R Notebook

Code

This is an R Markdown (http://rmarkdown.rstudio.com) Notebook. When you execute code within the notebook, the results appear beneath the code.

Try executing this chunk by clicking the *Run* button within the chunk or by placing your cursor inside it and pressing *Cmd+Shift+Enter*.

—SVM classification—

Red dataset

Hide

```
library(kernlab)
library(e1071)
library(MASS)

set.seed(1)
red <- read.csv('winequality-red.csv', header = TRUE, sep=";")
red <- na.omit(red)
red.quality <- red$quality
red[,-12] <- scale(red[,-12])

# Red dataset
train_red_idx <- sample(nrow(red) * 0.8) # 80-20 train-test split
train.red <- red[train_red_idx,]
train.red$quality <- as.factor(train.red$quality)
train.red.quality <- train.red$quality

test.red <- red[-train_red_idx,]
test.red$quality <- as.factor(test.red$quality)
test.red.quality <- test.red$quality
# test.red <- test.red[,-12]

# # Backward model selection
# library(leaps)
# regfit.bwd=regsubsets(quality~.,data=train.red,nvmax=11,method="backward")
# summary(regfit.bwd)
#
# test.mat=model.matrix(quality~.,data=test.red) # create an X matrix of test data
# val.errors=rep(NA,19)
# for(i in 1:19){
#    coefi=coef(regfit.best,id=i)
#    pred=test.mat[,names(coefi)]%*%coefi
#    val.errors[i]=mean((Hitters$Salary[test]-pred)^2)
# }
# val.errors
# which.min(val.errors)
# coef(regfit.best,10)

set.seed(1)

gamma <- 2^seq(-15, 3, 1)
test_accuracy <- rep(0, length(gamma))
fit_time <- rep(0, length(gamma))
i = 1

start <- proc.time()

for(g in gamma){
  start_it <- proc.time()
  model = svm(quality~., data = train.red, gamma=g)
  fit_time[i] <- proc.time() - start_it
  test_accuracy[i] <- mean(test.red.quality == predict(model, test.red))
  i = i + 1
}
```

```
number of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a mul
tiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to r
eplace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthn
umber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t length
```

Hide

```
print(cv_time_red <- proc.time() - start) # total CV time
```

```
   user  system elapsed 
  3.955   0.095   4.214 
```

Hide

```
print(fit_time[which.max(test_accuracy)]) # fit time of best model
```

```
[1] 0.15
```

Hide

```
print(best_gamma_red <- 2^(which.max(test_accuracy) - 16)) # gamma yielding highest test accuracy
```

```
[1] 0.25
```

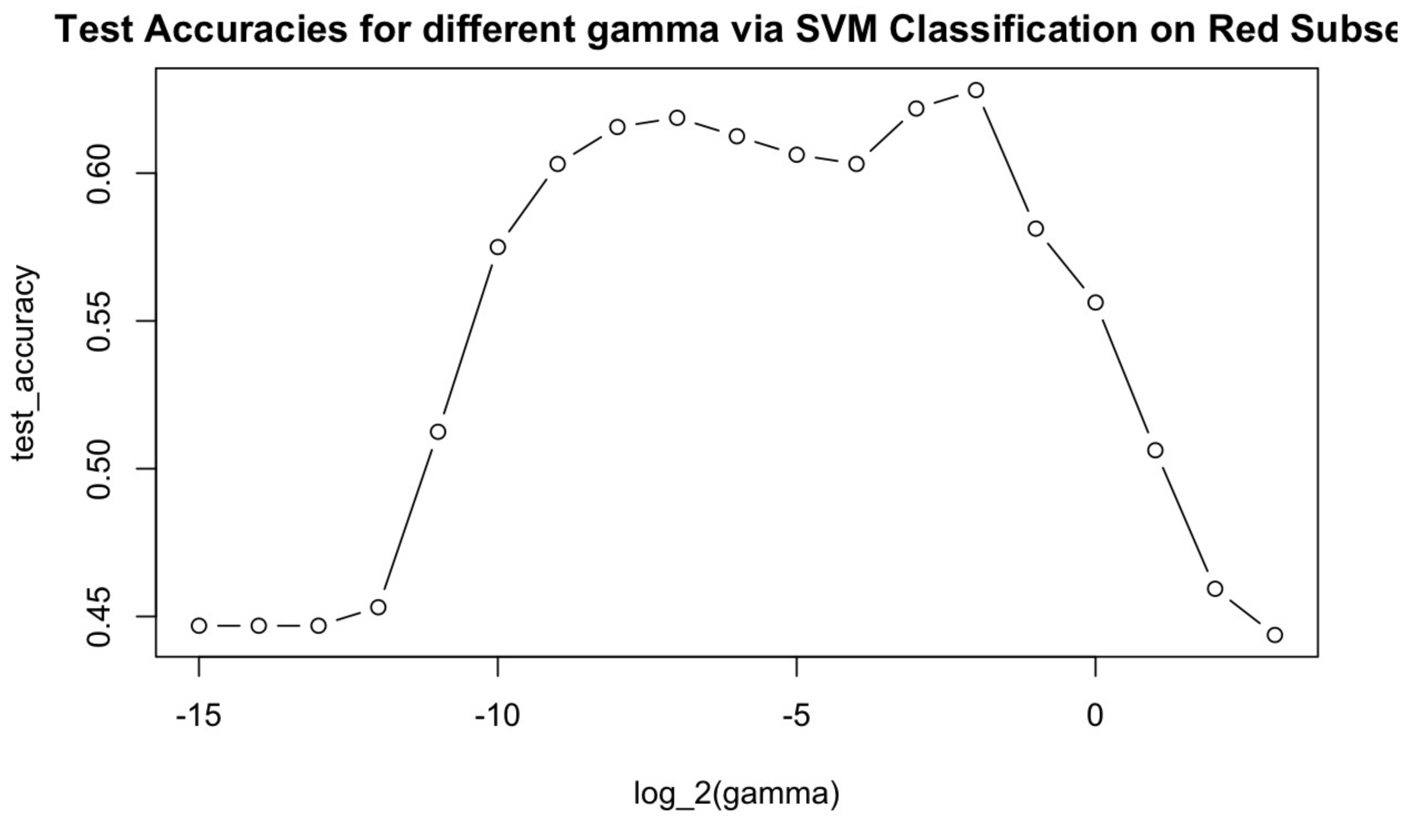
Hide

```
print(test_acc_red <- max(test_accuracy)) # test accuracy
```

```
[1] 0.628125
```

