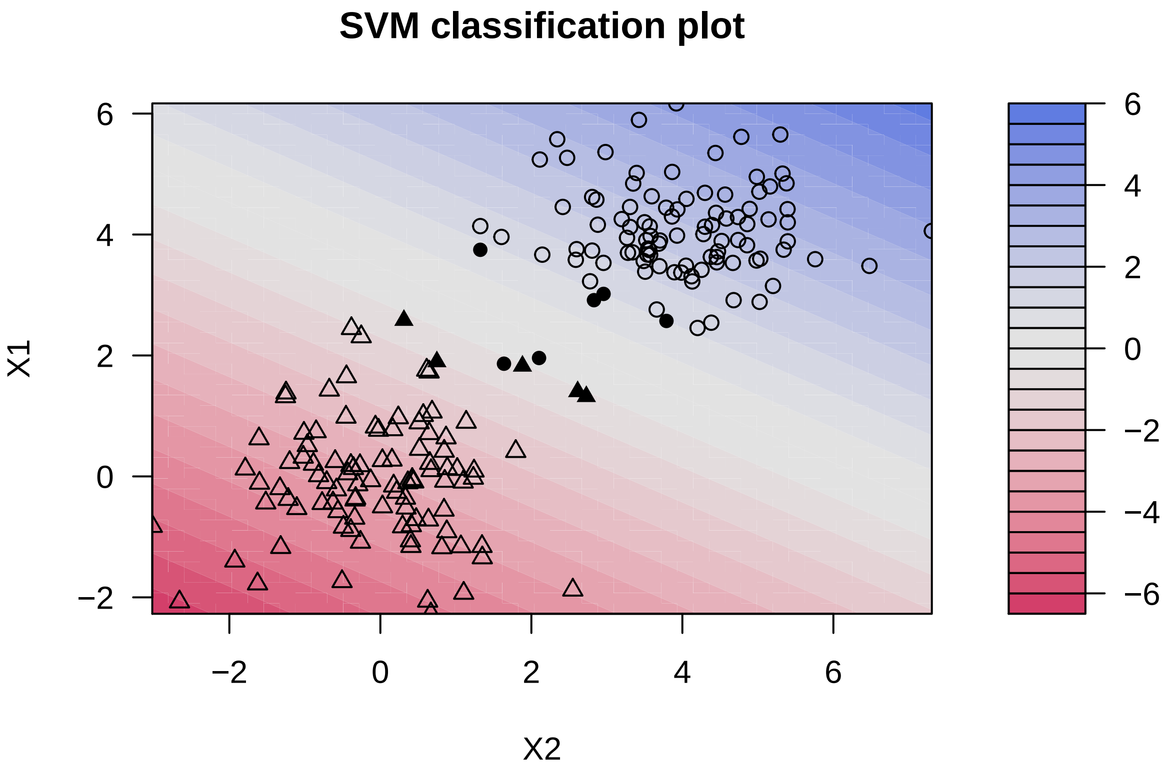
```
##  Setting default kernel parameters
```



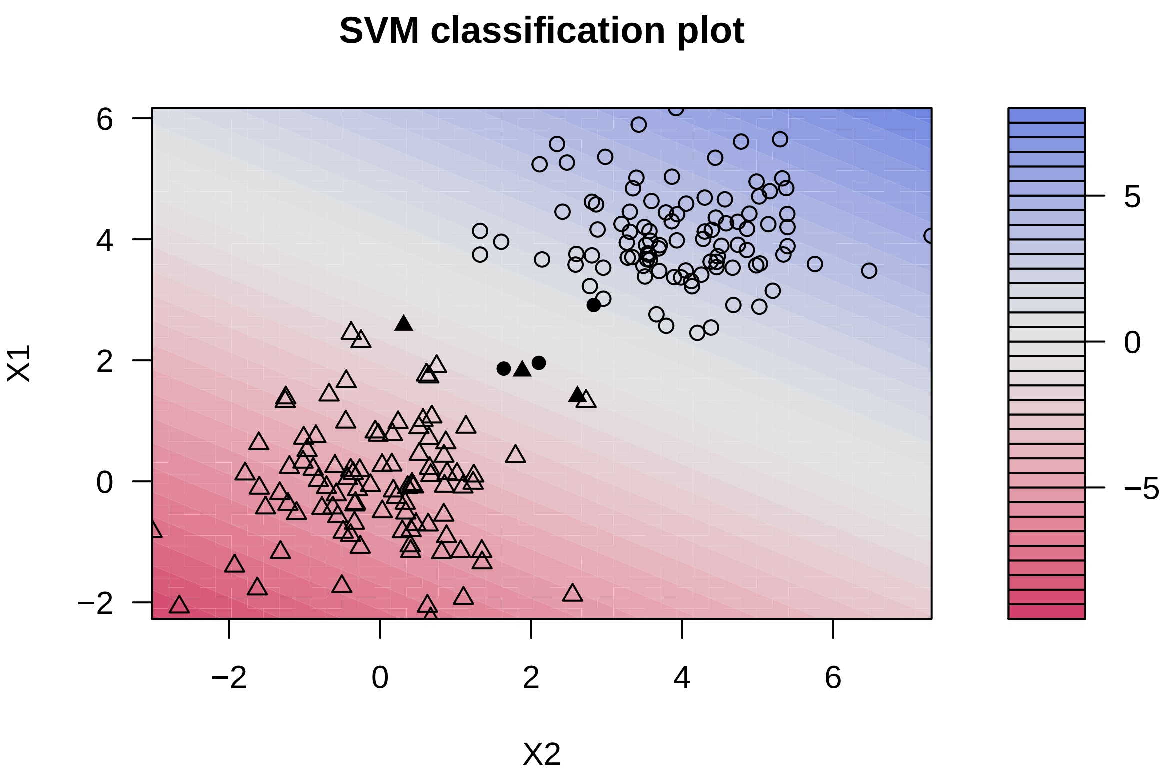

```
##  Setting default kernel parameters
```




```
##  Setting default kernel parameters
```

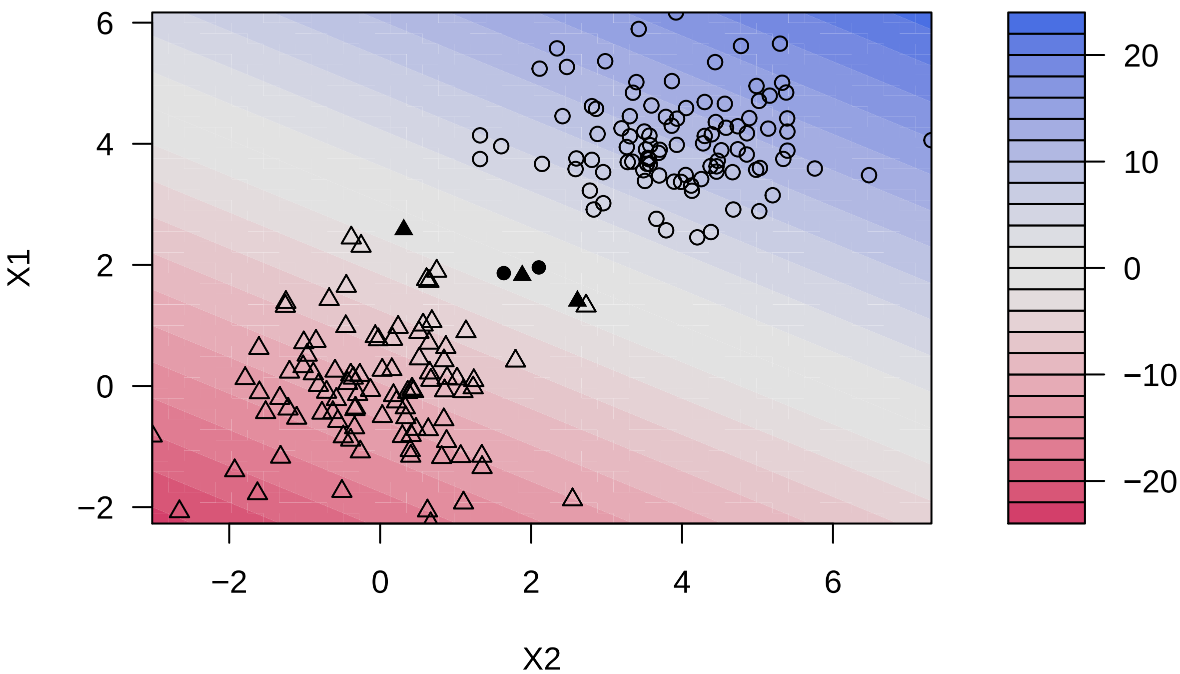



```
##  Setting default kernel parameters
```



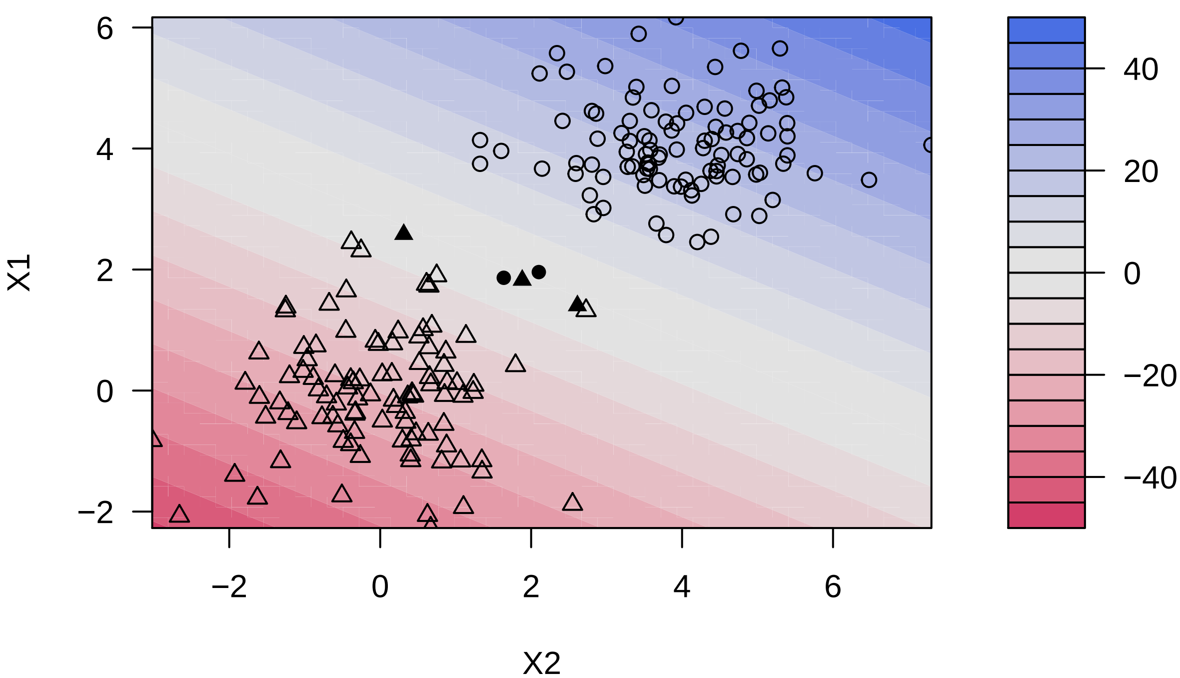

```
##  Setting default kernel parameters
```

SVM classification plot

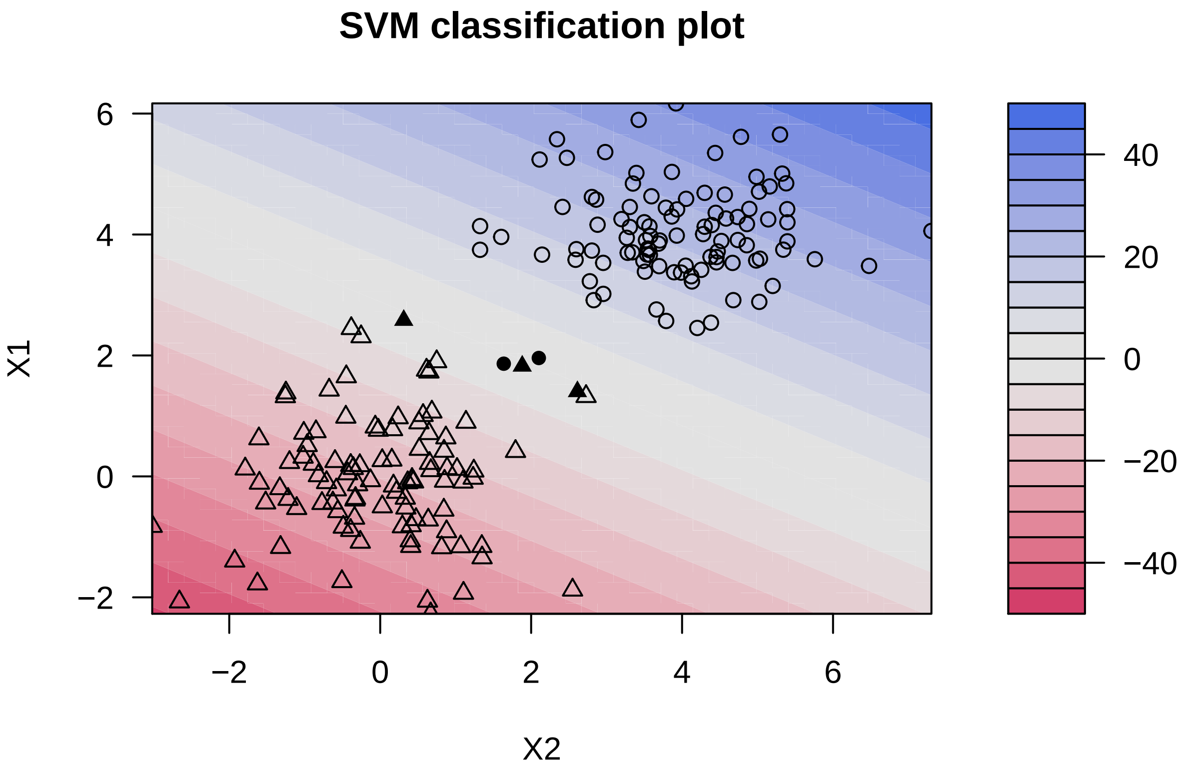



```
## Setting default kernel parameters
```

