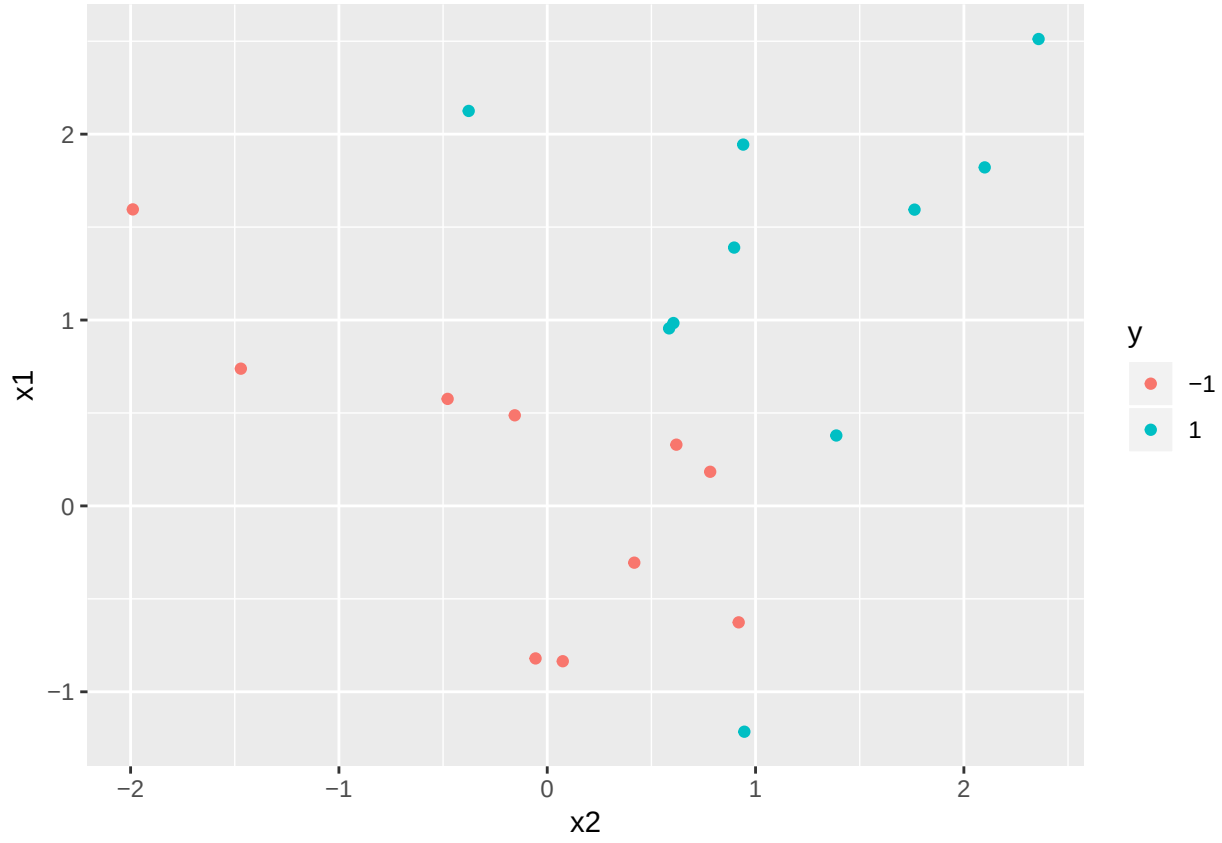
Support Vector Machine

```
library(tidyverse)
```

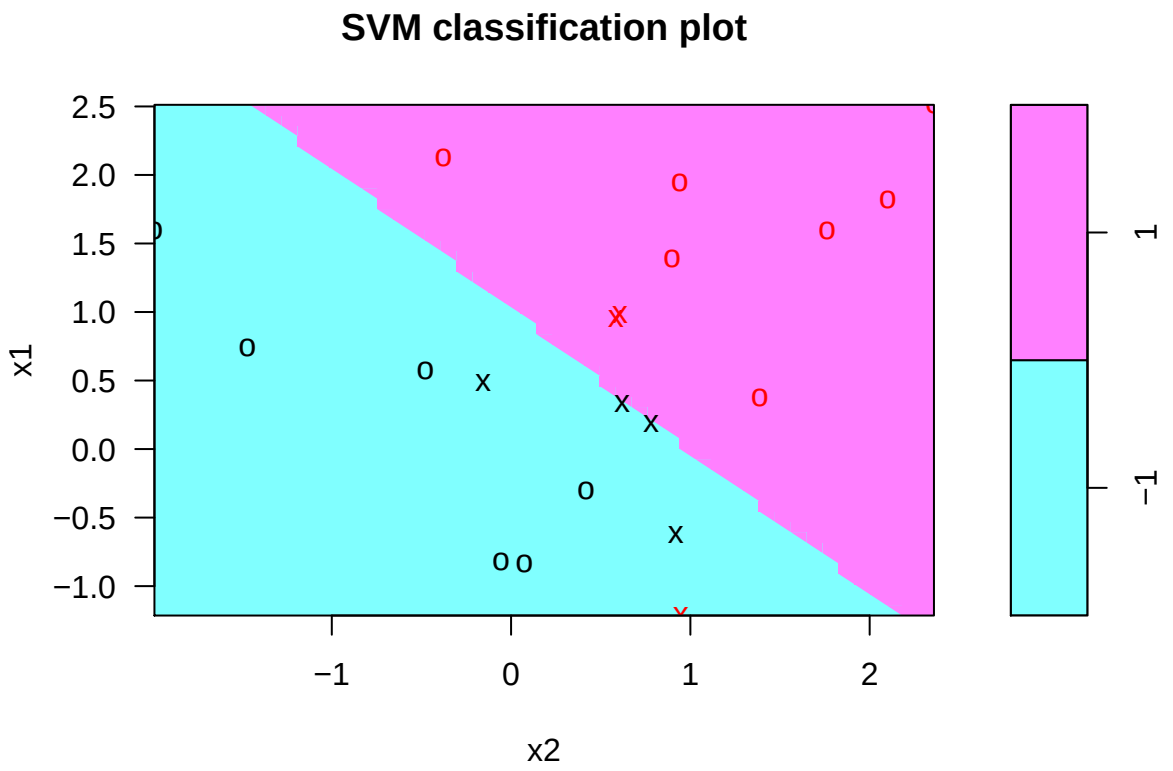
```
## -- Attaching packages ----------------------- tidyverse 1.2.1 --

## v ggplot2 3.1.0       v purrr   0.3.2
## v tibble  2.1.1       v dplyr   0.8.0.1
## v tidyr   0.8.3       v stringr 1.4.0
## v readr   1.3.1       v forcats 0.4.0

## -- Conflicts -------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
set.seed(1)
x <- matrix(rnorm(20 * 2), ncol = 2)
y <- c(rep(-1, 10), rep(1, 10))