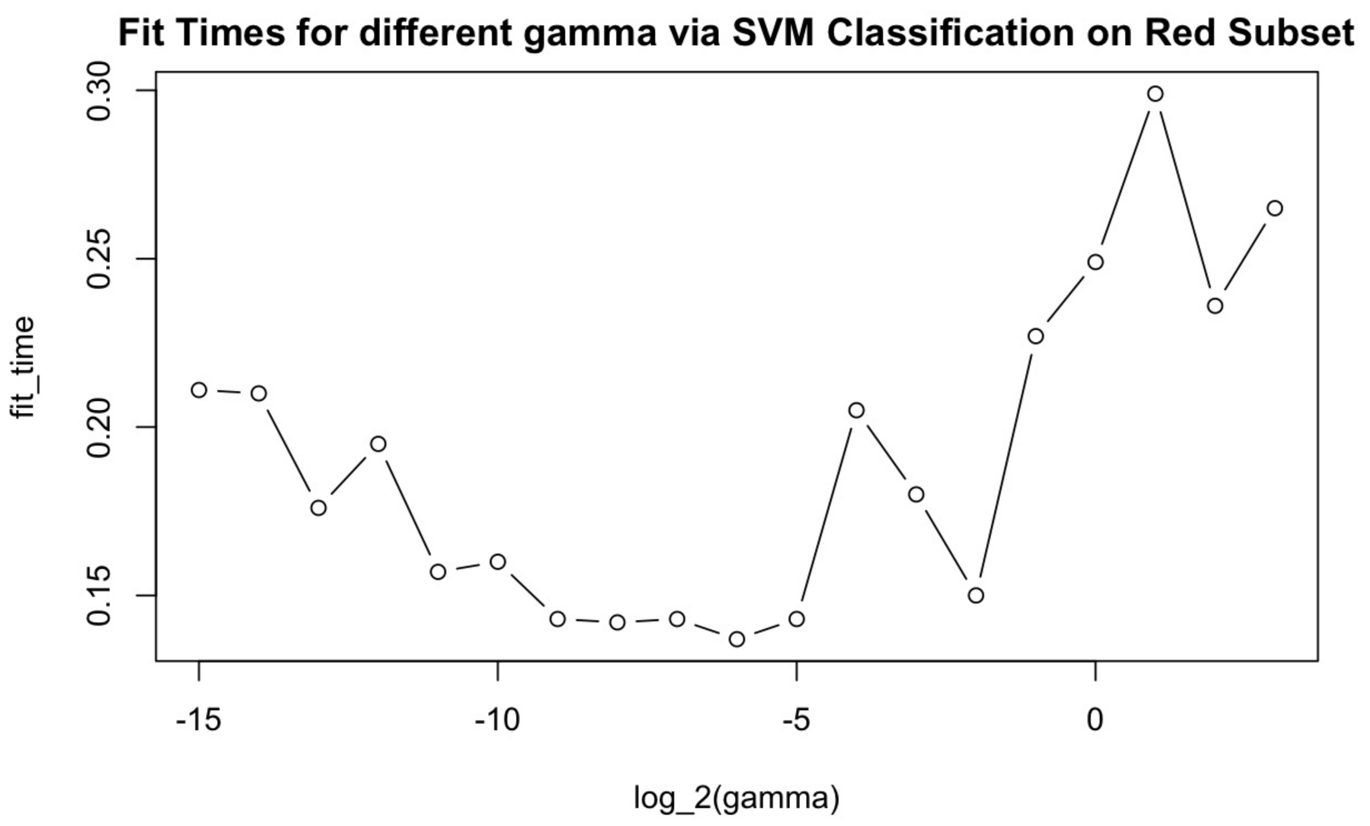
Hide

```
plot(log2(gamma), test_accuracy, type='b', xlab = 'log_2(gamma)', main='Test Accuracies for different gamma via S
VM Classification on Red Subset')
```



Hide

```
plot(log2(gamma), fit_time, type='b', xlab = 'log_2(gamma)', main='Fit Times for different gamma via SVM Classifi
cation on Red Subset')
```



