SLP answers exercises session 5

Exercise: Splines

```r
library("foreign")
NESDA <- read.spss("nesda.sav", use.value.labels = FALSE, to.data.frame = TRUE)
summary(NESDA)
```

```
##      pident              Sexe            Age             aedu      
##  Min.   :100013   Min.   :1.000   Min.   :18.00   Min.   : 5.00  
##  1st Qu.:110833   1st Qu.:1.000   1st Qu.:30.00   1st Qu.:10.00  
##  Median :210300   Median :2.000   Median :43.00   Median :12.00  
##  Mean   :207229   Mean   :1.685   Mean   :41.59   Mean   :12.25  
##  3rd Qu.:310132   3rd Qu.:2.000   3rd Qu.:53.00   3rd Qu.:15.00  
##  Max.   :330392   Max.   :2.000   Max.   :64.00   Max.   :18.00  
##     aframe02         neurot         extrave          openes     
##  Min.   :1.000   Min.   :12.00   Min.   :18.00   Min.   :12.00  
##  1st Qu.:1.000   1st Qu.:29.00   1st Qu.:32.00   1st Qu.:27.00  
##  Median :1.000   Median :37.00   Median :37.00   Median :31.00  
##  Mean   :1.652   Mean   :35.97   Mean   :37.22   Mean   :31.23  
##  3rd Qu.:2.000   3rd Qu.:43.00   3rd Qu.:43.00   3rd Qu.:35.00  
##  Max.   :3.000   Max.   :57.00   Max.   :54.00   Max.   :47.00  
##     agreeab         aconscie          dyst              mdd      
##  Min.   :24.00   Min.   :17.00   Min.   :0.00000   Min.   :0.00  
##  1st Qu.:41.00   1st Qu.:34.00   1st Qu.:0.00000   1st Qu.:0.00  
##  Median :44.00   Median :38.00   Median :0.00000   Median :0.00  
##  Mean   :44.01   Mean   :37.53   Mean   :0.09333   Mean   :0.34  
##  3rd Qu.:48.00   3rd Qu.:42.00   3rd Qu.:0.00000   3rd Qu.:1.00  
##  Max.   :58.00   Max.   :55.00   Max.   :1.00000   Max.   :1.00  
##       gad              sp               pd        
##  Min.   :0.000   Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.000   1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :0.000   Median :0.0000   Median :0.0000  
##  Mean   :0.135   Mean   :0.2233   Mean   :0.2933  
##  3rd Qu.:0.000   3rd Qu.:0.0000   3rd Qu.:1.0000  
##  Max.   :1.000   Max.   :1.0000   Max.   :1.0000  
```

```r
NESDA$mdd <- factor(NESDA$mdd)
train <- NESDA[1:400,]
test <- NESDA[401:600,]
```

```r
library("mgcv")
```

