```
#### Exam 2 - Assignment 3
### Libraries
library(kernlab)
library(neuralnet)

### Setup
data(spam)
C = c(1, 10, 50, 100)
### Functions

### Implementation
## Task 1
set.seed(1234567890)
error.cross = numeric()
i = 1
for(c in C){
  svm = ksvm(type~., data=spam, kernel="rbfdot",
kpar=list(sigma=0.05), C=c, cross=2)
  error.cross[i] = cross(svm)
  i = i+1
}
result = data.frame(C=C, Error=error.cross)
print(result)
plot(result)
## Task 2
# the error estimate may not be mono- tone with respect to the value
of C:
# Comment:
# If C is large enough the model will strive to find the global maxima
with
# as small bias as possible, regardless of how high C is

## Task 3
set.seed(1234567890)

rad = runif(50, 0, 10)
sin = sin(rad)
data = data.frame(rad, sin)
n = dim(data)[1]
left.fold = data[1:25,]
right.fold = data[26:50,]
w.init = runif(31, -1, 1)

# train nueral networks
nn.left = neuralnet(formula=sin~rad, data=left.fold, hidden=c(10),
startweights=w.init, threshold=0.001)
nn.right = neuralnet(formula=sin~rad, data=right.fold, hidden=c(10),
startweights=w.init, threshold=0.001)
# make predictions
yfit.right = compute(nn.left, right.fold$rad)$net.result
yfit.left = compute(nn.right, left.fold$rad)$net.result
# calculate errors
y.right = right.fold$sin
y.left = left.fold$sin
```

```
sse.left = sum((y.left-yfit.left)^2)
sse.right = sum((y.right-yfit.right)^2)
mse = (sse.left+sse.right)/n # 0.001368483974
```