x[y == 1, ] <- x[y == 1, ] + 1
dat <- tibble(x1 = x[,1], x2 = x[,2], y = as.factor(y))
```

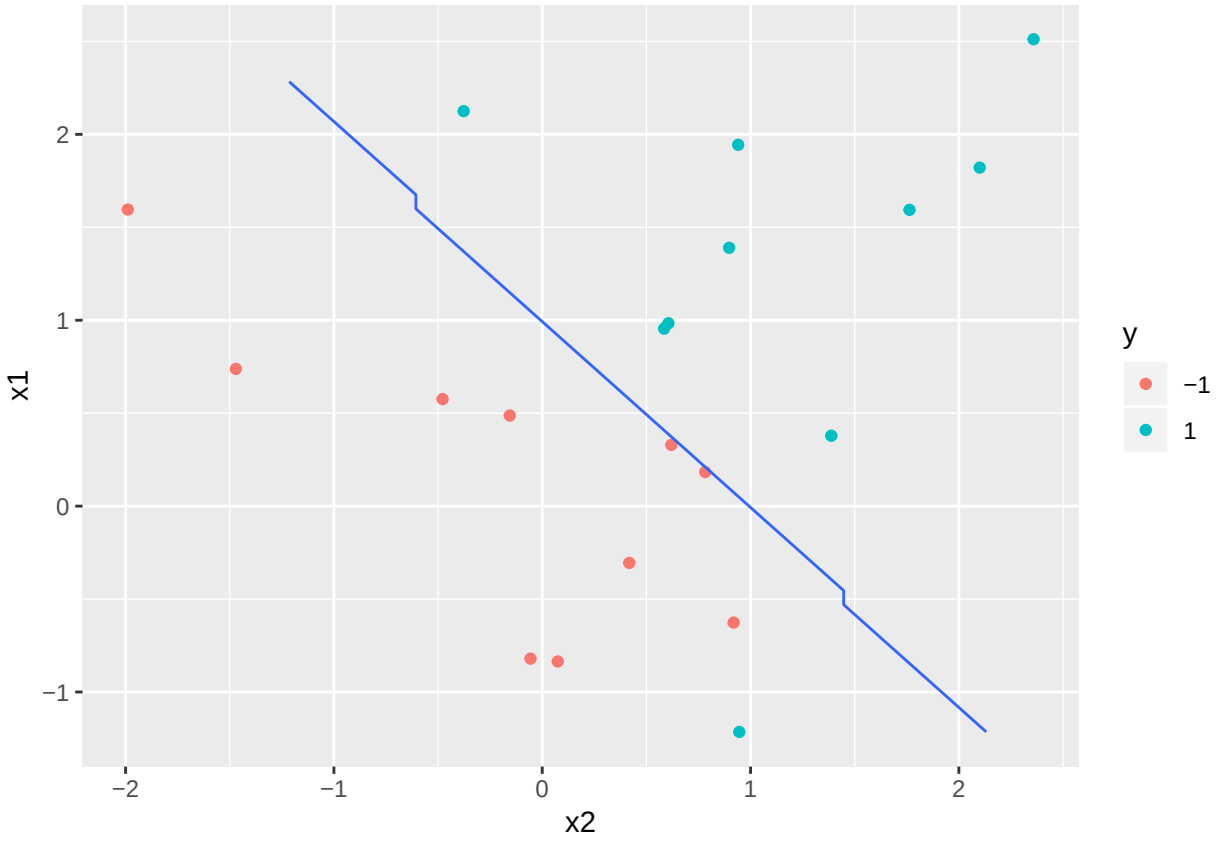
```
ggplot(dat) + geom_point(aes(x2, x1, color = y))
```



```
library(e1071)
svmfit <- svm(y ~ ., data = dat, kernel = "linear", cost = 10, scale = FALSE)
plot(svmfit, dat)
```



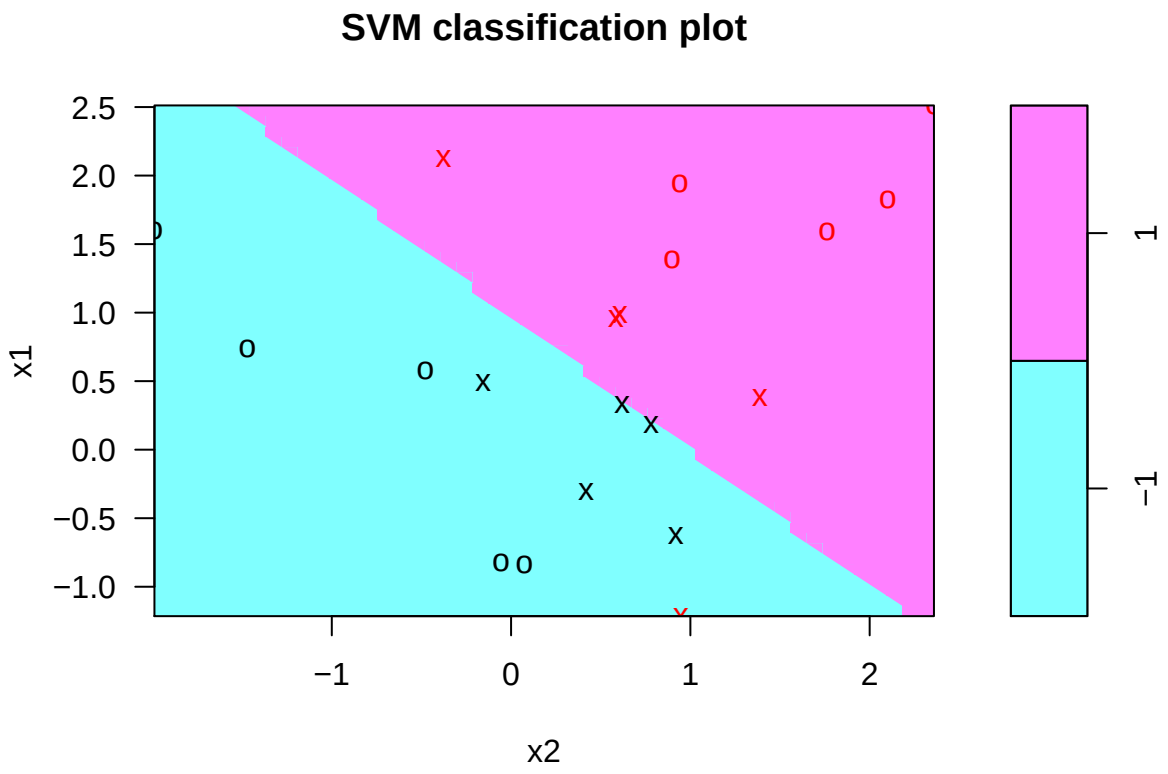

```
grid <- crossing(x1 = modelr::seq_range(dat$x1, 50), x2 = modelr::seq_range(dat$x1, 50))  %>%
    modelr::add_predictions(svmfit)