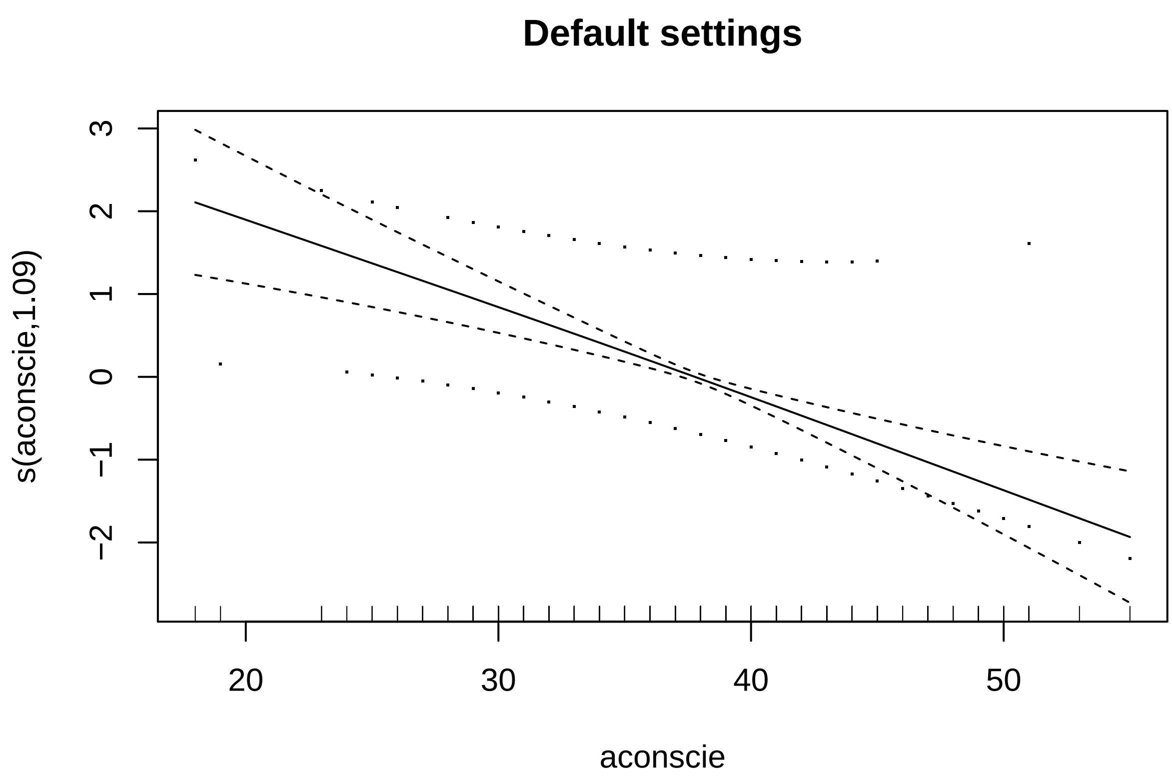
```
## Loading required package: nlme
```

```
## This is mgcv 1.8-31. For overview type 'help("mgcv-package")'.
```

```r
gam1 <- gam(mdd ~ s(aconscie), data = train,  family = "binomial",
            method = "REML")
summary(gam1)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## mdd ~ s(aconscie)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.7745     0.1133  -6.836 8.15e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq  p-value
## s(aconscie) 1.085  1.166  29.03 1.54e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0775   Deviance explained = 6.64%
## -REML = 239.39  Scale est. = 1         n = 400
```

```r
plot(gam1, residuals = TRUE, main = "Default settings")
```