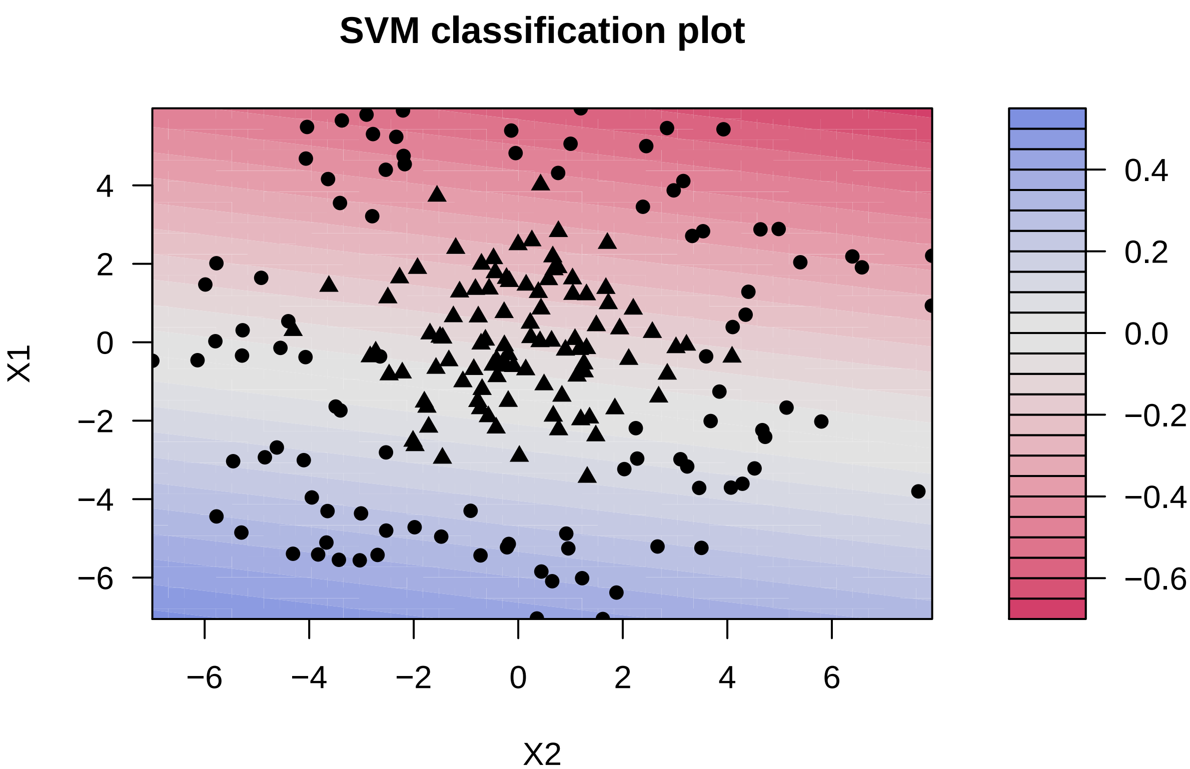
SVM classification plot



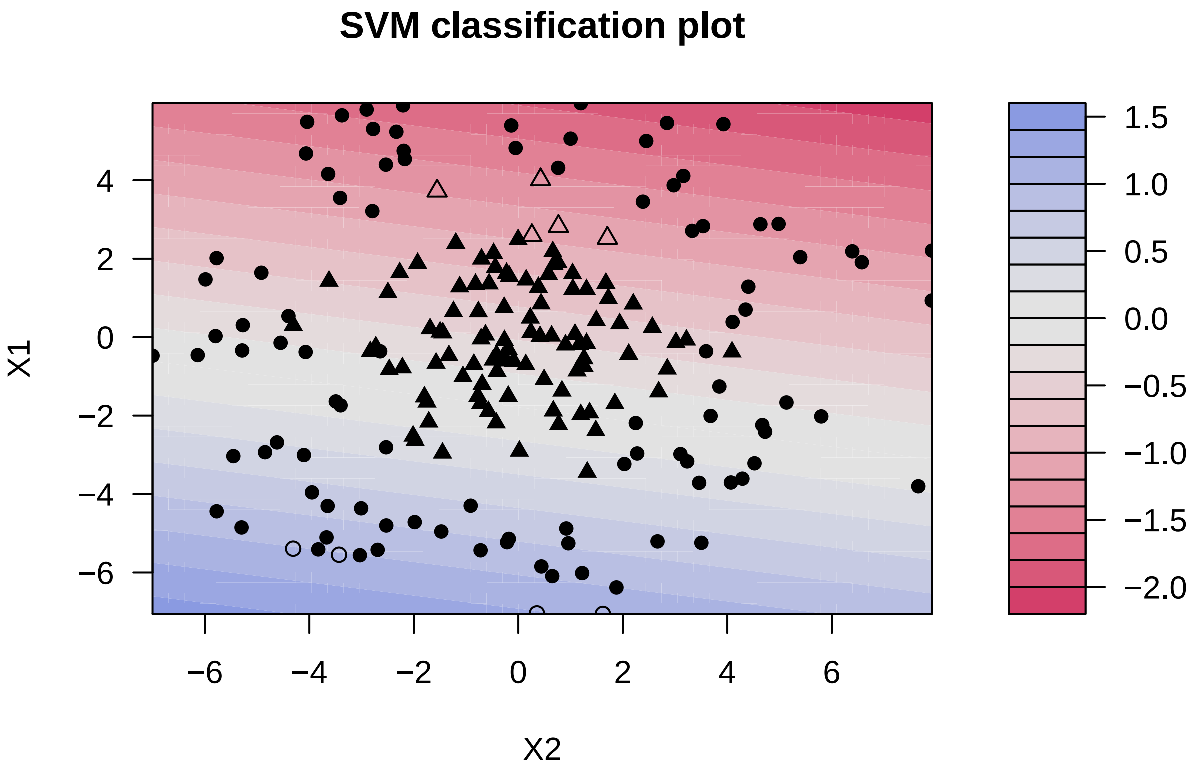

```
## Setting default kernel parameters
```



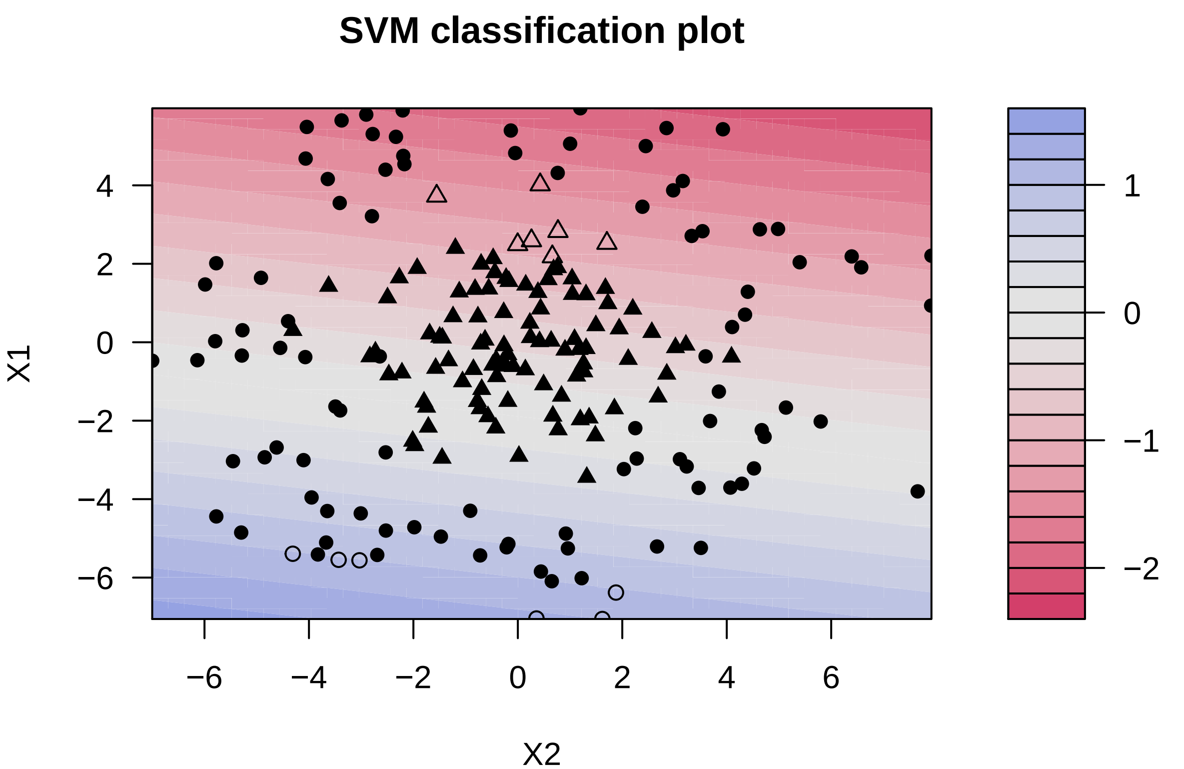

```
## Setting default kernel parameters
```



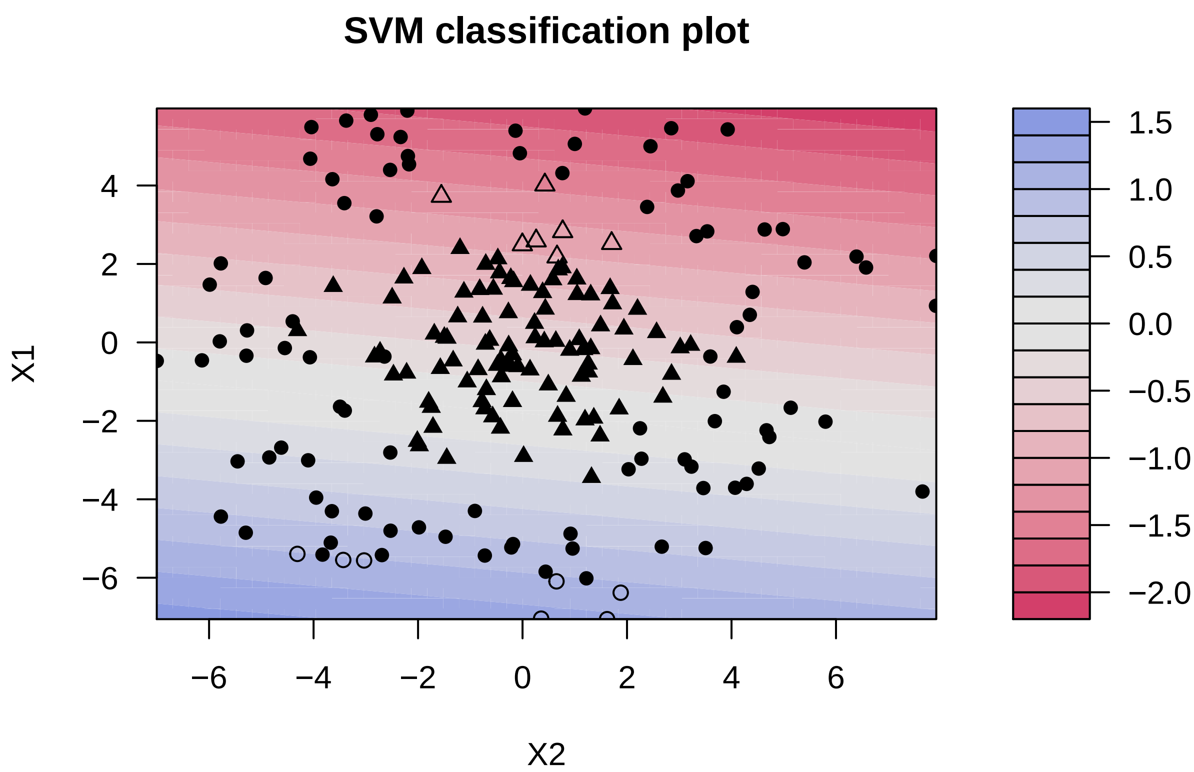

```
## Setting default kernel parameters
```