ggplot(dat) + geom_point(aes(x2, x1, color = y)) + geom_contour(data = grid, aes(x2, x1, z = as.numeric
```




```
svmfit$index  # support vectors
```

```
## [1]  1  2  5  7 14 16 17
```

```
svmfit <- svm(y ~ ., data = dat, kernel = "linear", cost = 1, scale = FALSE)
plot(svmfit, dat)
```




```
svmfit$index
```

```
##  [1]  1  2  5  7 10 13 14 15 16 17
```

```
set.seed(1)
tune_out <- tune(svm, y ~ ., data = dat, kernel = "linear",
                 ranges = list(cost = c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
summary(tune_out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##   0.1
##
## - best performance: 0.1
##
## - Detailed performance results:
##    cost error dispersion
## 1 1e-03  0.70  0.4216370
## 2 1e-02  0.70  0.4216370
## 3 1e-01  0.10  0.2108185
```

```
## 4 1e+00  0.15  0.2415229
## 5 5e+00  0.15  0.2415229
## 6 1e+01  0.15  0.2415229
## 7 1e+02  0.15  0.2415229
```

```
bestmod <- tune_out$best.model
summary(bestmod)
```

```
##
## Call:
## best.tune(method = svm, train.x = y ~ ., data = dat, ranges = list(cost = c(0.001,
##     