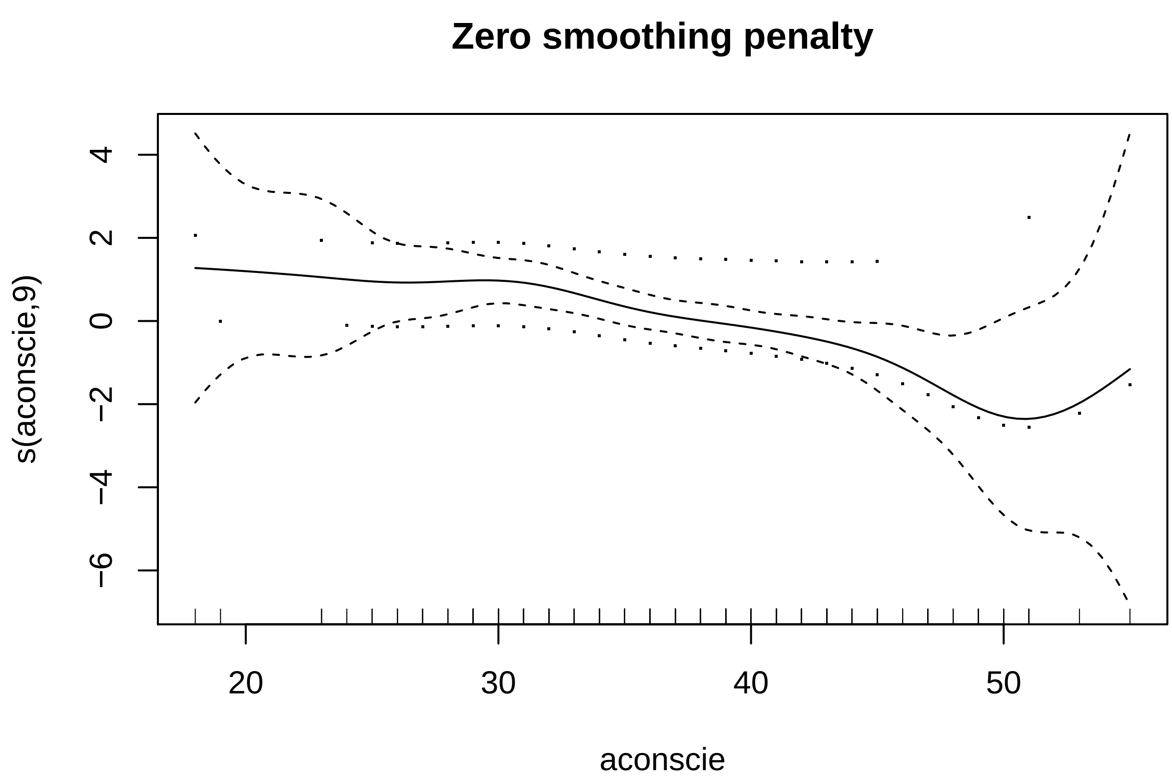


We get a curve that indicates a mostly linear effect. We also see that the edf is close to 1. The statistical test indicates the effect of conscientiousness on mdd is significant.

```r
gam2 <- gam(mdd ~ s(aconscie, sp = 0), data = train,  family = "binomial", method = "REML")
summary(gam2)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## mdd ~ s(aconscie, sp = 0)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.8047     0.1198  -6.718 1.85e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df Chi.sq  p-value
## s(aconscie)   9      9  29.34 0.000569 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0635   Deviance explained =  7.1%
## -REML =  375.2  Scale est. = 1         n = 400
```

```r
plot(gam2, residuals = TRUE, main = "Zero smoothing penalty")
```

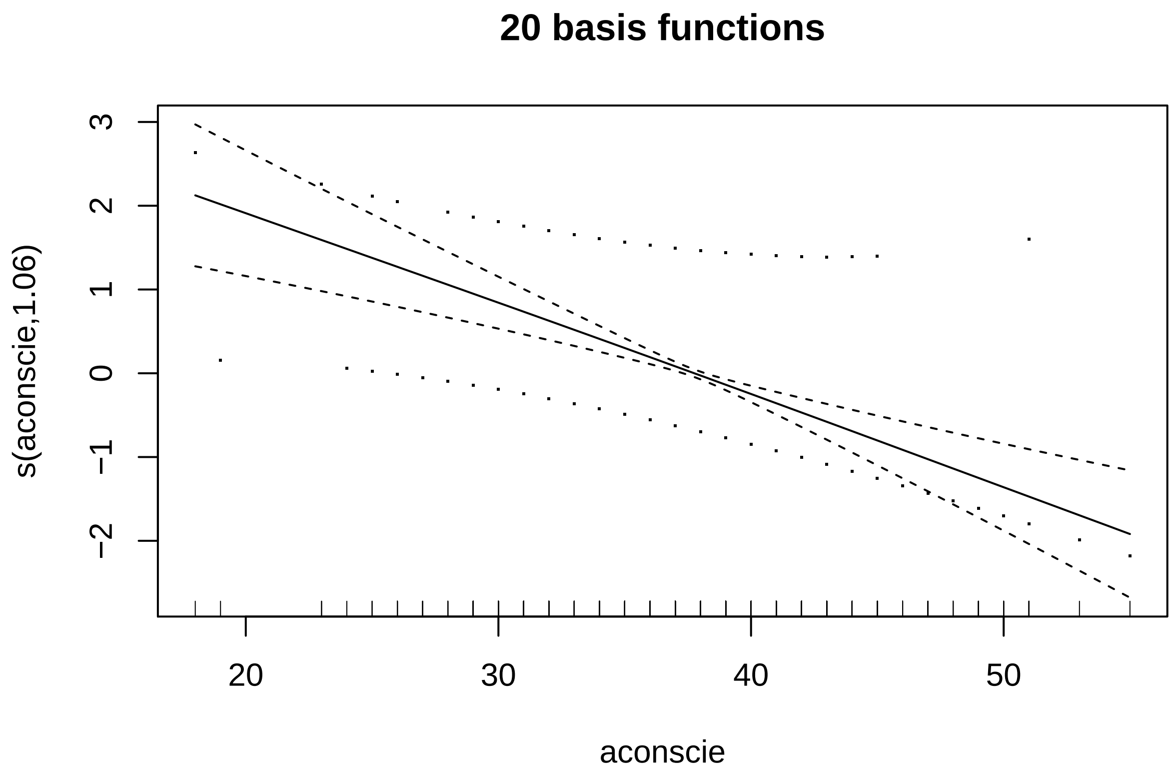


If we set the smoothing penalty to zero, we get a very wiggly line. By default, 9 basis functions are fitted, and we see from the edf that none of these functions is penalized out of the fitted function.

```
gam3 <- gam(mdd ~ s(aconscie, k = 20), data = train,  family = "binomial")
summary(gam3)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## mdd ~ s(aconscie, k = 20)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.7740     0.1132  -6.835 8.21e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq  p-value
## s(aconscie) 1.056   1.11  29.26 1.08e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.0774   Deviance explained = 6.63%
## UBRE = 0.19454  Scale est. = 1         n = 400
```

```
plot(gam3, residuals = TRUE, main = "20 basis functions")
```



