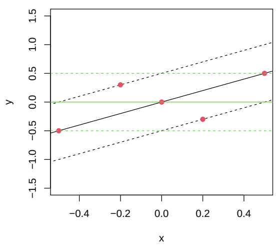
Solution 1: SVM – Regression

(a) Regarding the values of $d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), \mathbf{x}^{(i)}\right)$ for an outcome $y^{(i)}$ we have that:

- If $y^{(i)}$ is within the ϵ-tube around $f(\mathbf{x}^{(i)} \mid \theta_0, \boldsymbol{\theta})$, then $d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), \mathbf{x}^{(i)}\right) \geq 0$. The largest possible value of $d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), \mathbf{x}^{(i)}\right)$ is ϵ, which corresponds to a perfect prediction for that point, i.e., $y^{(i)} = f(\mathbf{x}^{(i)} \mid \theta_0, \boldsymbol{\theta})$.
- If $y^{(i)}$ is not within the ϵ-tube around $f(\mathbf{x}^{(i)} \mid \theta_0, \boldsymbol{\theta})$, then $d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), \mathbf{x}^{(i)}\right) < 0$.

A desirable choice of the parameters $(\theta_0, \boldsymbol{\theta})^\top$ with respect to γ_ϵ would be such that γ_ϵ is maximized, as this would make sure that the prediction errors are as far away as possible from the ϵ-boundaries, but still within the ϵ-tube.

The choice of the parameters $(\theta_0, \boldsymbol{\theta})^\top$ is not unique, as the plot on the right shows for $\epsilon = 0.5$. Both the black and the green model have $\gamma_\epsilon = \epsilon$, since $d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), -0.2\right) = \epsilon = d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), 0\right)$ and we cannot find another model such that its ϵ-tube covers the outcomes.



(b) We formulate the desired property of a maximal γ_ϵ as an optimization problem:

$$\begin{aligned} \max_{\boldsymbol{\theta}, \theta_0} \quad & \gamma_\epsilon \\ \text{s.t.} \quad & d_\epsilon\left(f(\cdot \mid \theta_0, \boldsymbol{\theta}), \mathbf{x}^{(i)}\right) \geq 0 \quad \forall\, i \in \{1, \ldots, n\}. \end{aligned}$$

The constraints mean that we require that any instance i should have a positive ϵ-distance of the prediction error for $f(\mathbf{x}^{(i)} \mid \theta_0, \boldsymbol{\theta})$. In other words, the differences between the predictions and the outcomes should be at most ϵ and within the ϵ-tube of the predictions. The latter optimization problem can be rewritten as

$$\begin{aligned} \max_{\boldsymbol{\theta}, \theta_0} \quad & \gamma_\epsilon \\ \text{s.t.} \quad & \epsilon - |y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \theta_0| \geq 0 \quad \forall\, i \in \{1, \ldots, n\}. \end{aligned}$$

And further to

$$\begin{aligned} \max_{\boldsymbol{\theta}, \theta_0} \quad & \gamma_\epsilon \\ \text{s.t.} \quad & \epsilon - y^{(i)} + \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0 \geq 0 \quad \forall\, i \in \{1, \ldots, n\} \\ \text{and} \quad & \epsilon + y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \theta_0 \geq 0 \quad \forall\, i \in \{1, \ldots, n\}. \end{aligned}$$

As we have seen before the solution might not be unique, so that we make the reference choice $\gamma_\epsilon = C/\|\boldsymbol{\theta}\|$ for some constant $C > 0$, leading to

$$\begin{aligned} \min_{\boldsymbol{\theta}, \theta_0} \quad & C\|\boldsymbol{\theta}\|^2 \\ \text{s.t.} \quad & \epsilon - y^{(i)} + \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0 \geq 0 \quad \forall\, i \in \{1, \ldots, n\} \\ \text{and} \quad & \epsilon + y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \theta_0 \geq 0 \quad \forall\, i \in \{1, \ldots, n\}. \end{aligned}$$

For sake of convenience, we set the constant to $\frac{1}{2}$.