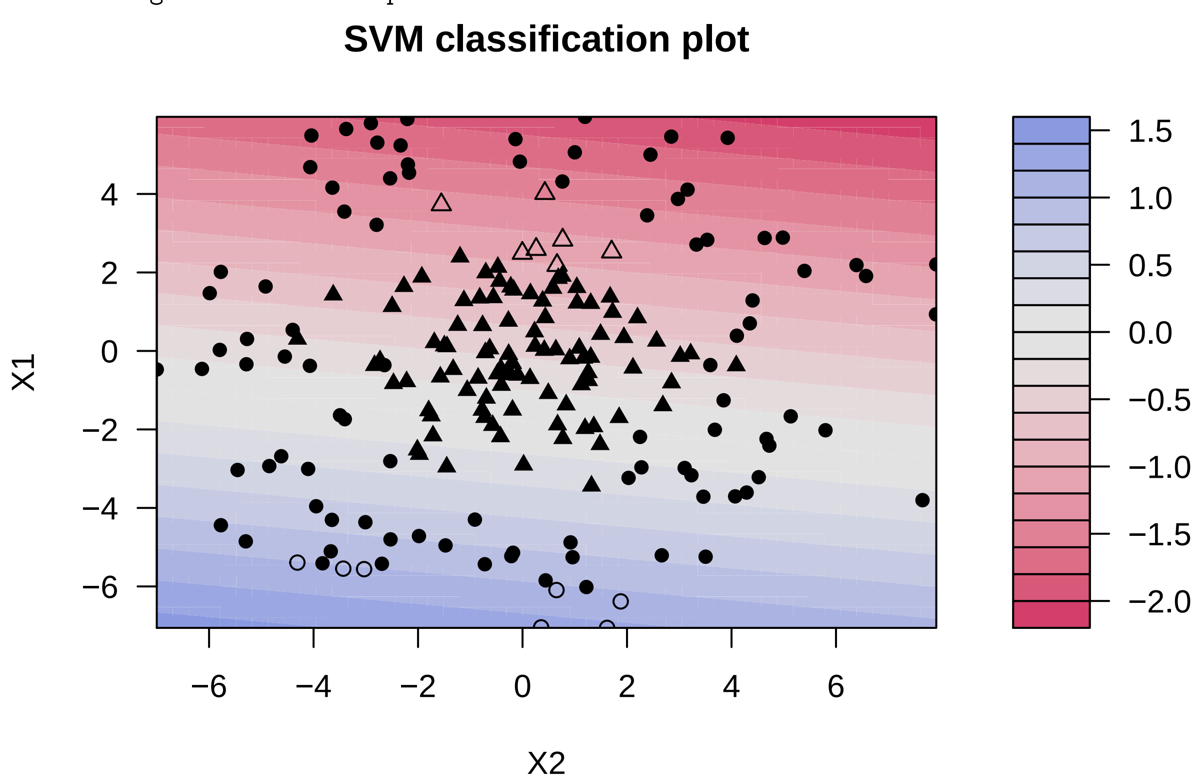

```
## Setting default kernel parameters
```



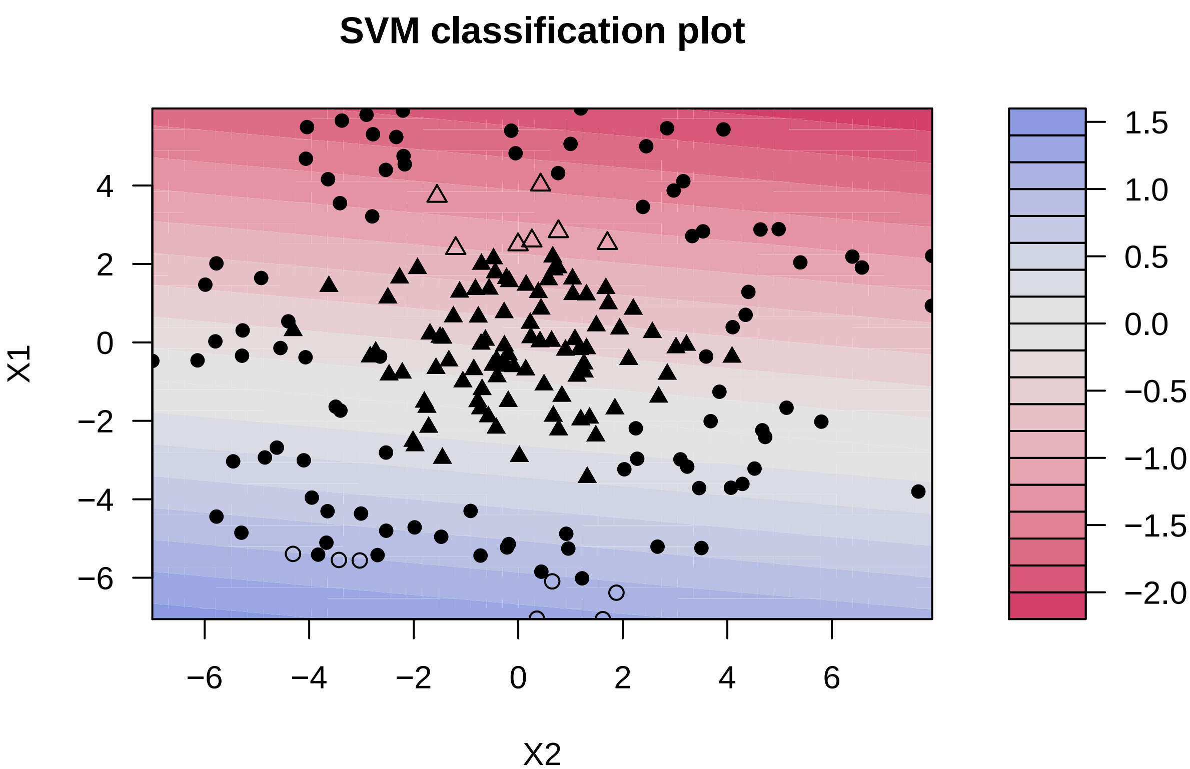

```
## Setting default kernel parameters
```



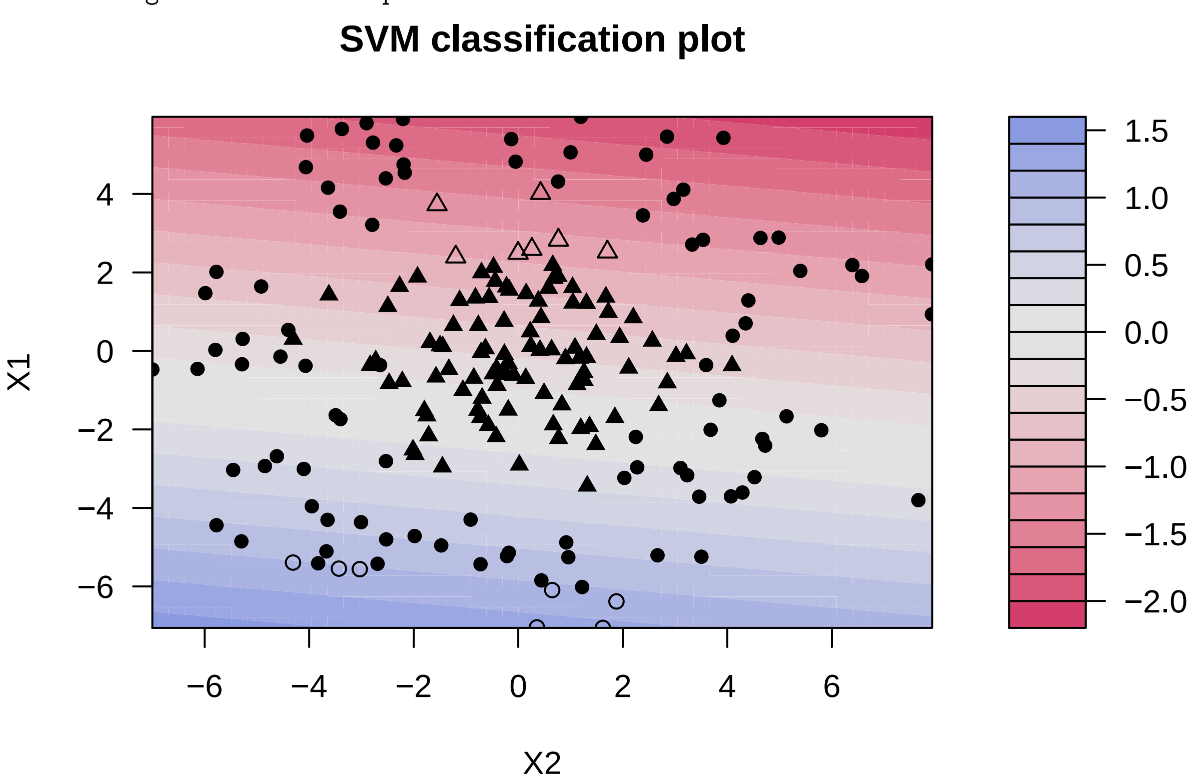

```
##  Setting default kernel parameters
```




```
##  Setting default kernel parameters
```



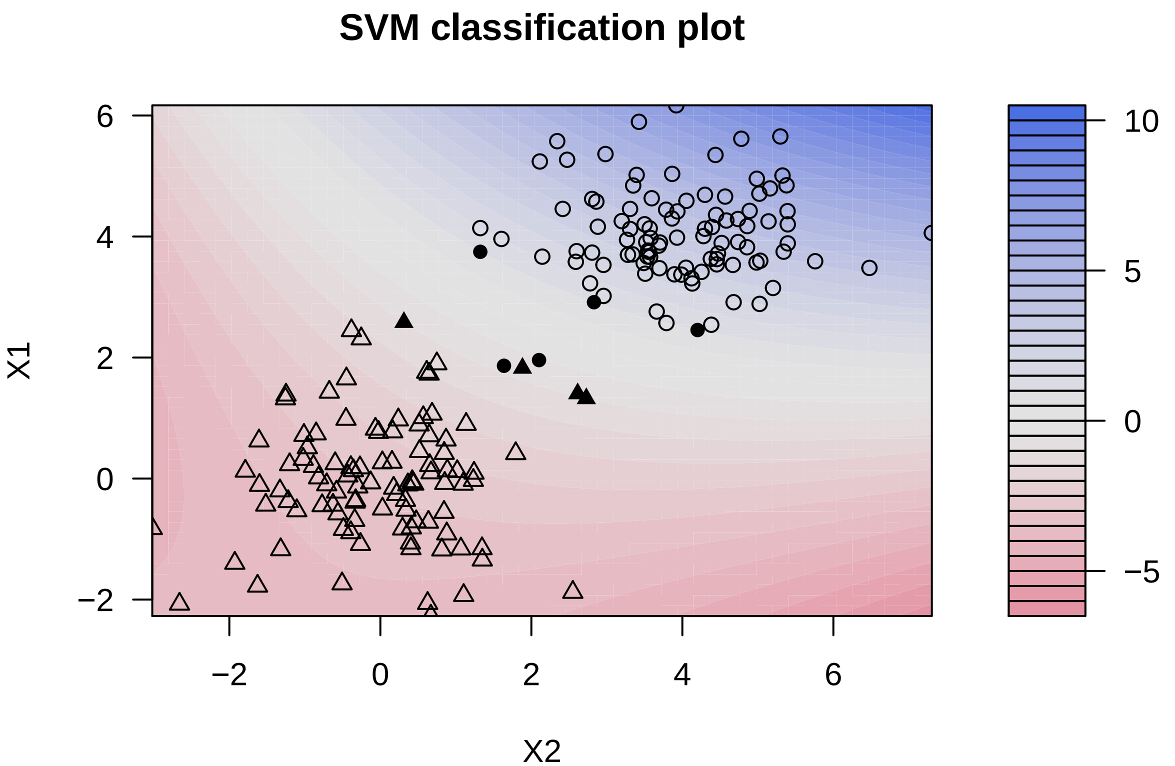

```
##  Setting default kernel parameters
```