Increasing the number of basis functions does not change our results. The smoothing penalty reduces the complexity of the fitted function back to something similar as we got with less basis functions.

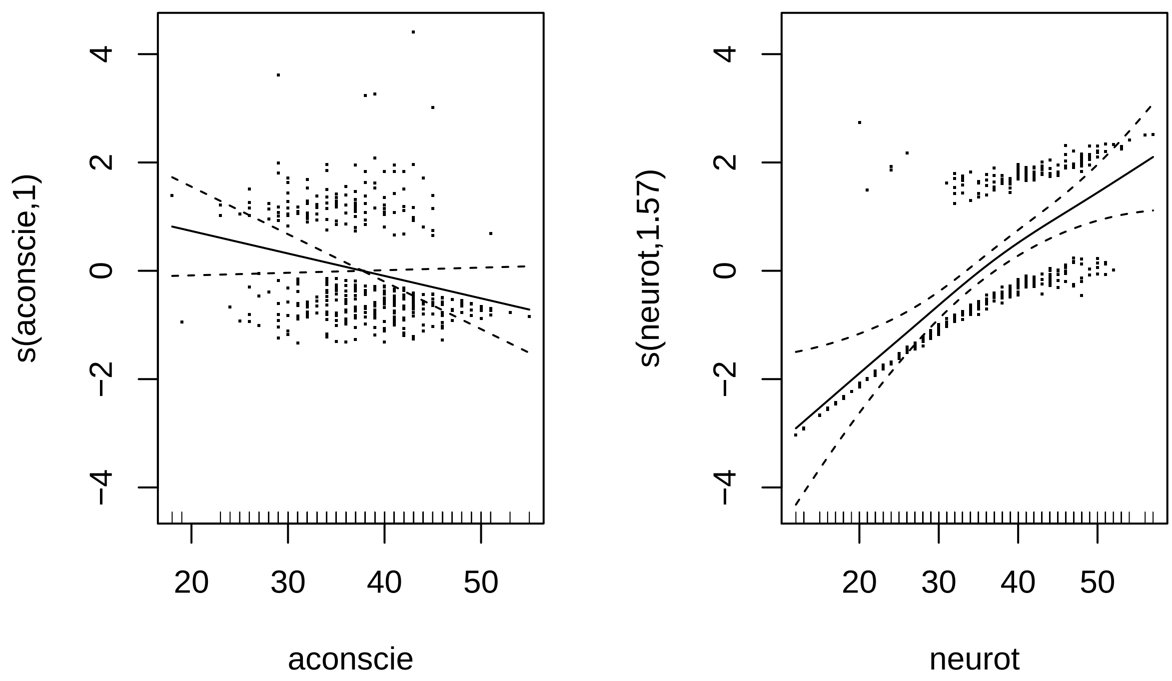
In sum: Knot location does not matter much when you specify more knots than you probably need, the smoothness penalty will penalize them down.

```
gam4 <- gam(mdd ~ s(aconscie) +  s(neurot), data = train,  family = "binomial")
summary(gam4)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## mdd ~ s(aconscie) + s(neurot)
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -0.9541     0.1354  -7.044 1.87e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##               edf Ref.df Chi.sq  p-value
## s(aconscie) 1.000  1.000  3.214    0.073 .
## s(neurot)   1.569  1.962 38.581 4.51e-09 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.177   Deviance explained = 16.4%
## UBRE = 0.078594  Scale est. = 1         n = 400
```

```r
par(mfrow = c(1, 2))
plot(gam4, residuals = TRUE)
```



When we add neuroticism as a predictor, the effect of conscientiousnous is reduced to a linear effect, while neuroticism has a slightly non-linear effect.

```r
gam.preds <- predict(gam4, newdata = test, type = "response")
head(gam.preds)
```

```
##       401       402       403       404       405       406
## 0.6172342 0.5874856 0.3559893 0.1404975 0.7028163 0.5034647
```

```r
gam.tab <- table(test$mdd, gam.preds > .5)
gam.tab
```

```
##
##     FALSE TRUE
##   0   111   17
##   1    43   29
```

```r
1 - sum(diag(prop.table(gam.tab)))
```

```
## [1] 0.3
```

Note: REML estimation

By default, the `gam()` function will use the default generalized cross-validation criterion for smoothness selection. For finite sample sizes, the (default) GCV citerion may overfit (undersmooth) and REML is often preferred.