(c) The Lagrange function of the SVM optimization problem is

$$
\begin{aligned}
L(\boldsymbol{\theta}, \theta_0, \boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}}) = \quad & \frac{1}{2}\|\boldsymbol{\theta}\|^2 - \sum_{i=1}^{n} \alpha_i \left[\epsilon - y^{(i)} + \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0\right)\right] - \sum_{i=1}^{n} \tilde{\alpha}_i \left[\epsilon - \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0\right) + y^{(i)}\right] \\
\text{s.t.} \quad & \alpha_i, \tilde{\alpha}_i \geq 0 \quad \forall i \in \{1, \dots, n\}.
\end{aligned}
$$

The **dual** form of this problem is

$$
\max_{\boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}}} \min_{\boldsymbol{\theta}, \theta_0} L(\boldsymbol{\theta}, \theta_0, \boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}}).
$$

(d) The stationary points of L can be derived by setting the derivative of the Lagrangian function to 0 and solve with respect to the corresponding term of interest, i.e., for $\boldsymbol{\theta}$:

$$
\nabla_{\boldsymbol{\theta}} L(\boldsymbol{\theta}, \theta_0, \boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}}) = \boldsymbol{\theta} - \sum_{i=1}^{n} \alpha_i \mathbf{x}^{(i)} + \sum_{i=1}^{n} \tilde{\alpha}_i \mathbf{x}^{(i)} \overset{!}{=} 0
$$

$$
\Leftrightarrow \boldsymbol{\theta} = \sum_{i=1}^{n} (\alpha_i - \tilde{\alpha}_i) \mathbf{x}^{(i)}.
$$

and for θ_0 :

$$
\nabla_{\theta_0} L(\boldsymbol{\theta}, \theta_0, \boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}}) = -\sum_{i=1}^{n} \alpha_i + \sum_{i=1}^{n} \tilde{\alpha}_i \overset{!}{=} 0
$$

$$
\Leftrightarrow 0 = \sum_{i=1}^{n} (\alpha_i - \tilde{\alpha}_i).
$$

If $(\boldsymbol{\theta}, \theta_0, \boldsymbol{\alpha}, \tilde{\boldsymbol{\alpha}})$ fulfills the KKT conditions (stationarity, primal/dual feasibility, complementary slackness), it solves both the primal and dual problem (strong duality). Under these conditions, and if we solve the dual problem and obtain $\hat{\boldsymbol{\alpha}}$ or $\widetilde{\hat{\boldsymbol{\alpha}}}$, we know that $\boldsymbol{\theta}$ is a linear combination of our data points:

$$
\hat{\theta} = \sum_{i=1}^{n} (\hat{\alpha}_i - \widetilde{\hat{\alpha}}_i) \mathbf{x}^{(i)}
$$

Complementary slackness means:

$$
\begin{aligned}
\hat{\alpha}_i \left[\epsilon - y^{(i)} + \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0\right)\right] &= 0 \quad \forall\, i \in \{1, ..., n\}, \\
\widetilde{\hat{\alpha}}_i \left[\epsilon - \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0\right) + y^{(i)}\right] &= 0 \quad \forall\, i \in \{1, ..., n\}.
\end{aligned}
$$

So either $\hat{\alpha}_i = 0$, or $\hat{\alpha}_i > 0$, then $\epsilon = y^{(i)} - \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0\right)$, and $(\mathbf{x}^{(i)}, y^{(i)})$ is exactly on the boundary of the ϵ-tube of the prediction and $\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0$ underestimates $y^{(i)}$ (by exactly ϵ). Similarly, it holds either $\widetilde{\hat{\alpha}}_i = 0$, or $\widetilde{\hat{\alpha}}_i > 0$, then $\epsilon = \left(\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0\right) - y^{(i)}$, and $(\mathbf{x}^{(i)}, y^{(i)})$ is exactly on the boundary of the ϵ-tube of the prediction and $\boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0$ overestimates $y^{(i)}$ (by exactly ϵ). For the bias term θ_0 we infer that

$$
\theta_0 = y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \epsilon
$$

in the case $\hat{\alpha}_i > 0$ and

$$
\theta_0 = y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \epsilon
$$

in the case $\widetilde{\hat{\alpha}}_i > 0$.