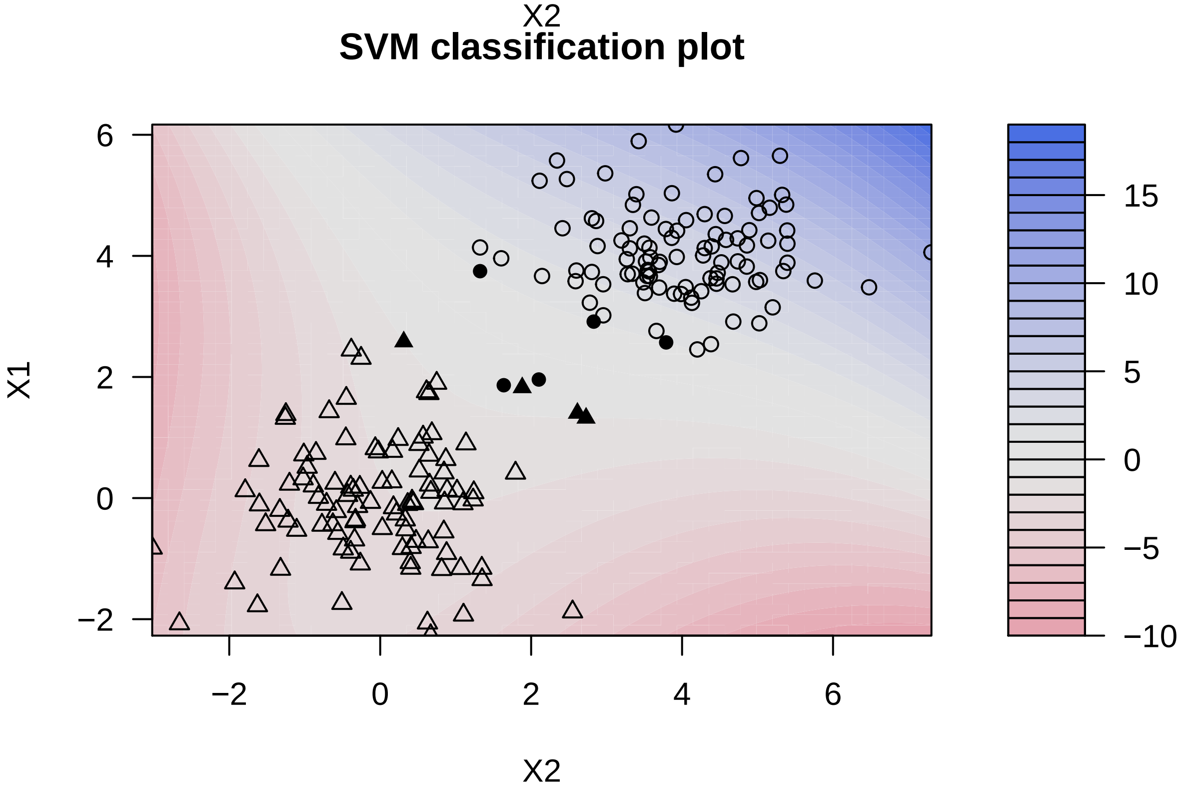


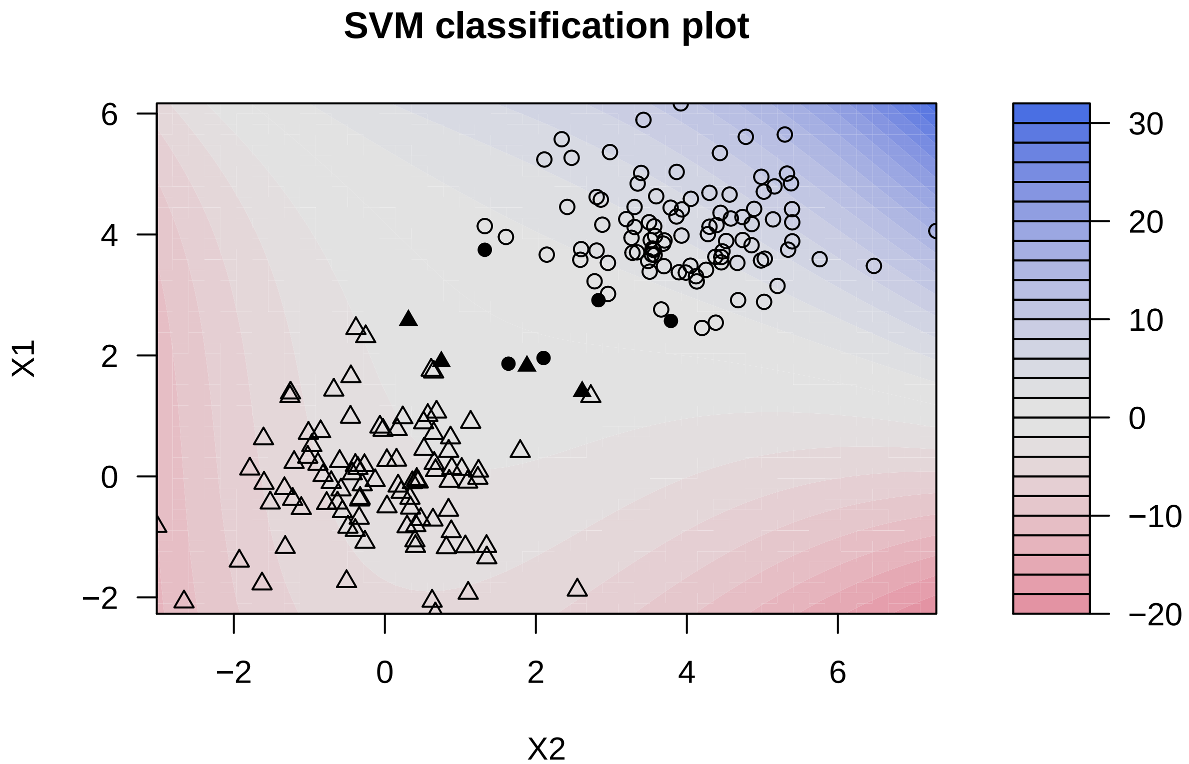

```
deg_vector <- 2:5
gam_vector <- c(0.01, 0.1, 1, 10, 100, 1000, 10000)

for (df in dataset_list) {
```

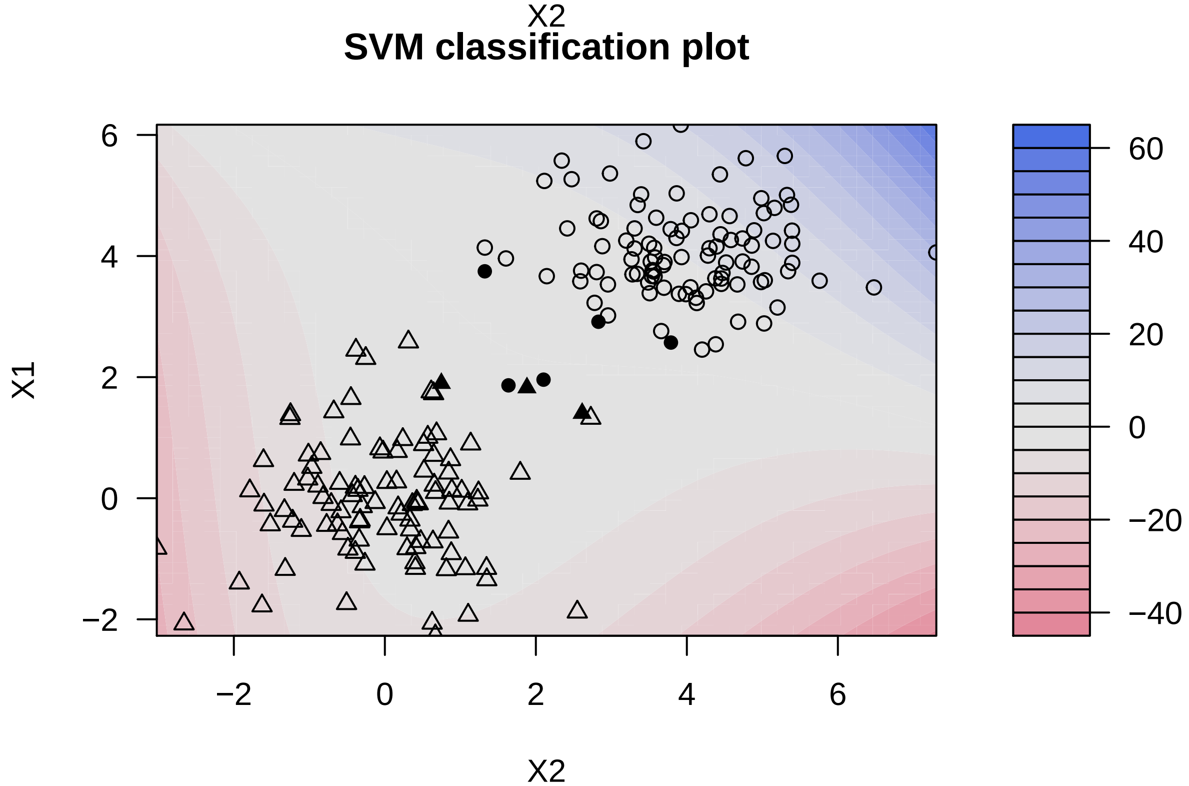
```
  for (deg in deg_vector) {
    fit <- ksvm(y~X1+X2, data = df, kernel = "polydot", kpar=list(degree=deg))
    plot(fit, data=df)
  }
}
```