0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.1
##       gamma:  0.5
##
## Number of Support Vectors:  16
##
##  ( 8 8 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```
xtest <- matrix(rnorm(20 * 2), ncol = 2)
ytest <- sample(c(-1, 1), 20, rep = TRUE)
xtest[ytest == 1, ] <- xtest[ytest == 1, ] + 1
testdat <- tibble(x1 = xtest[, 1], x2 = xtest[, 2], y = as.factor(ytest))
```

```
testdat %>% modelr::add_predictions(bestmod) %>% count(y, pred) %>% spread(pred, "n")
```

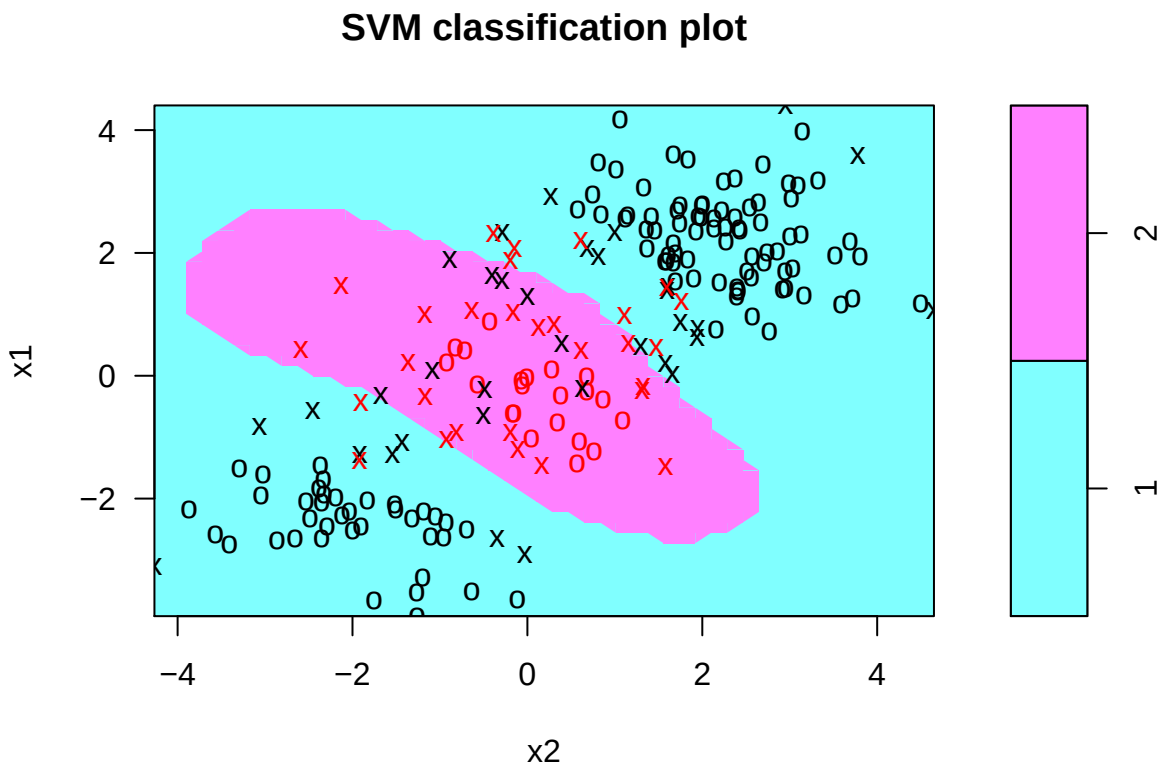
```