(e) The "softened" version of the optimization problem is obtained by introducing slack variables $\zeta^{(i)}, \widetilde{\zeta^{(i)}} \geq 0$ in the constraints:

$$
\begin{aligned}
\epsilon - y^{(i)} + \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0 &\geq \zeta^{(i)} \quad \forall i \in \{1, \dots, n\} \\
\text{and } \epsilon + y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \theta_0 &\geq \widetilde{\zeta^{(i)}} \quad \forall i \in \{1, \dots, n\}.
\end{aligned}
$$

We minimize then a weighted sum of $\|\boldsymbol{\theta}\|^2$ and the sum of the slack variables:

$$
\begin{aligned}
\min_{\boldsymbol{\theta},\theta_0} \quad & \frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{i=1}^{n} \zeta^{(i)} + \widetilde{\zeta^{(i)}} \\
\text{s.t.} \quad & \epsilon - y^{(i)} + \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0 \geq -\zeta^{(i)} \quad \forall i \in \{1,\dots,n\} \\
\text{and} \quad & \epsilon + y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \theta_0 \geq -\widetilde{\zeta^{(i)}} \quad \forall i \in \{1,\dots,n\}, \\
& \zeta^{(i)}, \widetilde{\zeta^{(i)}} \geq 0 \quad \forall i \in \{1,\dots,n\}.
\end{aligned}
$$

(f) In the optimum, the inequalities will hold with equality (as we minimize the slacks), so $\zeta^{(i)} = \epsilon - y^{(i)} + \boldsymbol{\theta}^\top \mathbf{x}^{(i)} + \theta_0$ and $\widetilde{\zeta^{(i)}} = \epsilon + y^{(i)} - \boldsymbol{\theta}^\top \mathbf{x}^{(i)} - \theta_0$, but the lowest value $\zeta^{(i)}$ and $\widetilde{\zeta^{(i)}}$ can take is 0. So we can rewrite the above:

$$
\frac{1}{2}\|\boldsymbol{\theta}\|^2 + C\sum_{i=1}^{n} L\left(y^{(i)}, f\left(\mathbf{x}^{(i)}\right)\right);\ L\left(y, f(\mathbf{x})\right) = \begin{cases} 0, & \text{if } |y - f(\mathbf{x})| \leq \epsilon, \\ |y - f(\mathbf{x})| - \epsilon, & \text{else.} \end{cases}
$$

This loss function is the ϵ-insensitive loss.

Solution 2: SVM – Optimization

- Implementation of the PEGASOS algorithm:

```r
#' @param y outcome vector
#' @param X design matrix (including a column of 1s for the intercept)
#' @param nr_iter number of iterations for the algorithm
#' @param theta starting values for thetas
#' @param lambda penalty parameter
#' @param alpha step size for weight decay
pegasos_linear <- function(
  y,
  X,
  nr_iter = 50000,
  theta = rnorm(ncol(X)),
  lambda = 1,
  alpha = 0.01)
{

  t <- 1
  n <- NROW(y)

  while(t <= nr_iter){

    f_current = X%*%theta
    i <- sample(1:n, 1)

    # update
    theta <- (1 - lambda * alpha) * theta
    # add second term if within margin
    if(y[i]*f_current[i] < 1) theta <- theta + alpha * y[i]*X[i,]

    t <- t + 1

  }

  return(theta)
```

```
}
```