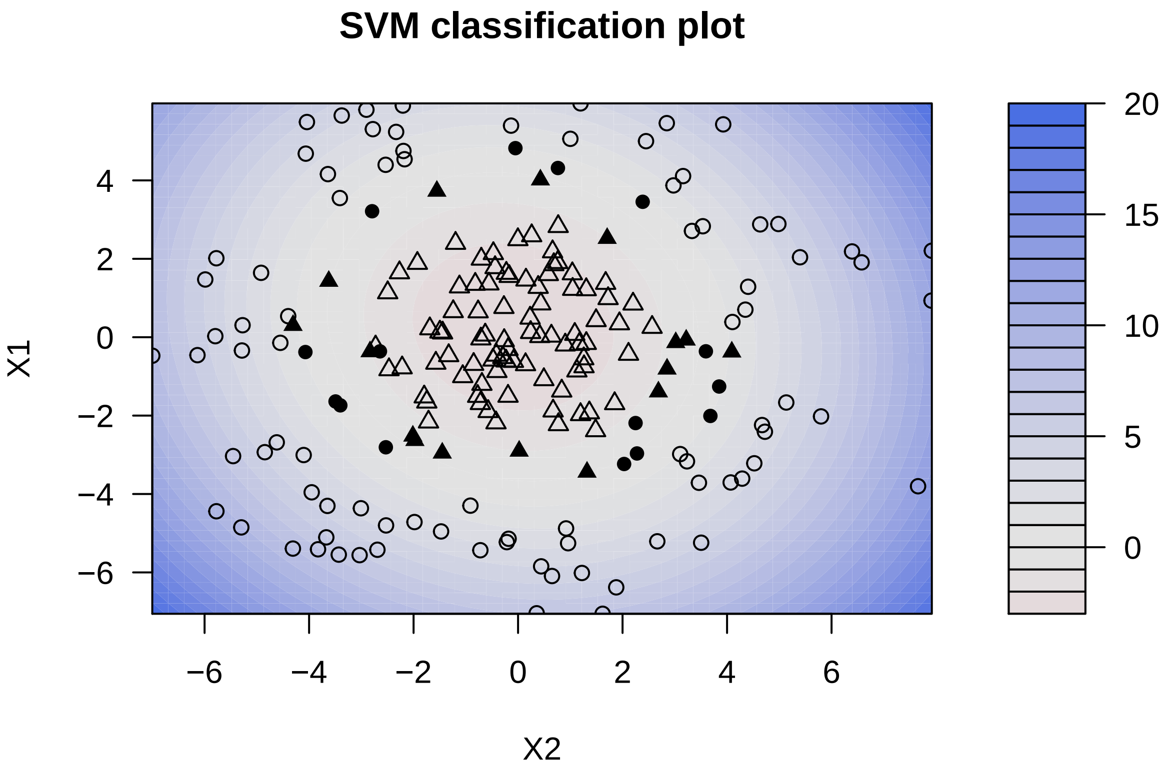


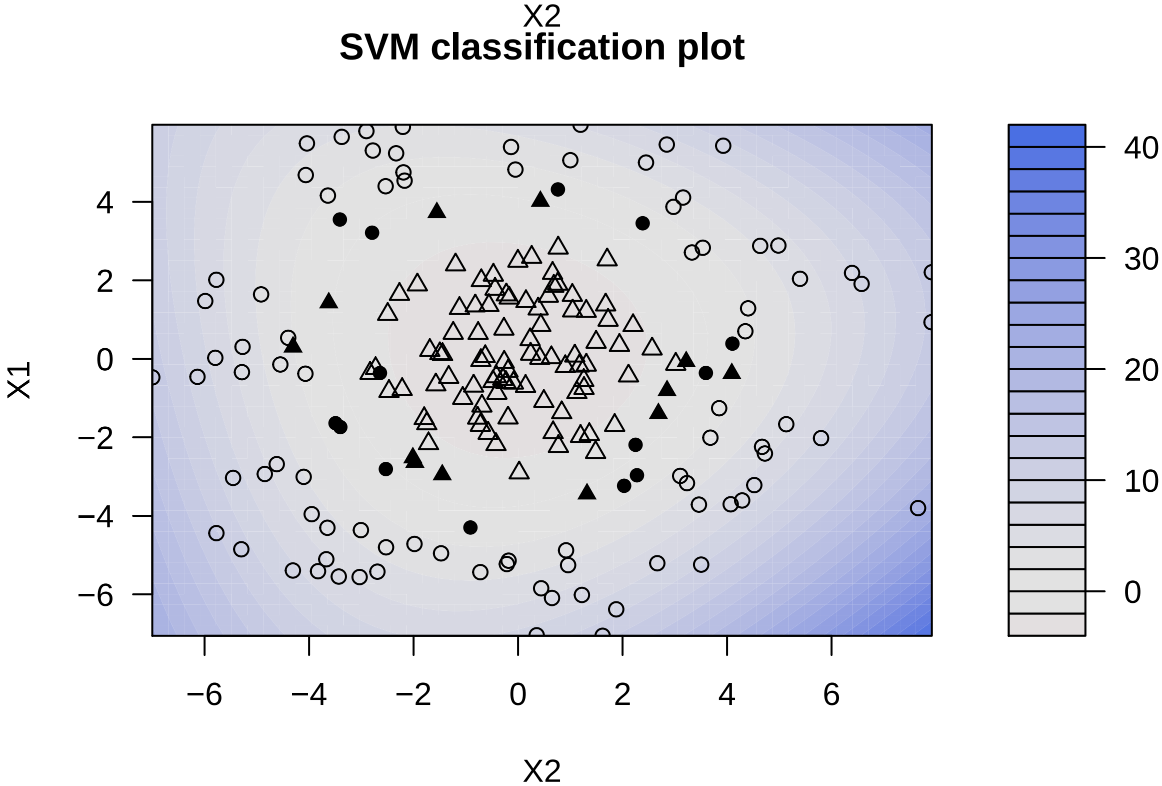


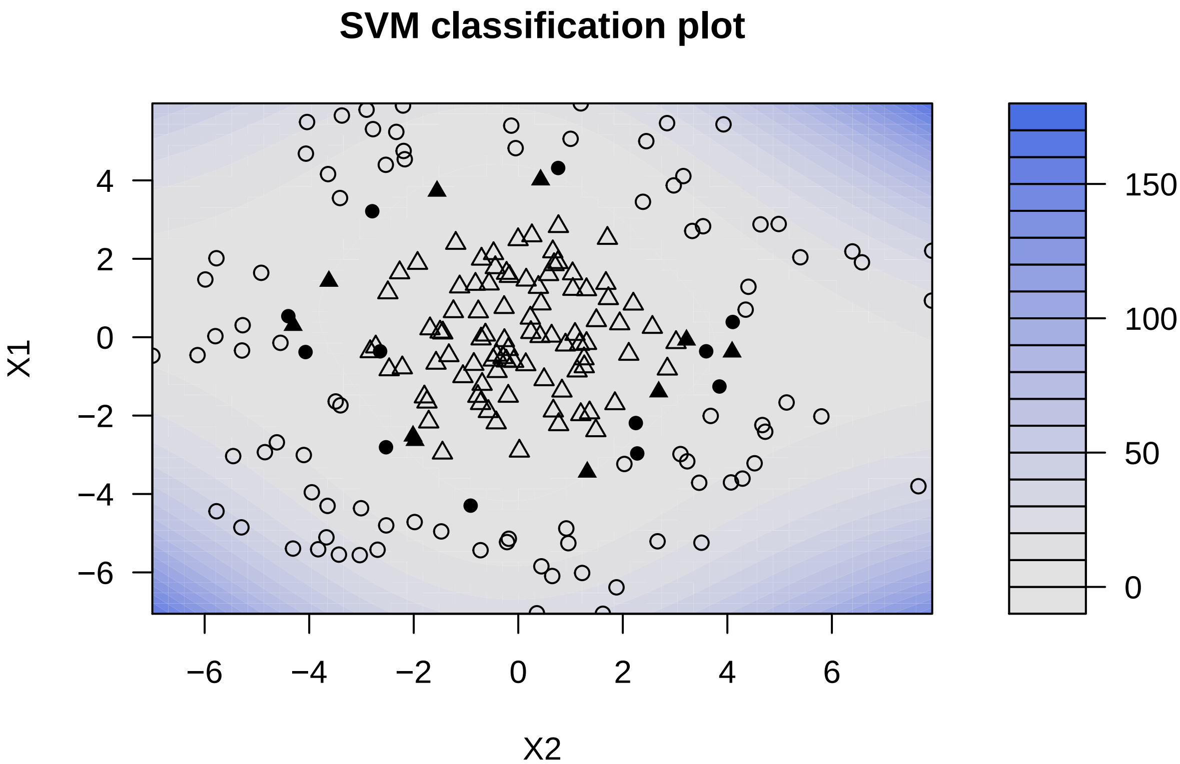
SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
30
20
10
0
−10
−20