REML and GCV try to do the same thing: make the smooths in your model just wiggly enough and no wigglier. GCV will select optimal smoothing parameters when the sample size is infinite.

There is some finite sample size at which this asymptotic result kicks in, but we rarely have data of that size. At smaller sample sizes GCV can develop multiple minima, making optimisation difficult, yielding more variable estimates of the smoothing parameter.

GCV also tends to undersmooth, as it penalizes overfit weakly.

REML penalizes overfitting more, yielding more prounounced optima, leading to fewer optimisation issues and less variable estimates of the smoothing parameter.

See also:

Reiss, P.T. and Ogden, R.T. (2009) Smoothing parameter selection for a class of semiparametric linear models. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 71, 505–523.

Wood, S.N. (2011) Fast stable restricted maximum likelihood and marginal likelihood estimation of semiparametric generalized linear models. Journal of the Royal Statistical Society: Series B (Statistical Methodology), 73, 3–36.

Exercise: Support vectors

Linear kernel

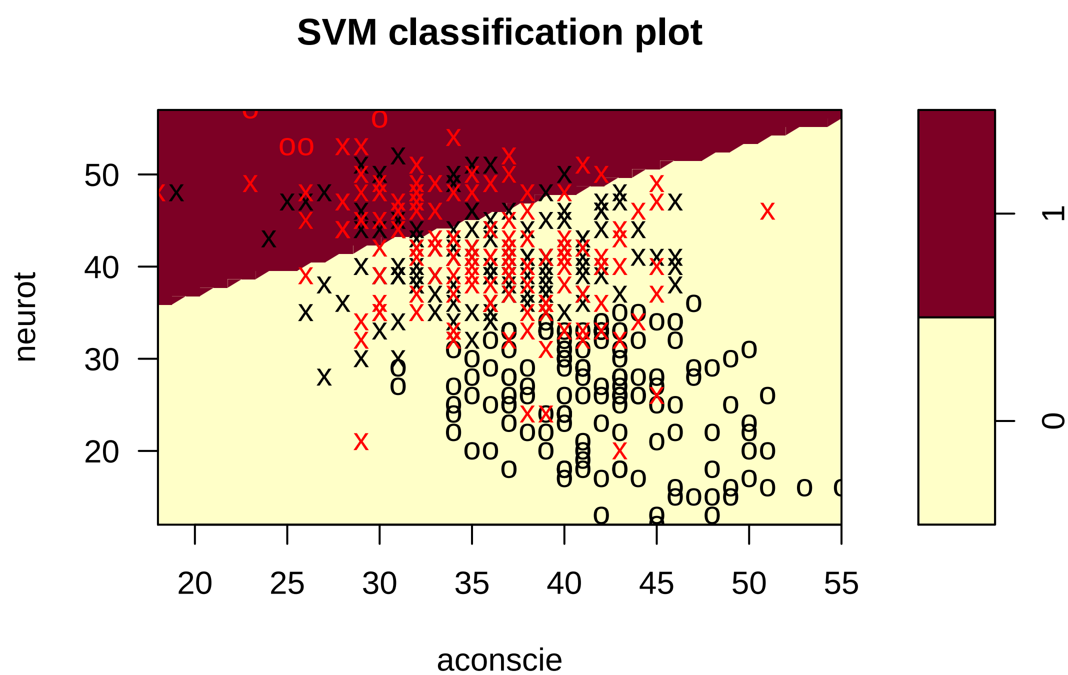
```
library("e1071")
tune.out <- tune(svm, mdd ~ neurot + aconscie, data = train,
                 kernel = "linear", scale = TRUE,
                 ranges = list(cost = c(.001, .01, .1, 1, 5, 10, 100)))
tune.out
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.31
```

```
tune.out$best.parameters
```

```
##   cost
## 4    1
```

```
svmfit <- svm(mdd ~ neurot + aconscie, data = train, kernel = "linear",
              cost = 10, scale = TRUE)
plot(svmfit, data = train, formula = neurot ~ aconscie)
```




```
svmfit
```

```
##
## Call:
## svm(formula = mdd ~ neurot + aconscie, data = train, kernel = "linear",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  10
##
## Number of Support Vectors:  257
```

```
svm.preds <- predict(svmfit, newdata = test)
head(svm.preds)
```

```
## 401 402 403 404 405 406
##   1   1   0   0   1   0
## Levels: 0 1
```

```
svm.tab <- table(test$mdd, svm.preds)
svm.tab
```

```
##    svm.preds
##       0   1
##   0 114  14
##   1  50  22
```

```
1 - sum(diag(prop.table(svm.tab)))
```