Hide

```
NA
NA
```

White dataset

Hide

```
library(kernlab)
library(e1071)
```

```
package ‘e1071’ was built under R version 3.6.2
```

Hide

```r
library(MASS)

white <- read.csv('winequality-white.csv', header = TRUE, sep=";")
white <- na.omit(white)
white.quality <- white$quality
white[,-12] <- scale(white[,-12])

set.seed(1)
train_white_idx <- sample(nrow(white) * 0.8) # 80-20 train-test split
train.white <- white[train_white_idx,]
train.white$quality <- as.factor(train.white$quality)
train.white.quality <- train.white$quality

test.white <- white[-train_white_idx,]
test.white$quality <- as.factor(test.white$quality)
test.white.quality <- test.white$quality
# test.white <- test.white[,-12]

# # Backward model selection
# library(leaps)
# regfit.bwd=regsubsets(quality~.,data=train.white,nvmax=11,method="backward")
# summary(regfit.bwd)
#
# test.mat=model.matrix(quality~.,data=test.white) # create an X matrix of test data
# val.errors=rep(NA,19)
# for(i in 1:19){
#    coefi=coef(regfit.best,id=i)
#    pred=test.mat[,names(coefi)]%*%coefi
#    val.errors[i]=mean((Hitters$Salary[test]-pred)^2)
# }
# val.errors
# which.min(val.errors)
# coef(regfit.best,10)

set.seed(1)

gamma <- 2^seq(-15, 3, 1)
test_accuracy <- rep(0, length(gamma))
fit_time <- rep(0, length(gamma))
i = 1

start <- proc.time()

for(g in gamma){
  start_it <- proc.time()
  model = svm(quality~., data = train.white, gamma=g)
  fit_time[i] <- proc.time() - start_it
  test_accuracy[i] <- mean(as.character(test.white.quality) == as.character(predict(model, test.white)))
  i = i + 1
}
```

```
number of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a mul
tiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to r
eplace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthn
umber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t length
```

Hide

```r
print(cv_time_white <- proc.time() - start) # total CV time
```

```
   user  system elapsed 
 36.059   0.505  37.336 
```

Hide

```r
print(fit_time[which.max(test_accuracy)]) # fit time of best model
```

```
[1] 1.668
```

Hide

```
print(best_gamma_white <- 2^(which.max(test_accuracy) - 16)) # gamma yielding highest test accuracy
```

```
[1] 0.015625
```

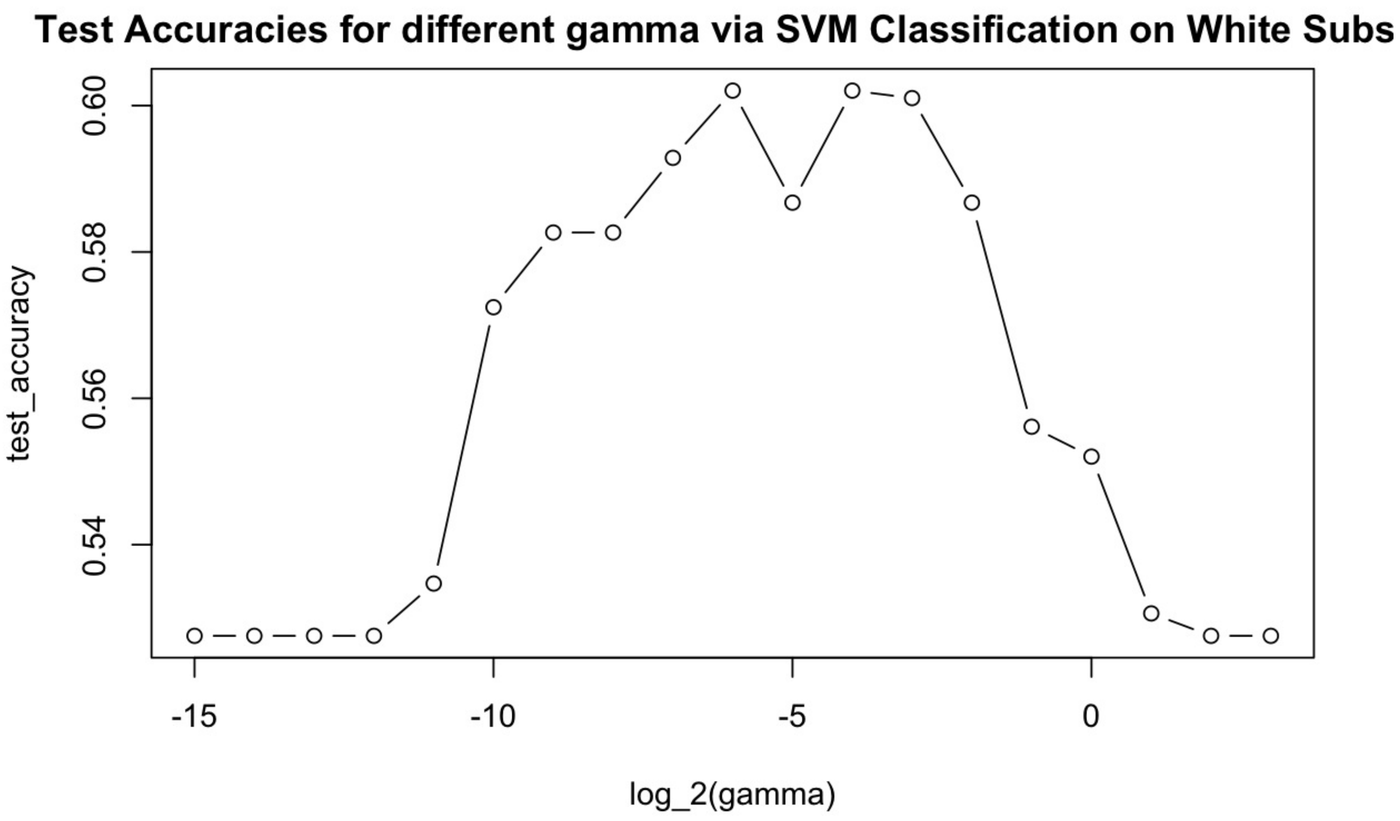
Hide

```
print(test_acc_white <- max(test_accuracy)) # test accuracy
```

```
[1] 0.6020408
```

