```
##Linearly separable case
#set working directory if needed (modify path as needed)
setwd("C:/Users/Kailash/Documents/svm")
#load required library
library(e1071)
#load built-in iris dataset
data(iris)
#set seed to ensure reproducible results
set.seed(42)
#split into training and test sets
iris[,"train"] <- ifelse(runif(nrow(iris))<0.8,1,0)
#separate training and test sets
trainset <- iris[iris$train==1,]
testset <- iris[iris$train==0,]
#get column index of train flag
trainColNum <- grep("train",names(trainset))
#remove train flag column from train and test sets
trainset <- trainset[,-trainColNum]
testset <- testset[,-trainColNum]
#get column index of predicted variable in dataset
typeColNum <- grep("Species",names(iris))
#build model – linear kernel and C-classification (soft margin) with default cost (C=1)
svm_model <- svm(Species~ ., data=trainset, method="C-classification", kernel="linear")
#training set predictions
pred_train <-predict(svm_model,trainset)
mean(pred_train==trainset$Species)
#test set predictions
pred_test <-predict(svm_model,testset)
mean(pred_test==testset$Species)

##Non linearly separable data; linear kernel
#load required library (assuming e1071 is already loaded)
library(mlbench)
#load Sonar dataset
data(Sonar)
#set seed to ensure reproducible results
set.seed(42)
#split into training and test sets
Sonar[,"train"] <- ifelse(runif(nrow(Sonar))<0.8,1,0)
#separate training and test sets
trainset <- Sonar[Sonar$train==1,]
testset <- Sonar[Sonar$train==0,]
#get column index of train flag
trainColNum <- grep("train",names(trainset))
#remove train flag column from train and test sets
trainset <- trainset[,-trainColNum]
testset <- testset[,-trainColNum]
#get column index of predicted variable in dataset
typeColNum <- grep("Class",names(Sonar))
#build model – linear kernel and C-classification with default cost (C=1)
svm_model <- svm(Class~ ., data=trainset, method="C-classification", kernel="linear")
#training set predictions
pred_train <-predict(svm_model,trainset)
```

```
mean(pred_train==trainset$Class)
#test set predictions
pred_test <-predict(svm_model,testset)
mean(pred_test==testset$Class)

##Non linear data; radial kernel, default params
#build model: radial kernel, default params
svm_model <- svm(Class~ ., data=trainset, method="C-classification", kernel="radial")
#print params
svm_model$cost
svm_model$gamma
#training set predictions
pred_train <-predict(svm_model,trainset)
mean(pred_train==trainset$Class)
#test set predictions
pred_test <-predict(svm_model,testset)
mean(pred_test==testset$Class)

##Non linear data; radial kernel; tuned parameters
#find optimal parameters in a specified range
tune_out <- tune.svm(x=trainset[,-
typeColNum],y=trainset[,typeColNum],gamma=10^(-3:3),cost=c(0.01,0.1,1,10,100,1000),kernel=”radial”)
#print best values of cost and gamma
tune_out$best.parameters$cost
tune_out$best.parameters$gamma
#build model
svm_model <- svm(Class~ ., data=trainset, method="C-classification",
kernel="radial",cost=tune_out$best.parameters$cost,gamma=tune_out$best.parameters$gamma)
#training set predictions
pred_train <-predict(svm_model,trainset)
mean(pred_train==trainset$Class)
#test set predictions
pred_test <-predict(svm_model,testset)
mean(pred_test==testset$Class)
```