## # A tibble: 2 x 3
##   y      `-1`   `1`
##   <fct> <int> <int>
## 1 -1       11    NA
## 2 1         1     8
```

Other kernels

```
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 <- tibble(x1 = x[,1], x2 = x[,2], y = as.factor(y))
```

```
ggplot(dat) + geom_point(aes(x2, x1, color = y))
```

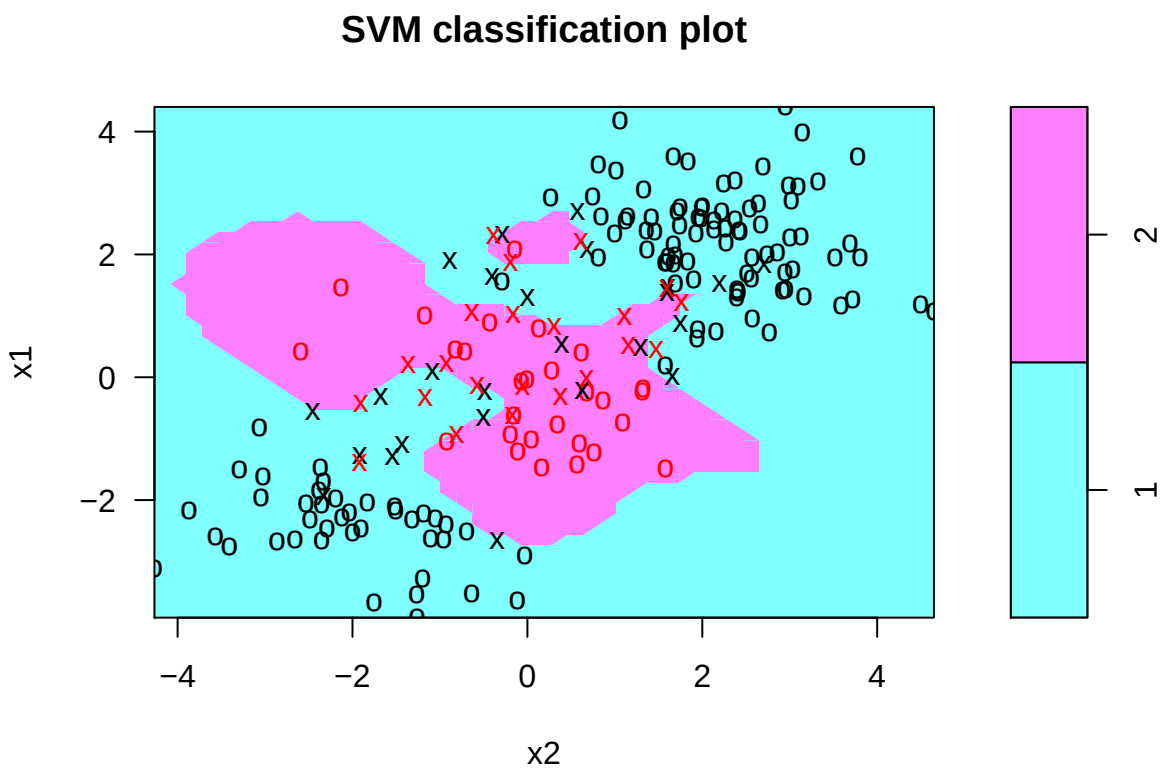

```
svmfit <- svm(y ~ ., data = dat, kernel = "radial", gamma = 1, cost = 1)
plot(svmfit, dat)
```



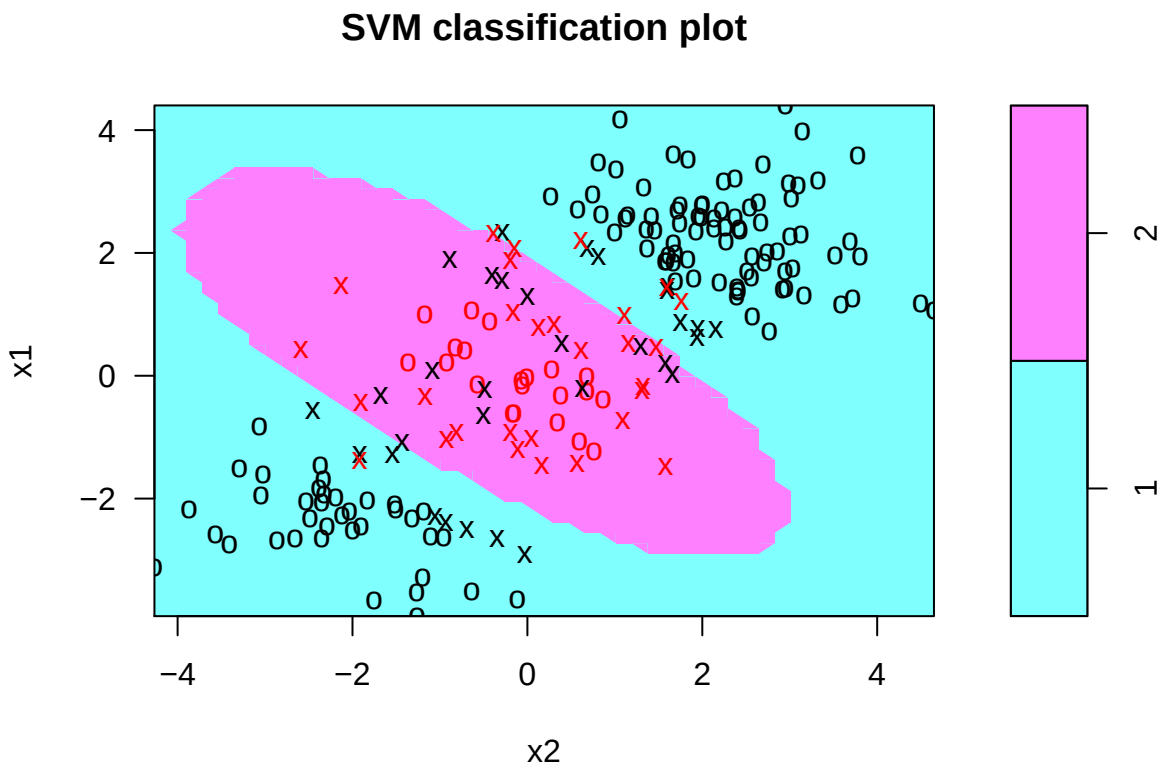

```
summary(svmfit)
```

```