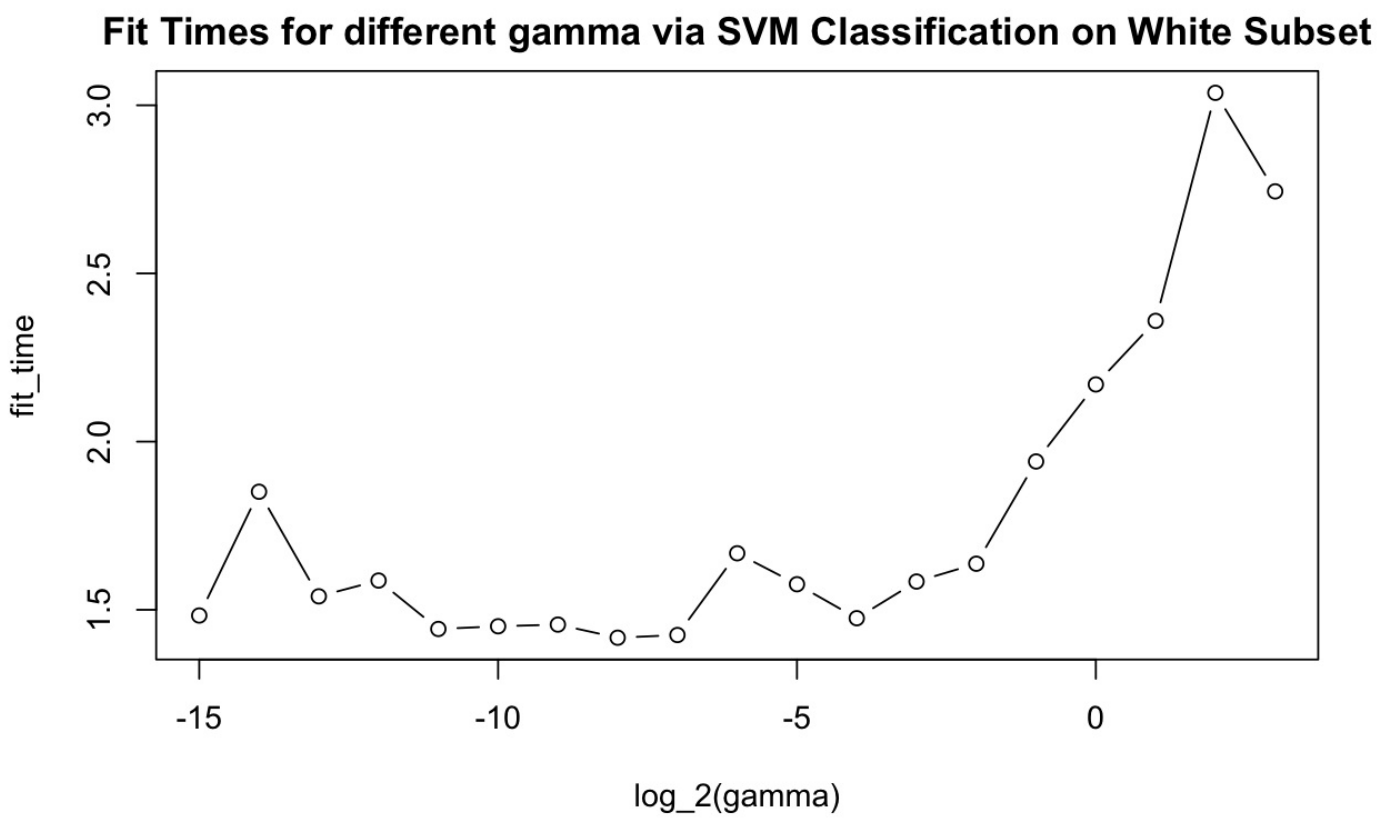
Hide

```
plot(log2(gamma), test_accuracy, type='b', xlab = 'log_2(gamma)', main='Test Accuracies for different gamma via S
VM Classification on White Subset')
```



Hide

```
plot(log2(gamma), fit_time, type='b', xlab = 'log_2(gamma)', main='Fit Times for different gamma via SVM Classifi
cation on White Subset')
```



Weighted Test Accuracy

Hide

```
(nrow(test.red) * test_acc_red + nrow(test.white) * test_acc_white) / (nrow(test.red) + nrow(test.white))
```

```
[1] 0.6084615
```