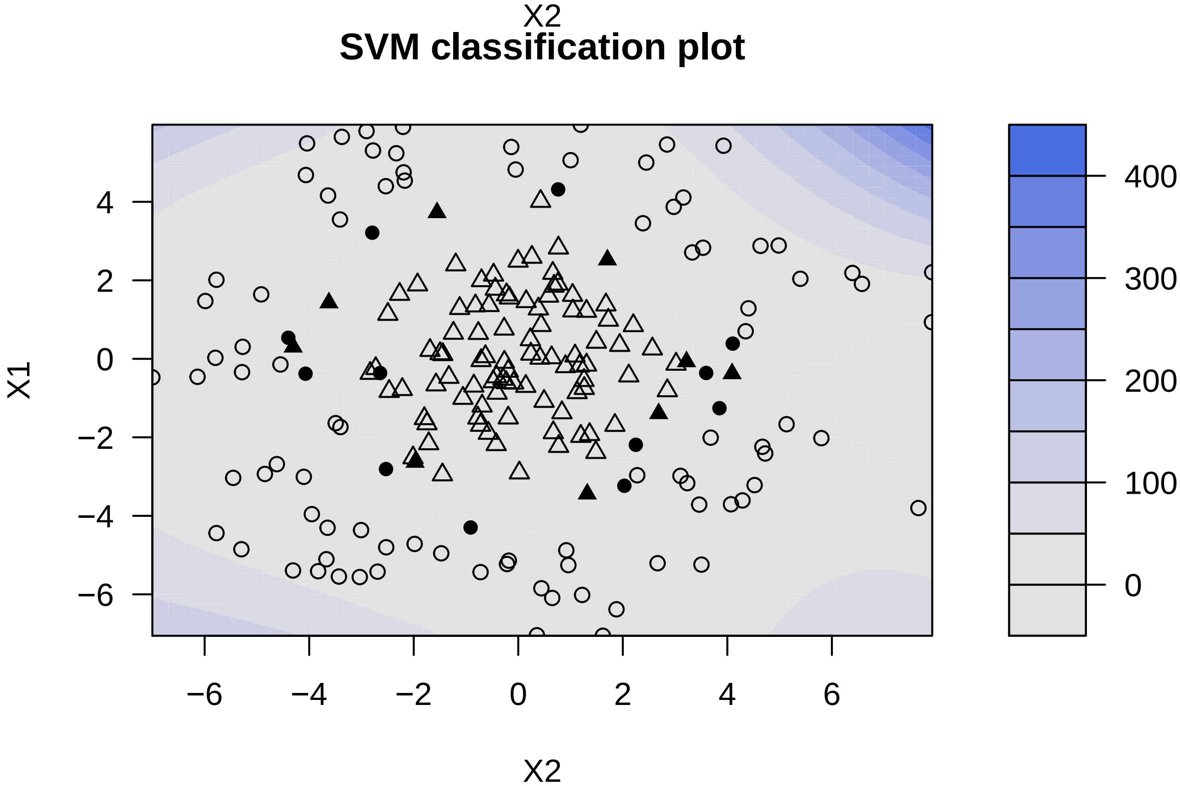


SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
60
40
20
0
−20
−40

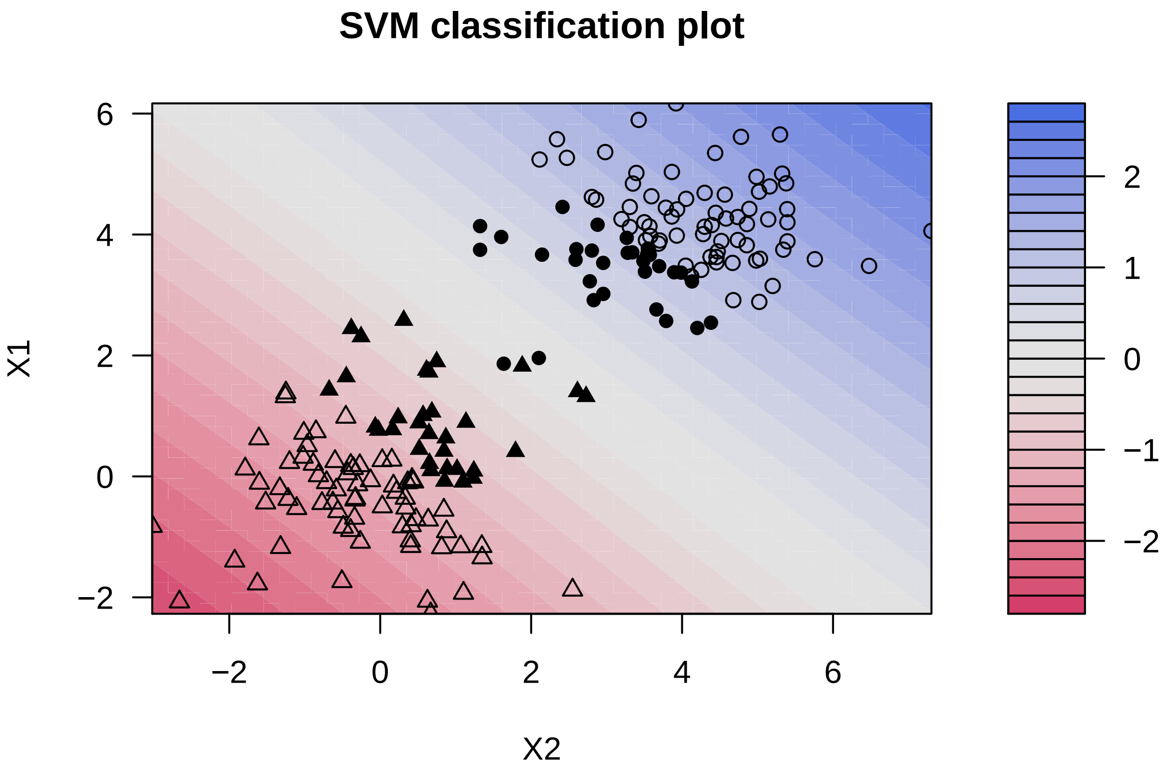

SVM classification plot
X1
X2
−6
−4
−2
0
2
4
6
20
15
10
5
0


SVM classification plot
X1
X2
−6
−4
−2
0
2
4
6
40
30
20
10
0

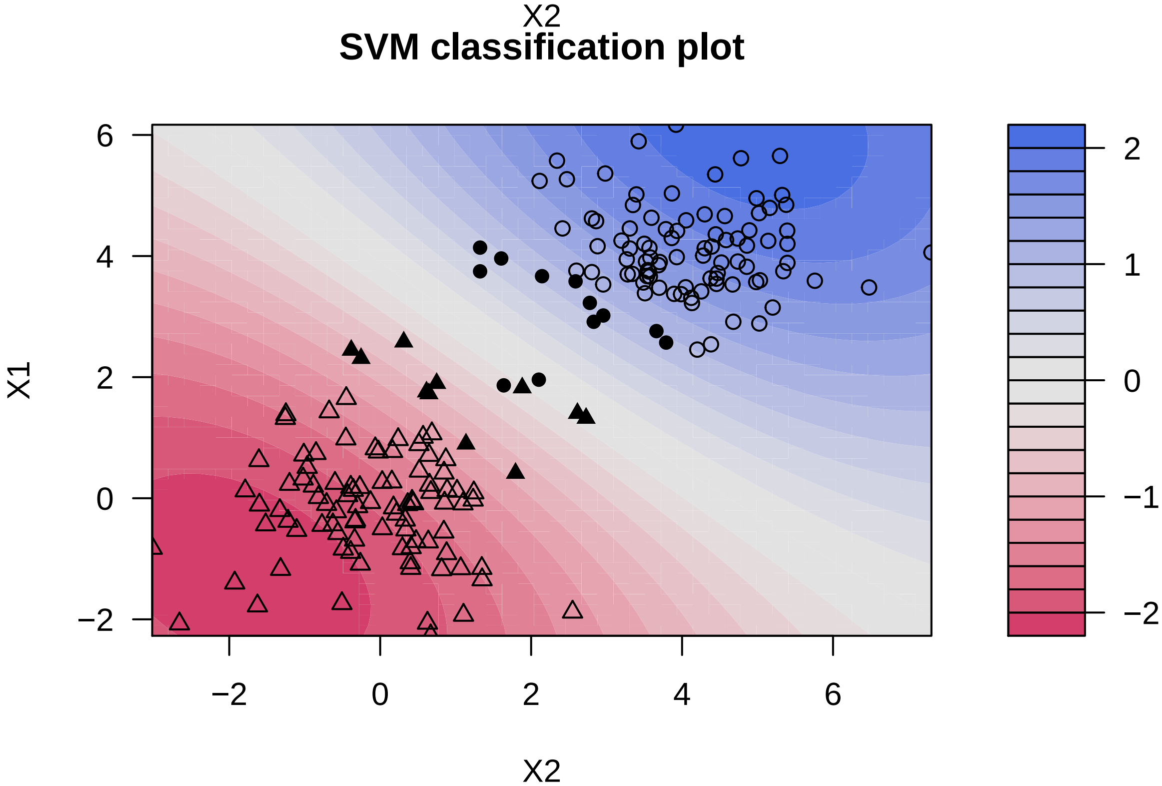




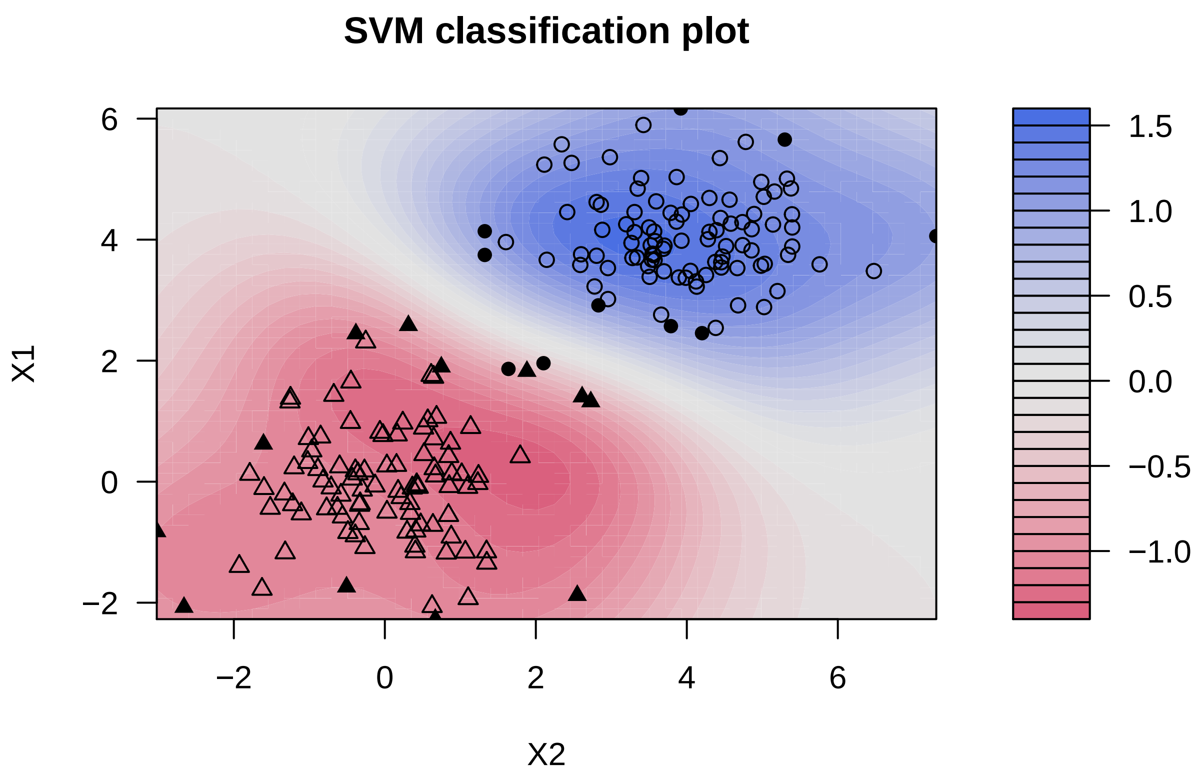

```
for (df in dataset_list) {
  for (gam in gam_vector) {
    fit <- ksvm(y~X1+X2, data = df, kernel = "rbfdot", kpar=list(sigma=gam))
    plot(fit, data=df)
  }
}
```

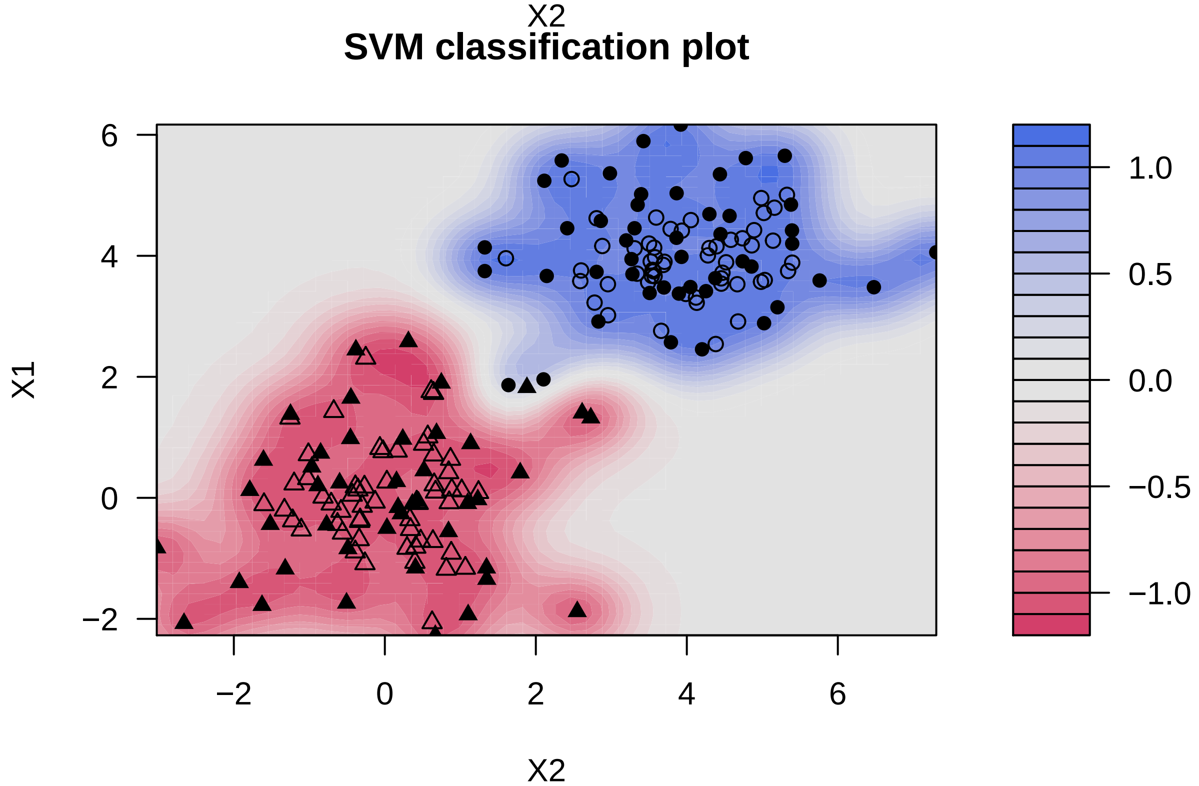


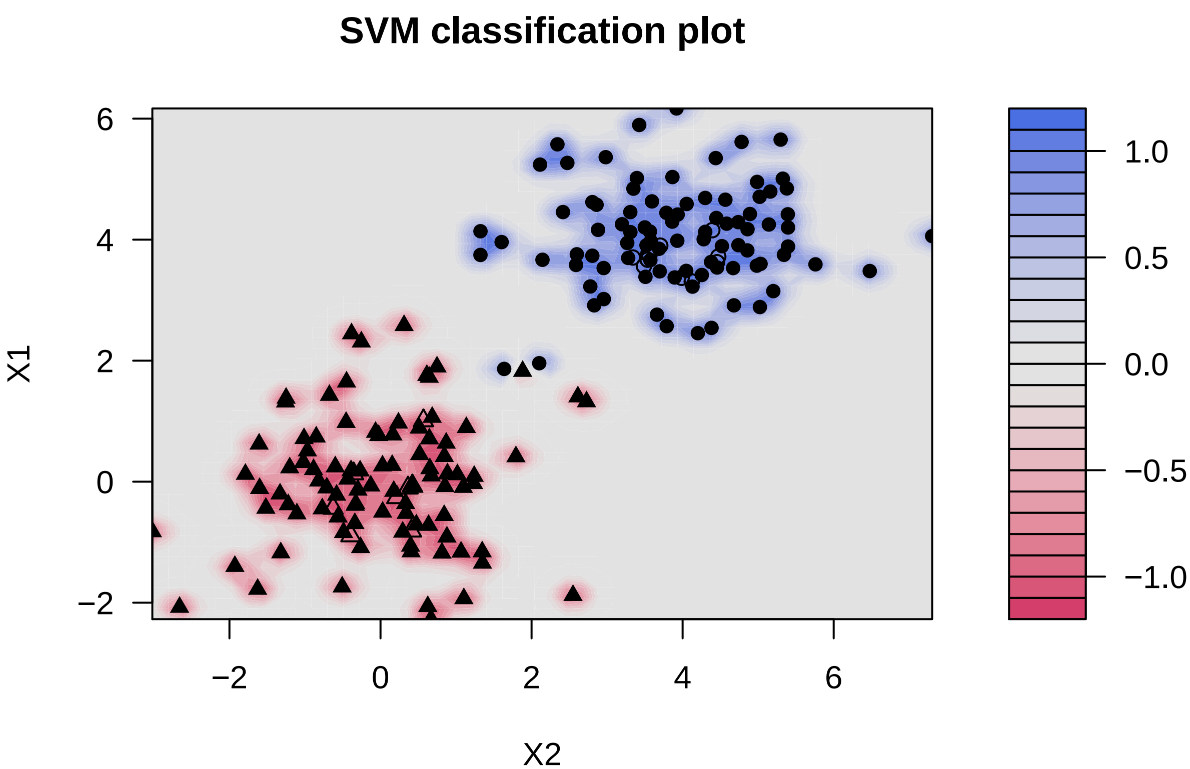
SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
2
1
0
−1
−2

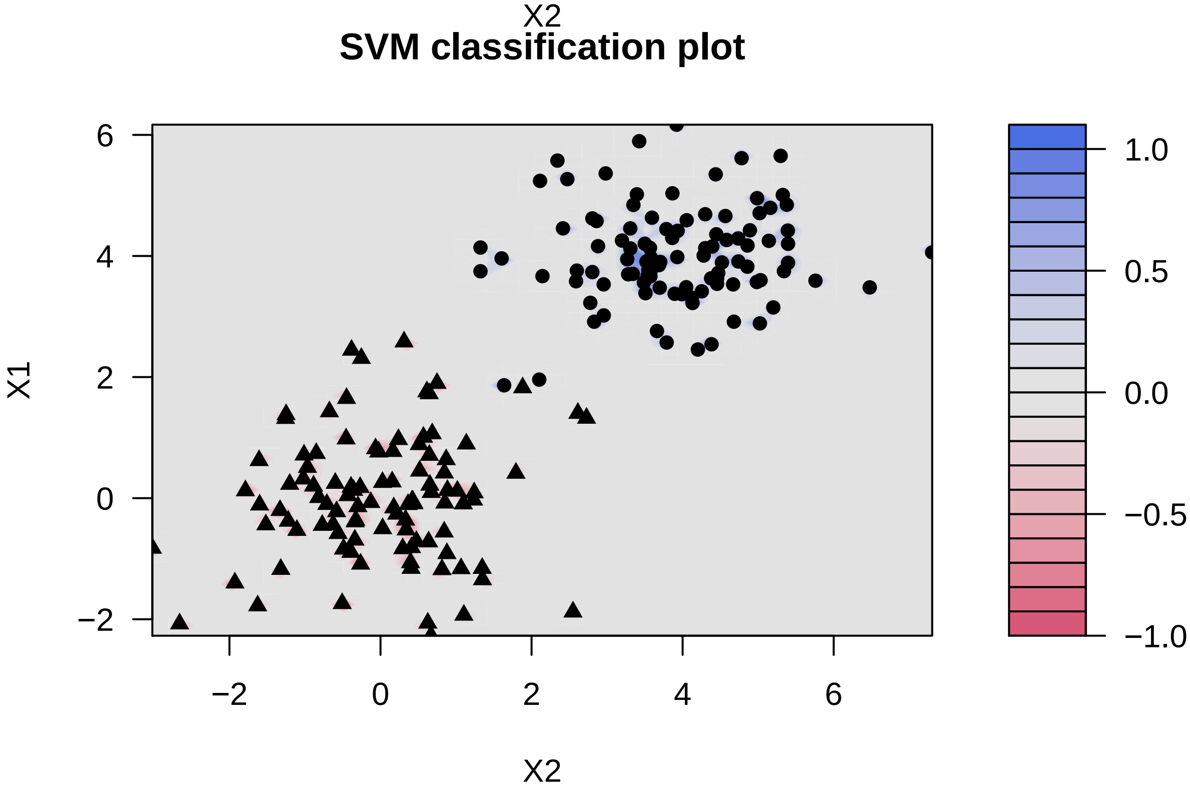


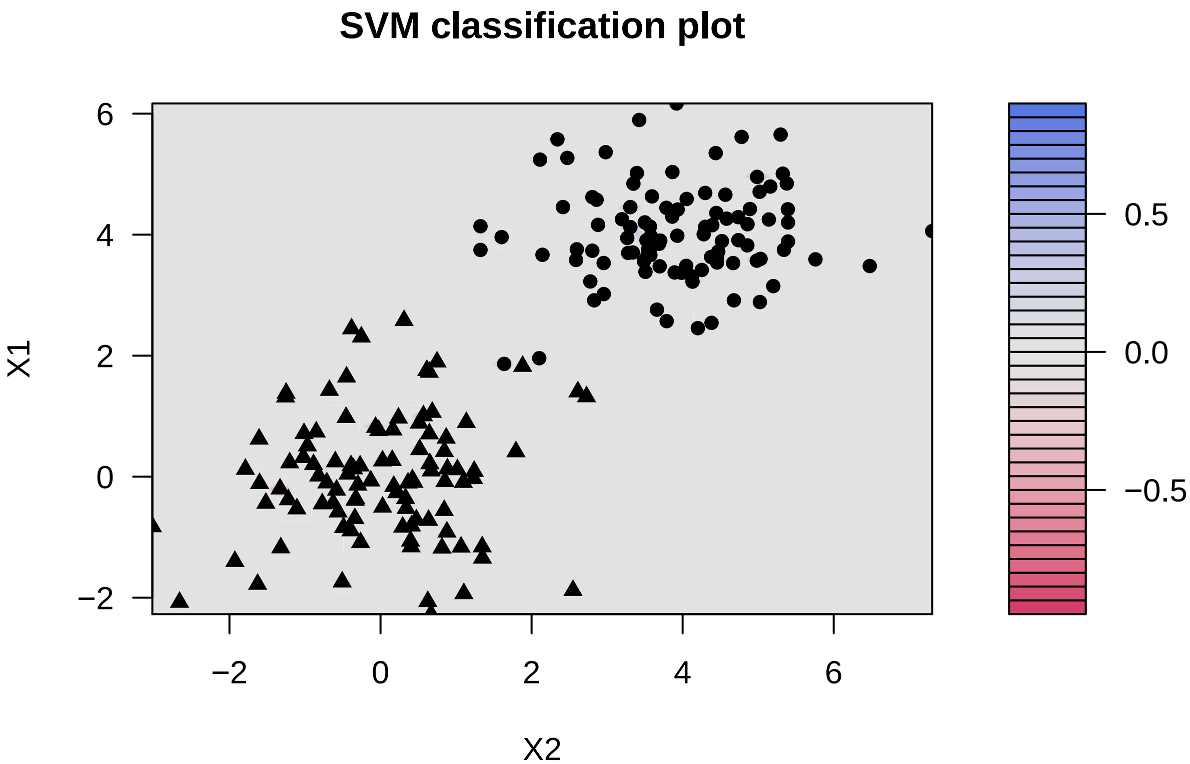
SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
2
1
0
−1
−2