- Check on a simple example

```r
## Check on a simple example
## --------------------------------------------

set.seed(2L)

C = 1

library(mlbench)
library(kernlab)

## Error in library(kernlab):  there is no package called 'kernlab'

data = mlbench.twonorm(n = 100, d = 2)


data = as.data.frame(data)
X = as.matrix(data[, 1:2])
y = data$classes
par(mar = c(5,4,4,6))
plot(x = data$x.1, y = data$x.2, pch = ifelse(data$classes == 1, "-", "+"), col = "black",
     xlab = "x1", ylab = "x2")

# recode y
y = ifelse(y == "2", 1, -1)
mod_pegasos = pegasos_linear(y, cbind(1,X), lambda = C/(NROW(y)))

# Add estimated decision boundary:
abline(a = - mod_pegasos[1] / mod_pegasos[2],
       b = - mod_pegasos[2] / mod_pegasos[3], col = "#D55E00")

# Compare to logistic regression:
mod_logreg = glm(classes ~ ., data = data, family = binomial())
abline(a = - coef(mod_logreg)[1] / coef(mod_logreg)[2],
       b = - coef(mod_logreg)[2] / coef(mod_logreg)[3], col =  "#56B4E9",
       lty = 3, lwd = 2)

# decision values
f_pegasos = cbind(1,X) %*% mod_pegasos

# How many wrong classified examples?
table(sign(f_pegasos * y))

##
## -1  1
##  5 95

## compare to kernlab. we CANNOT expect a PERFECT match
## ------------------------------------------------------------

mod_kernlab = ksvm(classes~.,
                   data = data,
                   kernel = "vanilladot",
                   C = C,
                   kpar = list(),
                   scaled = FALSE)
```

```
## Error in ksvm(classes ~ ., data = data, kernel = "vanilladot", C = C, :  could not find
function "ksvm"

f_kernlab = predict(mod_kernlab, newdata = data, type = "decision")

## Error in predict(mod_kernlab, newdata = data, type = "decision"):  object 'mod_kernlab'
not found

# How many wrong classified examples?
table(sign(f_kernlab * y))

## Error in table(sign(f_kernlab * y)):  object 'f_kernlab' not found

# compare outputs
print(range(abs(f_kernlab - f_pegasos)))

## Error in print(range(abs(f_kernlab - f_pegasos))):  object 'f_kernlab' not found

# compare coeffs
rbind(
  mod_pegasos,
  mod_kernlab = c(mod_kernlab@b,
  (params <- colSums(X[mod_kernlab@SVindex, ] *
                       mod_kernlab@alpha[[1]] *
                       y[mod_kernlab@SVindex])))
)

## Error in rbind(mod_pegasos, mod_kernlab = c(mod_kernlab@b, (params <-
colSums(X[mod_kernlab@SVindex, :  object 'mod_kernlab' not found

# seems we were reasonably close

# recompute margin
margin = 1 / sqrt(sum(params^2))

## Error in eval(expr, envir, enclos):  object 'params' not found

# compute value of intercept shift (the margin shift is in orthogonal direction
#   to the decision boundary, so this has to be transformed first)
m = - params[1] / params[2]

## Error in eval(expr, envir, enclos):  object 'params' not found

t_0 = margin * m / (cos(atan(1/m)))

## Error in eval(expr, envir, enclos):  object 'margin' not found

# add margins to visualization:
abline(a = - mod_kernlab@b / params[1],
       b = m, col = "#0072B2")

## Error in abline(a = -mod_kernlab@b/params[1], b = m, col = "#0072B2"):  object
'mod_kernlab' not found

abline(a = - mod_kernlab@b / params[1] + t_0,
       b = m, col = "#0072B2", lty = 2)

## Error in abline(a = -mod_kernlab@b/params[1] + t_0, b = m, col = "#0072B2", :  object
'mod_kernlab' not found
```

```r
abline(a = - mod_kernlab@b / params[1] - t_0,
       b = m, col = "#0072B2", lty = 2)

## Error in abline(a = -mod_kernlab@b/params[1] - t_0, b = m, col = "#0072B2", :  object
'mod_kernlab' not found

# add legends
legend(par('usr')[2], par('usr')[4], , bty='n', xpd=NA, legend=c("1","2"),
       pch=c("-","+"), title="Classes", cex = 0.8)
legend(par('usr')[2], 1.8, , bty='n', xpd=NA,
       legend=c("Pegasos","Logistic","Kernlab","Margin"),
       lty=c(1,3,1,2),
       col = c("#D55E00","#56B4E9","#0072B2","#0072B2"),
       title="", cex = 0.8, lwd = c(1,2,1,1))
```