— SVM regression (epsilon-insensitive loss) —

Red dataset

Hide

```
set.seed(1)
red <- read.csv('winequality-red.csv', header = TRUE, sep=";")
red <- na.omit(red)
red.quality <- red$quality
red[,-12] <- scale(red[,-12])

# Red dataset
train_red_idx <- sample(nrow(red) * 0.8) # 80-20 train-test split
train.red <- red[train_red_idx,]
# train.red$quality <- as.factor(train.red$quality)
train.red.quality <- train.red$quality

test.red <- red[-train_red_idx,]
# test.red$quality <- as.factor(test.red$quality)
test.red.quality <- test.red$quality
# test.red <- test.red[,-12]

# # Backward model selection
# library(leaps)
# regfit.bwd=regsubsets(quality~.,data=train.red,nvmax=11,method="backward")
# summary(regfit.bwd)
#
# test.mat=model.matrix(quality~.,data=test.red) # create an X matrix of test data
# val.errors=rep(NA,19)
# for(i in 1:19){
#    coefi=coef(regfit.best,id=i)
#    pred=test.mat[,names(coefi)]%*%coefi
#    val.errors[i]=mean((Hitters$Salary[test]-pred)^2)
# }
# val.errors
# which.min(val.errors)
# coef(regfit.best,10)

set.seed(1)

gamma <- 2^seq(-15, 3, 1)
test_accuracy <- rep(0, length(gamma))
fit_time <- rep(0, length(gamma))
i = 1

start <- proc.time()

for(g in gamma){
  start_it <- proc.time()
  model = svm(quality~., data = train.red, gamma=g, type='eps-regression')
  fit_time[i] <- proc.time() - start_it
  test_accuracy[i] <- mean(abs(test.red.quality - predict(model, test.red)) <= 0.5)
  i = i + 1
}
```

```
number of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a mul
tiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to r
eplace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthn
umber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t length
```

Hide

```
print(cv_time_red <- proc.time() - start) # total CV time
```

```
   user  system elapsed 
  3.205   0.057   3.267 
```

Hide

```
print(fit_time[which.max(test_accuracy)]) # fit time of best model
```

```
[1] 0.128
```

Hide

```
print(best_gamma_red <- 2^(which.max(test_accuracy) - 16)) # gamma yielding highest test accuracy
```

```
[1] 0.001953125
```

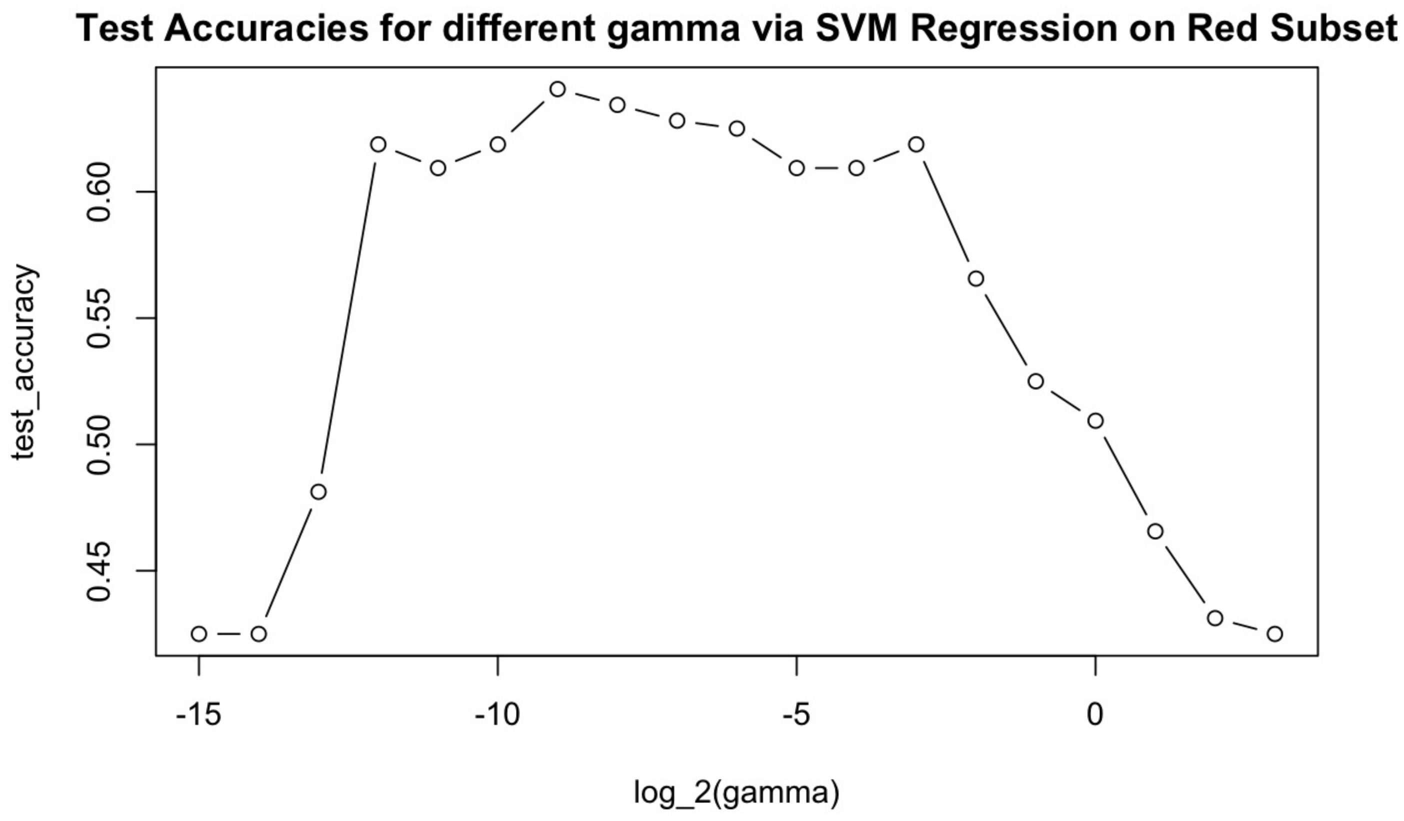
Hide

```
print(test_acc_red <- max(test_accuracy)) # test accuracy
```

```
[1] 0.640625
```