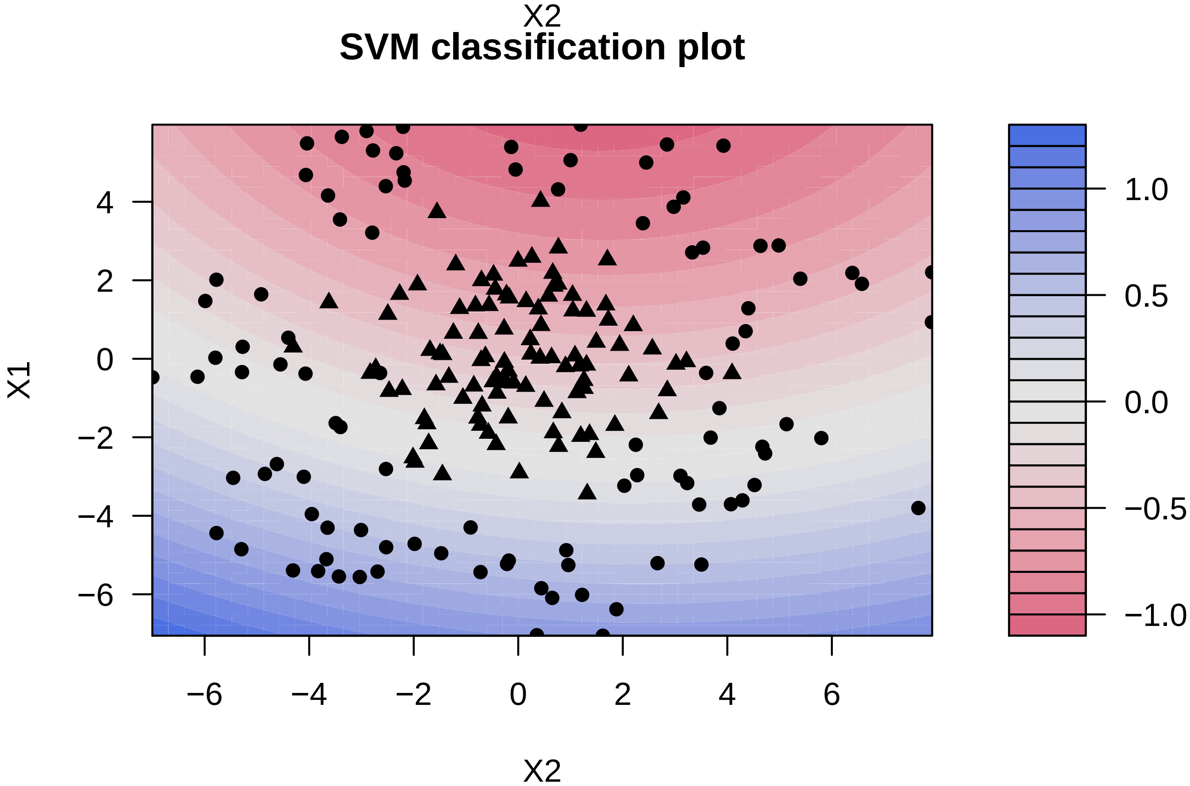

SVM classification plot
6
4
2
0
−2
X1
−2
0
2
4
6
X2
1.5
1.0
0.5
0.0
−0.5
−1.0

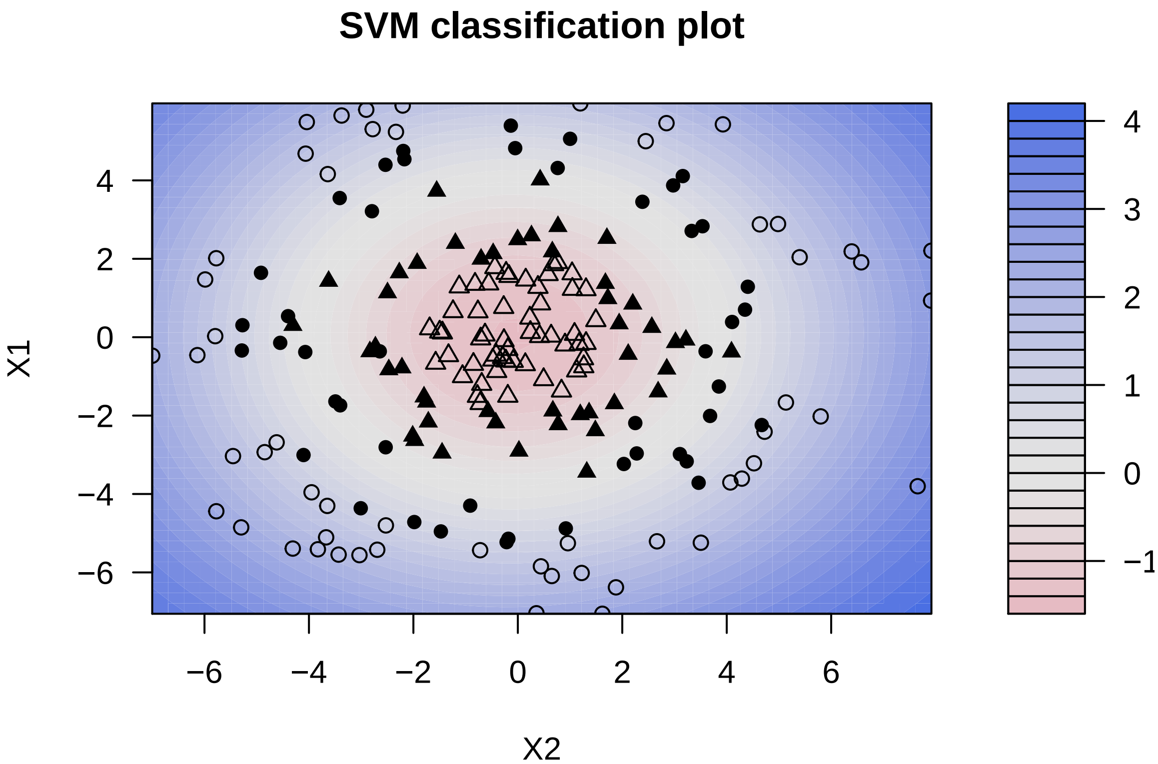

SVM classification plot
6
4
2
0
−2
X1
−2
0
2
4
6
X2
1.0
0.5
0.0
−0.5
−1.0

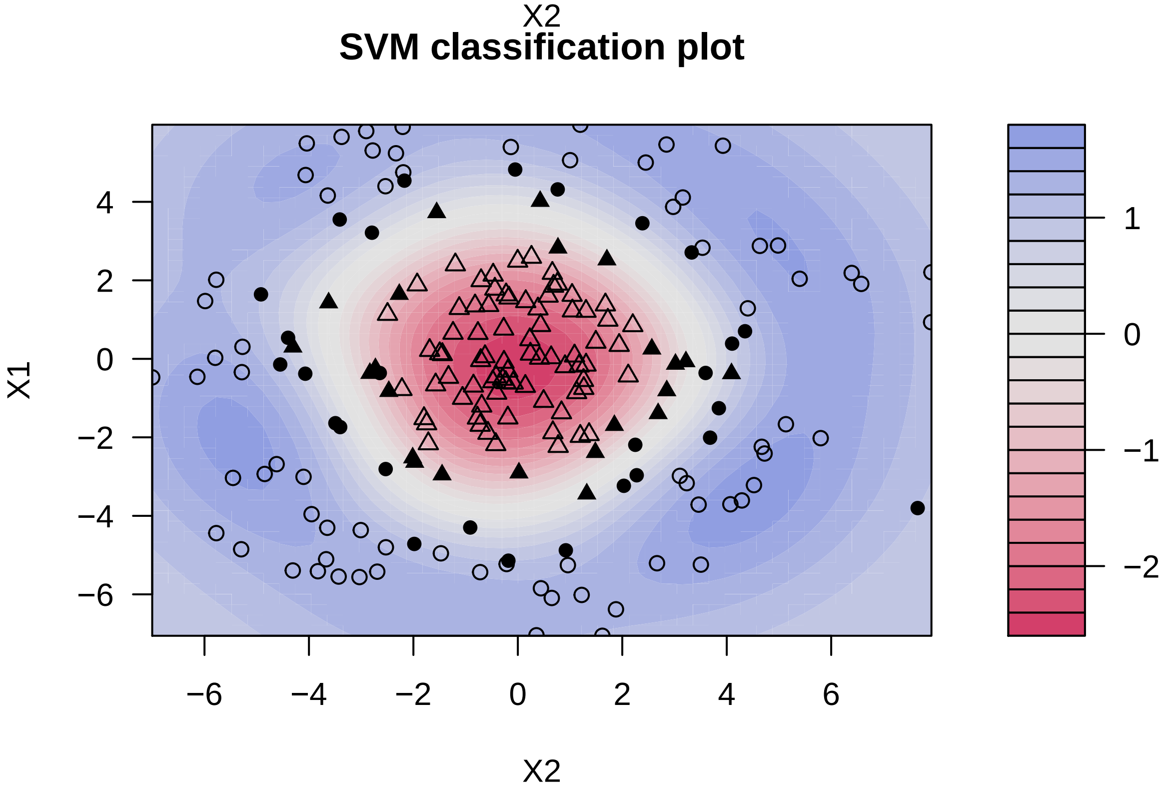

SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
1.0
0.5
0.0
−0.5
−1.0

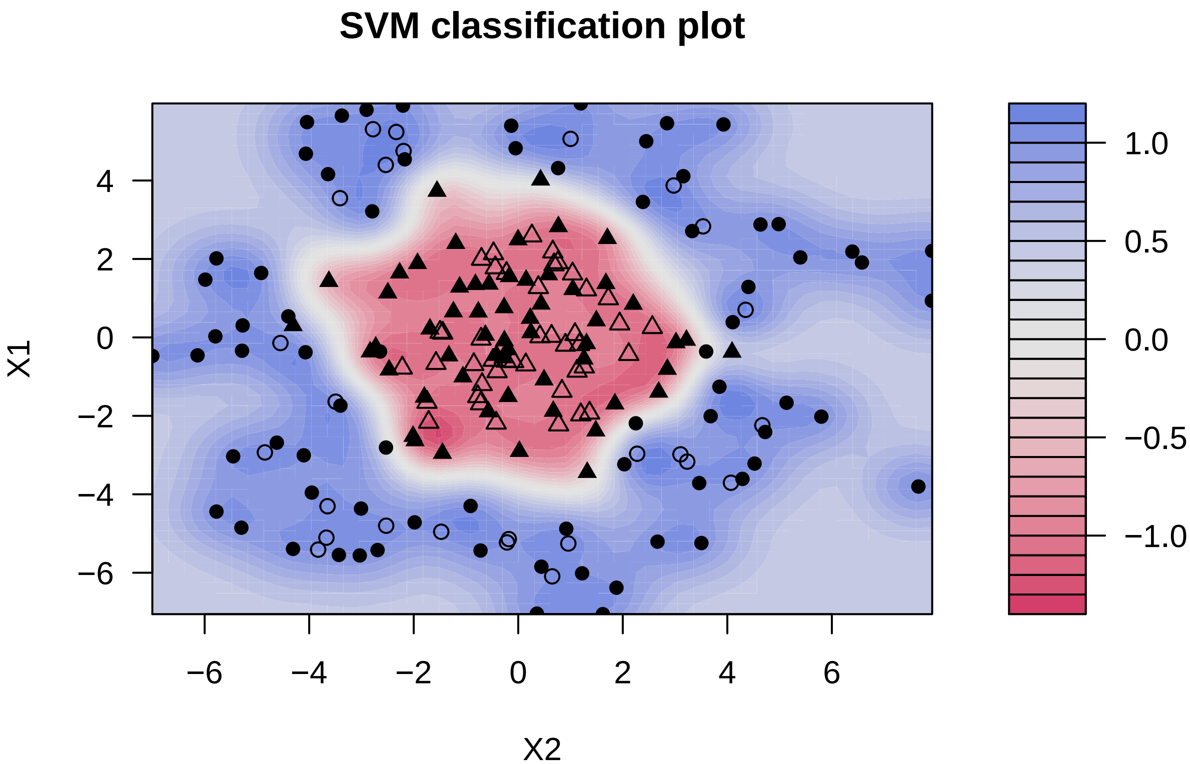

SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
1.0
0.5
0.0
−0.5
−1.0

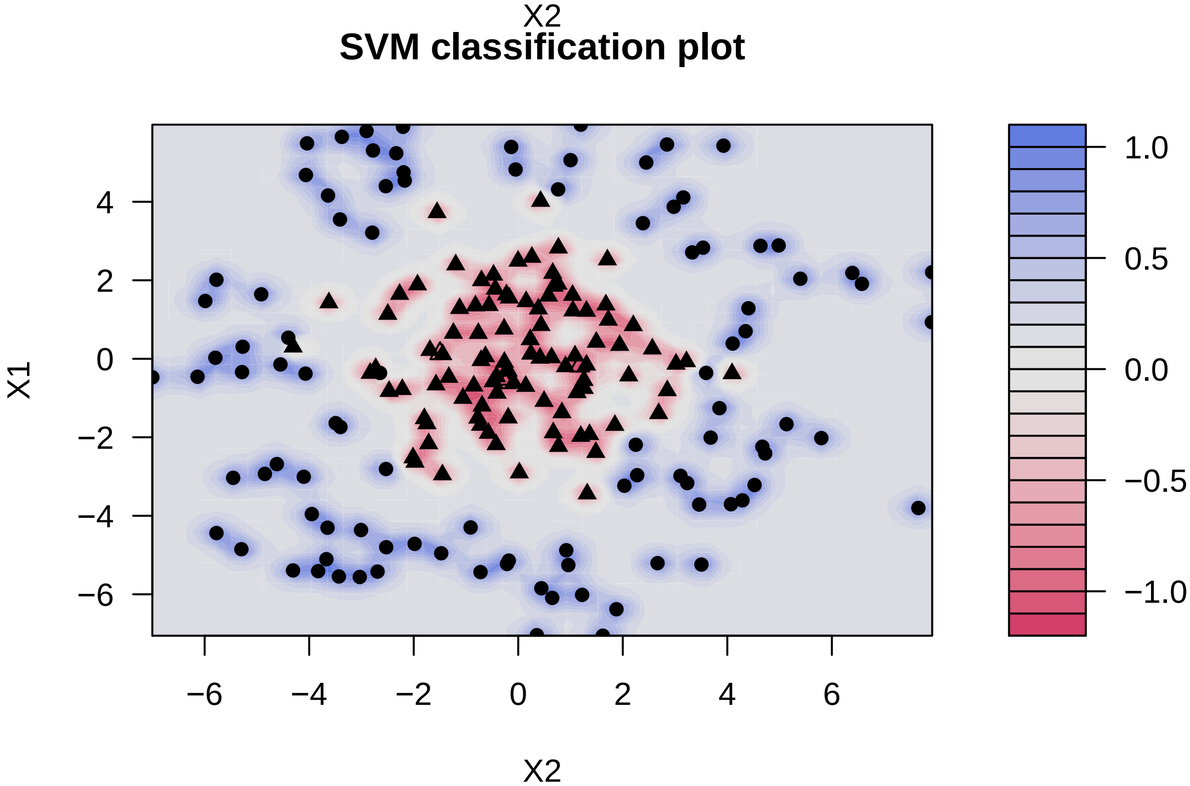

SVM classification plot
X1
X2
6
4
2
0
−2
−2
0
2
4
6
0.5
0.0
−0.5

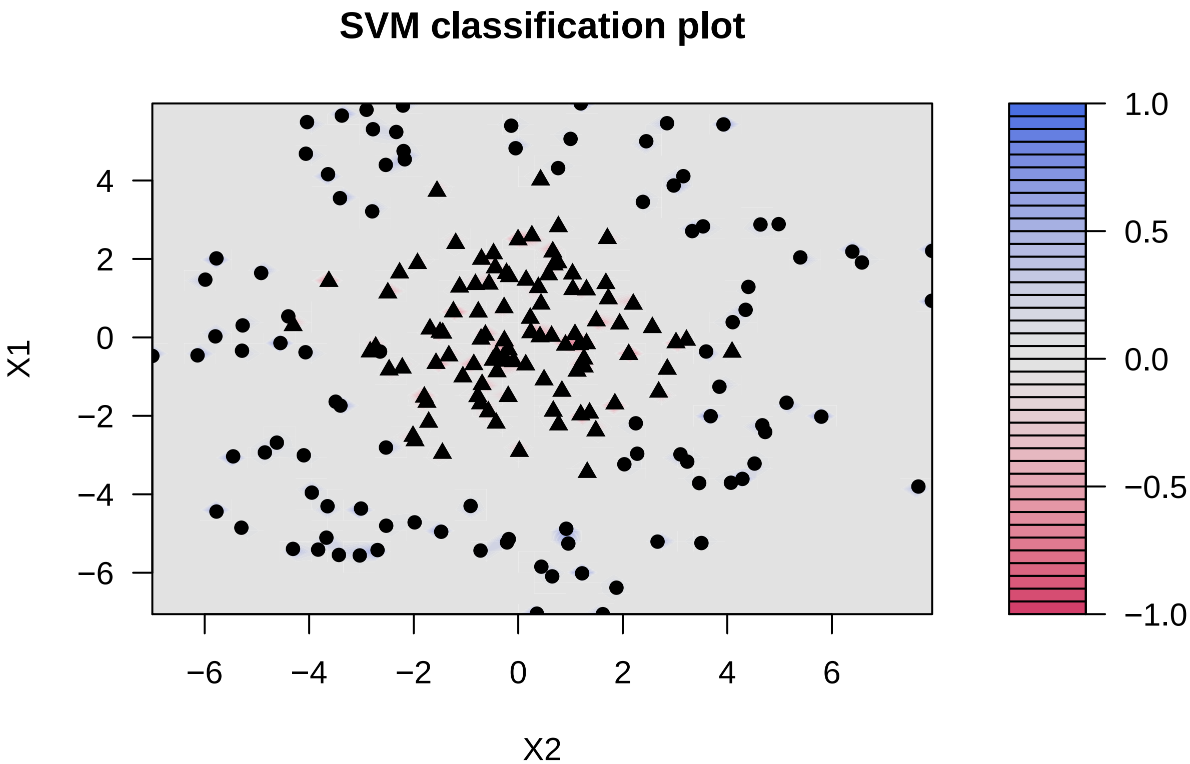

SVM classification plot
X1
X2
4
2
0
−2
−4
−6
−6
−4
−2
0
2
4
6
1.0
0.5
0.0
−0.5
−1.0

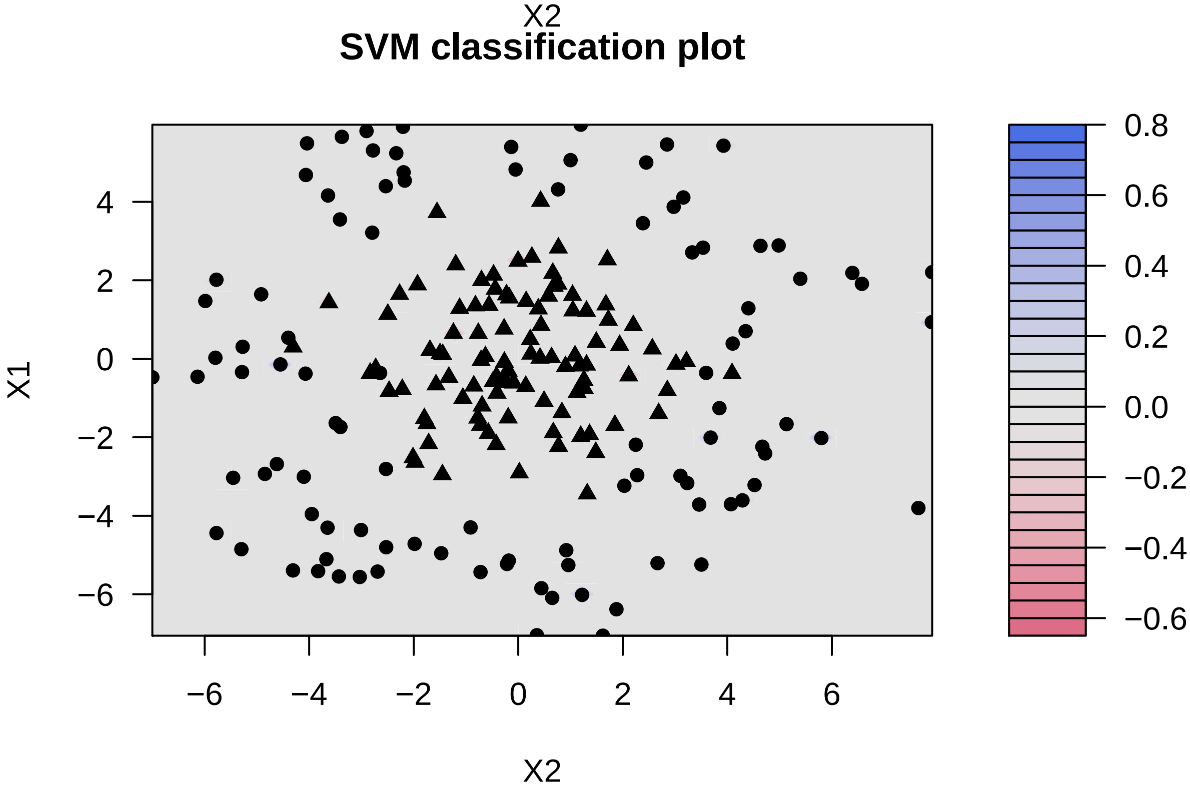

SVM classification plot
X1
X2
4
2
0
−2
−4
−6
6
3
1
−1


SVM classification plot
X1
X2
4
2
0
−2
−4
−6
6
1
−1


SVM classification plot
X1
X2
4
2
0