##
## Call:
## svm(formula = y ~ ., data = dat, kernel = "radial", gamma = 1,
##     cost = 1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  1
##
## Number of Support Vectors:  63
##
##  ( 34 29 )
##
##
## Number of Classes:  2
##
## Levels:
##  1 2
```

```
svmfit <- svm(y ~ ., data = dat, kernel = "radial", gamma = 1, cost = 1e5)
plot(svmfit, dat)
```



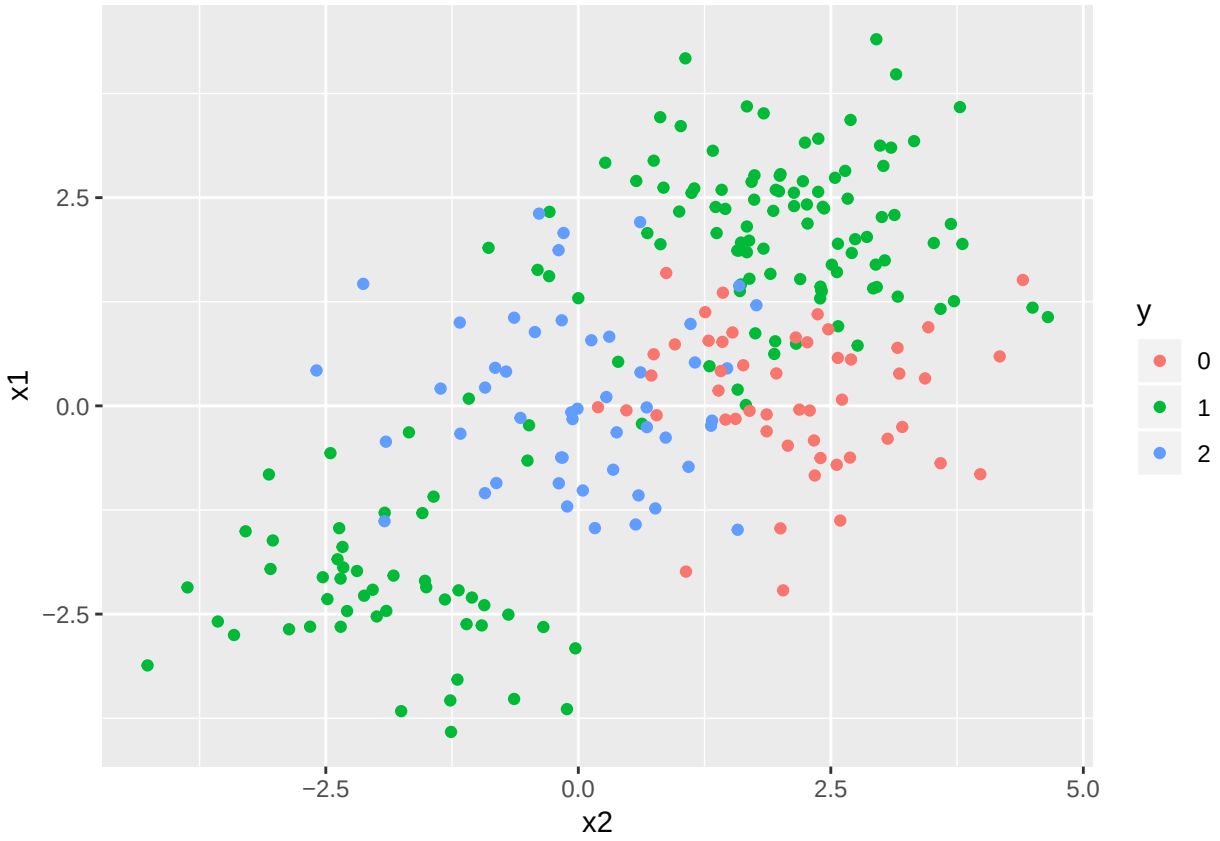

```
set.seed(1)
tune_out <- tune(svm, y ~ ., data = dat, kernel = "radial",
                 ranges = list(cost = c(0.1, 1, 10, 100, 1000), gamma = c(0.1, 0.5, 1, 2, 3, 4)))