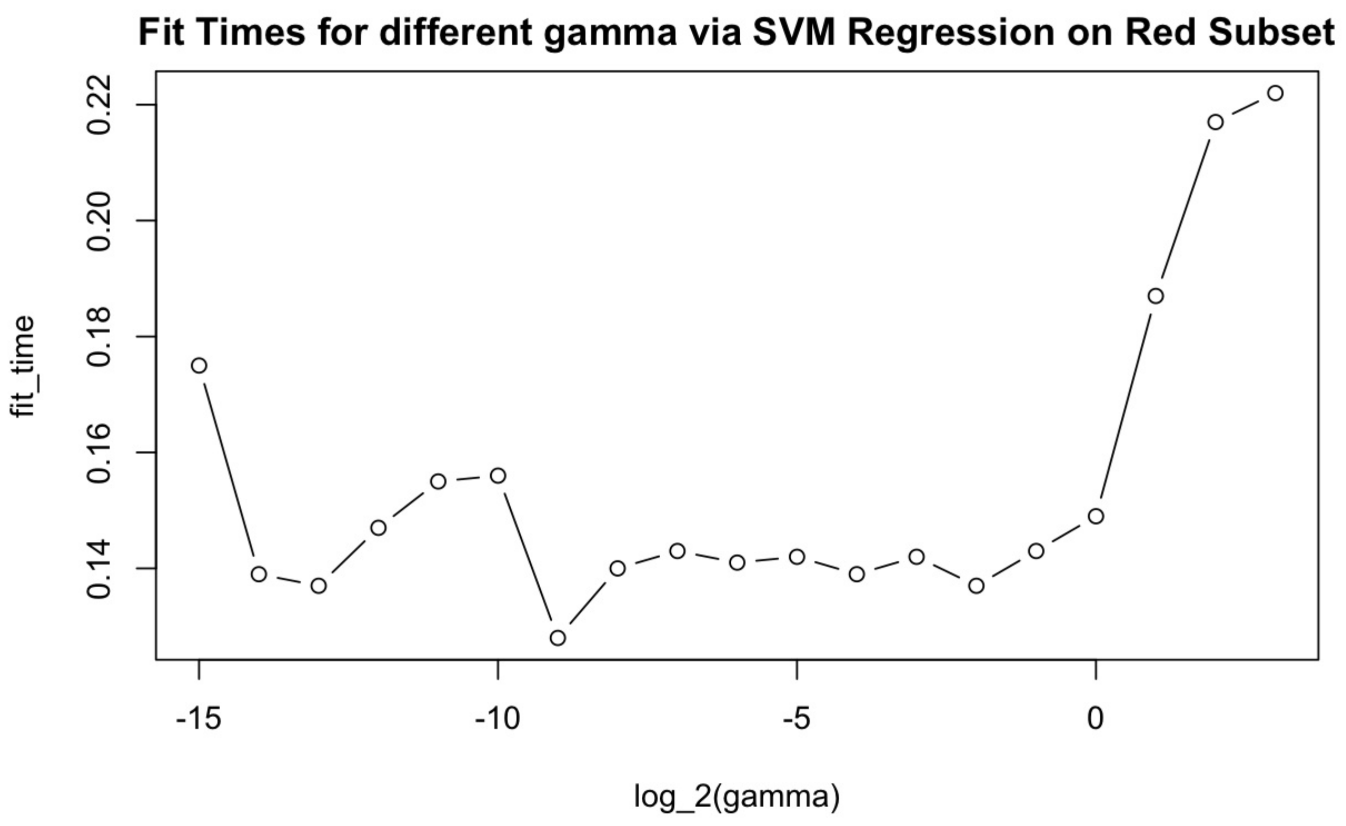
Hide

```
plot(log2(gamma), test_accuracy, type='b', xlab = 'log_2(gamma)', main='Test Accuracies for different gamma via S
VM Regression on Red Subset')
```



Hide

```
plot(log2(gamma), fit_time, type='b', xlab = 'log_2(gamma)', main='Fit Times for different gamma via SVM Regressi
on on Red Subset')
```