−2
−4
−6
6
1.0
0.5
0.0
−0.5
−1.0


SVM classification plot
X1
X2
4
2
0
−2
−4
−6
6
1.0
0.5
0.0
−0.5
−1.0


SVM classification plot
X1
X2
1.0
0.5
0.0
−0.5
−1.0


SVM classification plot
X1
X2
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4
−0.6

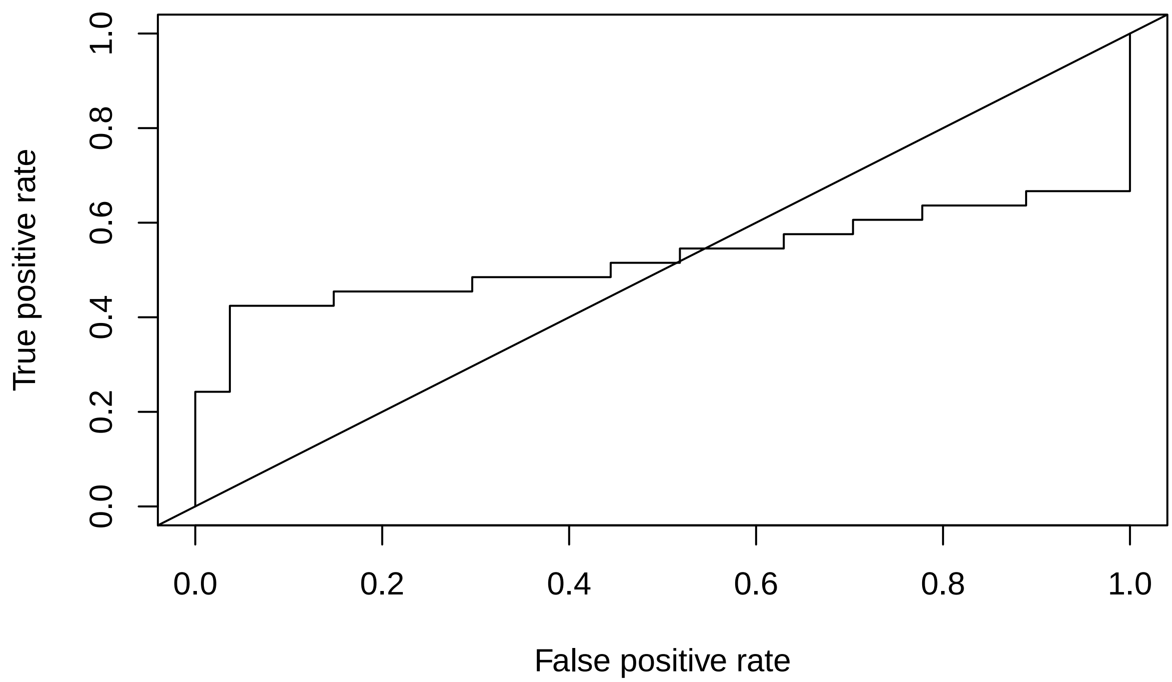
ROC curve

```
train_index <- sample(1:nrow(df2), 0.7 * nrow(df2))
train <- df2[train_index, ]
test <- df2[-train_index, ]
```

```
lda.mod <- MASS::lda(y~X1+X2, data=train)
lda.pred <- predict(lda.mod, newdata = test)
lda.post <- lda.pred$posterior

lda.prediction <- prediction(lda.post[,1], test$y)
roc <- performance(lda.prediction, measure = "tpr", x.measure = "fpr")
```

```
plot(roc)
abline(0, 1)
```




```
auc.perf <- performance(lda.prediction, measure = "auc")
auc.perf@y.values
```

```
## [[1]]
## [1] 0.5263749
```

Because the area under the curve < 0.5, it is worse than a random classifier.