[1] 0.32

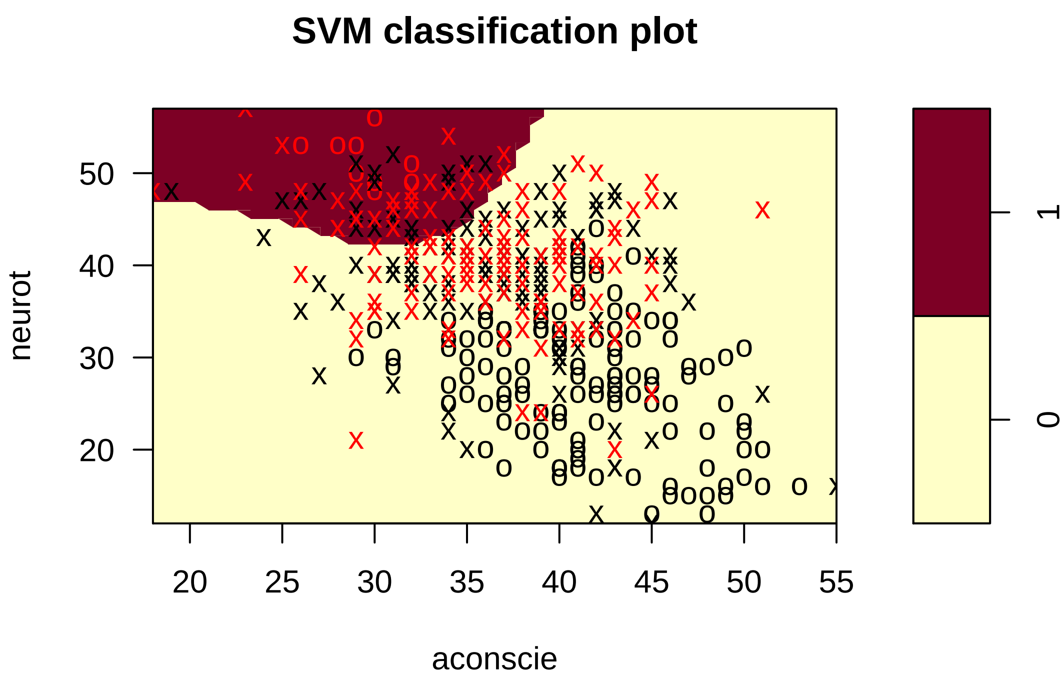
In the plot, `x` is a support vector, `o` are for other data points. The colors represent the classes the observations belong to (black for the first class (non depressed), red for the second class (depressed)).

Radial basis kernel

```
tune.out <- tune(svm, mdd ~ neurot + aconscie, data = train,
                 kernel = "radial", ranges = list(
                   cost = c(.1, 1, 10, 100, 1000),
                   gamma = c(.5, 1, 2, 3, 4)))
tune.out
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1   0.5
##
## - best performance: 0.305
```

```
svfit <- svm(mdd ~ neurot + aconscie, data = train,
             kernel = "radial", gamma = tune.out$best.parameters$gamma,
             cost = tune.out$best.parameters$cost)
plot(svmfit, train, formula = neurot ~ aconscie)
```



The misclassification rate in the test data is:

```r
svm.preds <- predict(svmfit, newdata = test)
svm.tab <- table(test$mdd, svm.preds)
svm.tab
```

```
##    svm.preds
##       0   1
##   0 118  10
##   1  52  20
```

```r
1 - sum(diag(prop.table(svm.tab)))
```

```
## [1] 0.31
```