White dataset

Hide

```
white <- read.csv('winequality-white.csv', header = TRUE, sep=";")
white <- na.omit(white)
white.quality <- white$quality
white[,-12] <- scale(white[,-12])

set.seed(1)
train_white_idx <- sample(nrow(white) * 0.8) # 80-20 train-test split
train.white <- white[train_white_idx,]
# train.white$quality <- as.factor(train.white$quality)
train.white.quality <- train.white$quality

test.white <- white[-train_white_idx,]
# test.white$quality <- as.factor(test.white$quality)
test.white.quality <- test.white$quality
# test.white <- test.white[,-12]

# # Backward model selection
# library(leaps)
# regfit.bwd=regsubsets(quality~.,data=train.white,nvmax=11,method="backward")
# summary(regfit.bwd)
#
# test.mat=model.matrix(quality~.,data=test.white) # create an X matrix of test data
# val.errors=rep(NA,19)
# for(i in 1:19){
#    coefi=coef(regfit.best,id=i)
#    pred=test.mat[,names(coefi)]%*%coefi
#    val.errors[i]=mean((Hitters$Salary[test]-pred)^2)
# }
# val.errors
# which.min(val.errors)
# coef(regfit.best,10)

set.seed(1)

gamma <- 2^seq(-15, 3, 1)
test_accuracy <- rep(0, length(gamma))
fit_time <- rep(0, length(gamma))
i = 1

start <- proc.time()

for(g in gamma){
  start_it <- proc.time()
  model = svm(quality~., data = train.white, gamma=g, type='eps-regression')
  fit_time[i] <- proc.time() - start_it
  test_accuracy[i] <- mean(abs(test.white.quality - predict(model, test.white)) <= 0.5)
  i = i + 1
}
```

```
number of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a mul
tiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to r
eplace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthn
umber of items to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of
replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of items to replace is
not a multiple of replacement lengthnumber of items to replace is not a multiple of replacement lengthnumber of i
tems to replace is not a multiple of replacement lengthnumber of items to replace is not a multiple of replacemen
t length
```

Hide

```
print(cv_time_white <- proc.time() - start) # total CV time
```

```
   user  system elapsed 
 27.852   0.369  28.544 
```

Hide

```
print(fit_time[which.max(test_accuracy)]) # fit time of best model
```

```
[1] 1.276
```

Hide

```
print(best_gamma_white <- 2^(which.max(test_accuracy) - 16)) # gamma yielding highest test accuracy
```

```
[1] 0.125
```

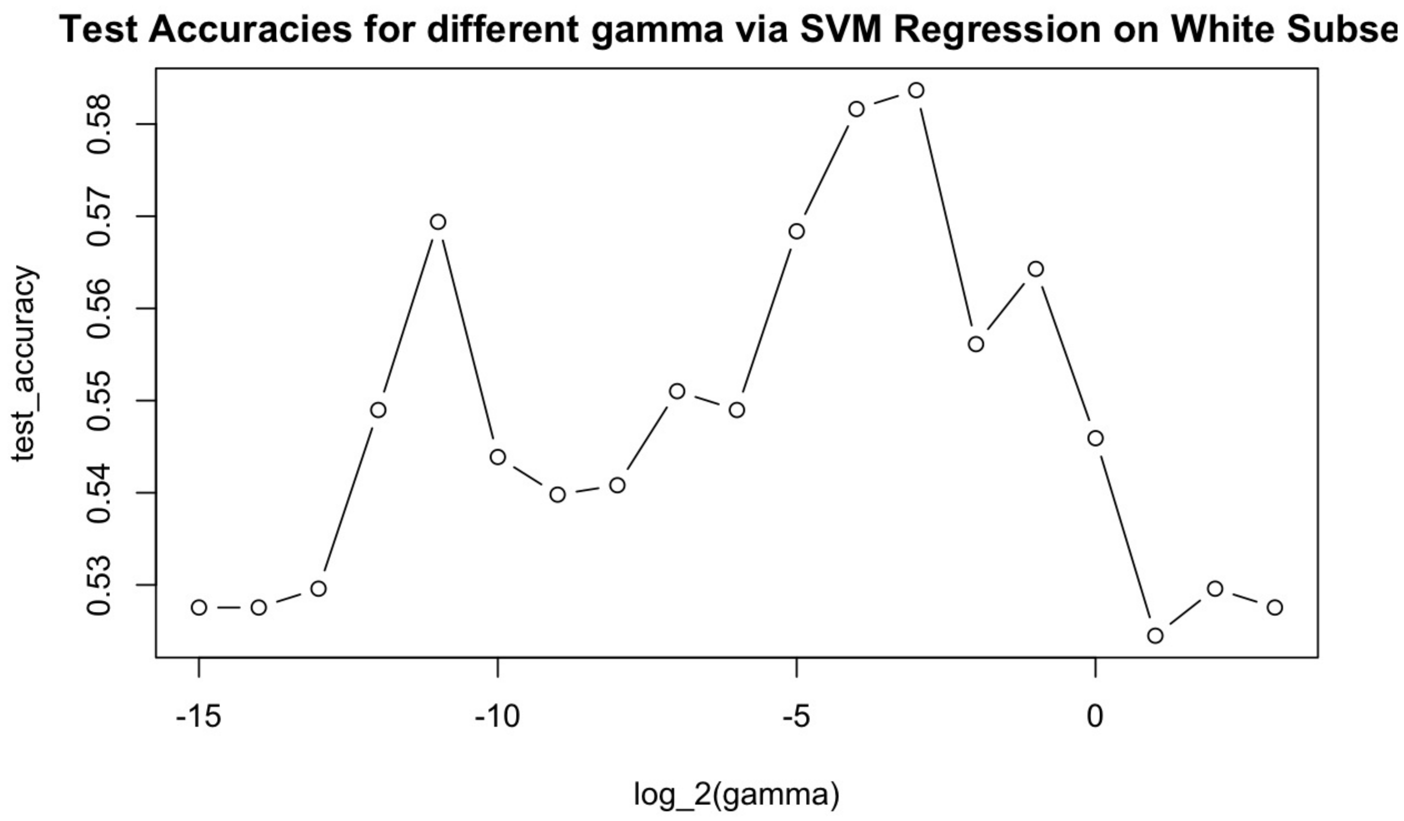
Hide

```
print(test_acc_white <- max(test_accuracy)) # test accuracy
```

```
[1] 0.5836735
```