# summary(tune_out)
plot(tune_out$best.model, dat)
```



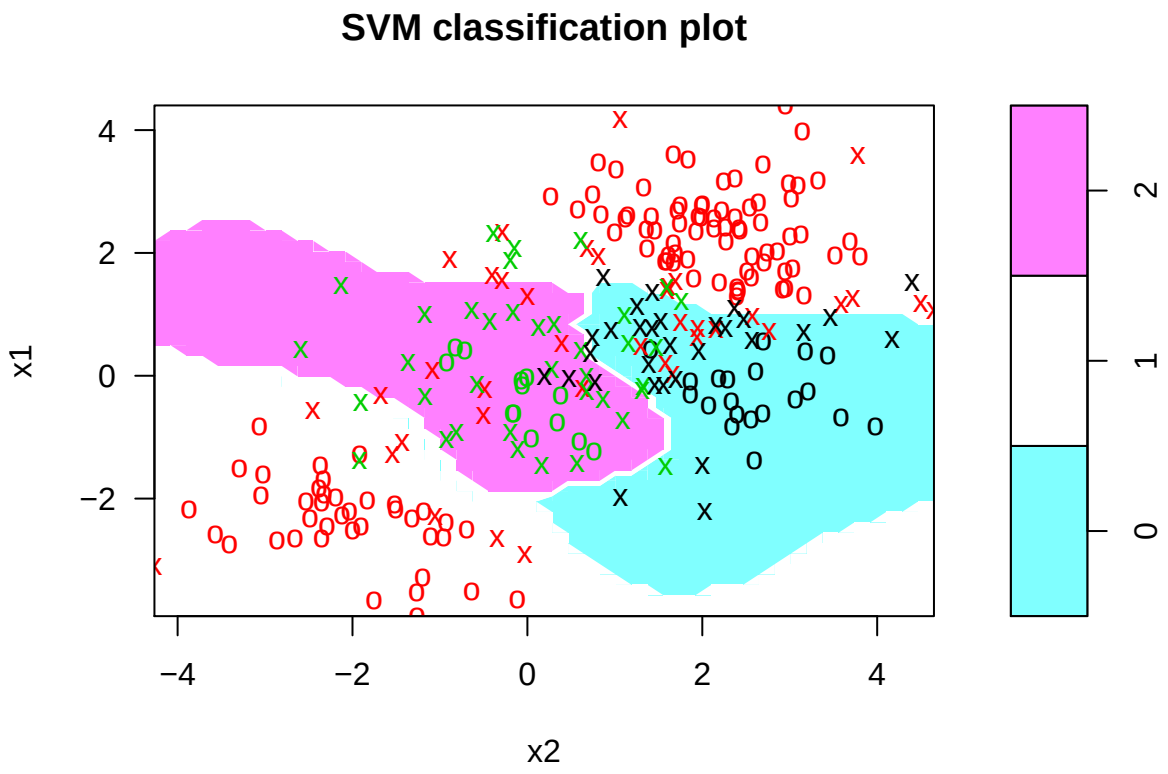
###SVM with multiple classes

```
set.seed(1)
x <- rbind(x, matrix(rnorm(50 * 2), ncol = 2))
y <- c(y, rep(0, 50))
x[y == 0, 2] <- x[y == 0, 2] + 2
dat <- tibble(x1 = x[,1], x2 = x[,2], y = as.factor(y))
```

```
ggplot(dat) + geom_point(aes(x2, x1, color = y))
```




```
svmfit <- svm(y ~ ., data = dat, kernel = "radial", cost = 10, gamma = 1)
plot(svmfit, dat)
```


SVM classification plot
x1
x2
4
2
0
−2
−4
−2
0
2
4
2
1
0