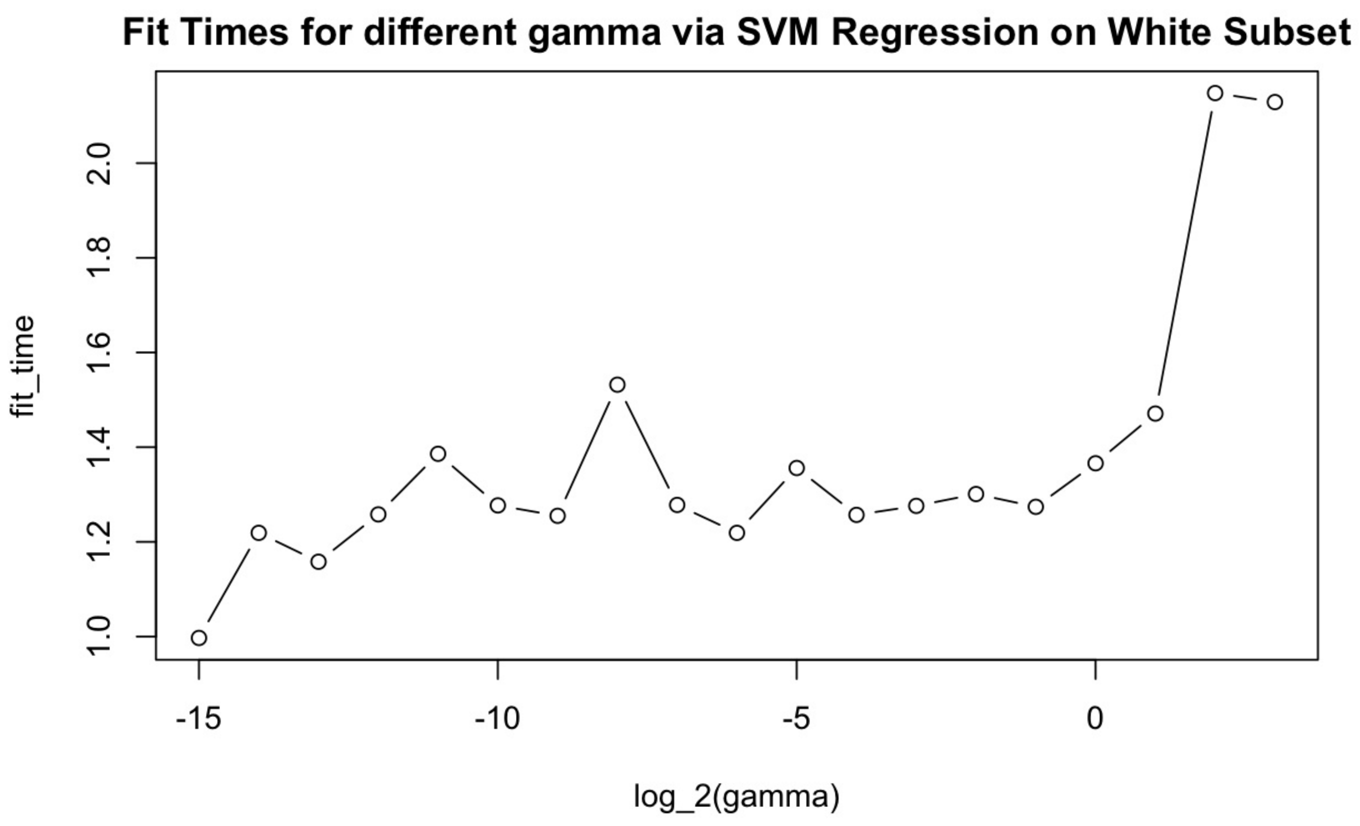
Hide

```
plot(log2(gamma), test_accuracy, type='b', xlab = 'log_2(gamma)', main='Test Accuracies for different gamma via S
VM Regression on White Subset')
```



Hide

```
plot(log2(gamma), fit_time, type='b', xlab = 'log_2(gamma)', main='Fit Times for different gamma via SVM Regressi
on on White Subset')
```



Weighted Test Accuracy

```
(nrow(test.red) * test_acc_red + nrow(test.white) * test_acc_white) / (nrow(test.red) + nrow(test.white))
```

```
[1] 0.5976923
```

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.

When you save the notebook, an HTML file containing the code and output will be saved alongside it (click the *Preview* button or press *Cmd+Shift+K* to preview the HTML file).

The preview shows you a rendered HTML copy of the contents of the editor. Consequently, unlike *Knit*, *Preview* does not run any R code chunks. Instead, the output of the chunk when it was last run in the editor is displayed.