Support Vector Machine, classification

```r
Titanic <- read.csv("Titanic.csv")
Titanic <- Titanic[,-c(1,4,9,11)]
Titanic <- Titanic[-which(is.na(Titanic$Age)),]
Titanic$Survived <- as.factor(Titanic$Survived)
Titanic$Pclass <- as.factor(Titanic$Pclass)
Titanic$Age <- scale(Titanic$Age)
Titanic$Fare <- scale(Titanic$Fare)
cat("There are",nrow(Titanic), "passengers left.")
```

```
## There are 714 passengers left.
```

```r
library(caret)
```

```
## Loading required package: ggplot2

## Loading required package: lattice
```

```r
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v lubridate 1.9.3     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.1

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## x purrr::lift()   masks caret::lift()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
set.seed(123)
training_samples <- Titanic$Survived %>%
  createDataPartition(p=0.75,list=FALSE)
train_data <- Titanic[training_samples,]
test_data <- Titanic[-training_samples,]
nrow(train_data)
```

```
## [1] 536
```

```r
nrow(test_data)
```

```
## [1] 178
```

```r
library(kernlab)
```

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:purrr':
##
##     cross
```

```
## The following object is masked from 'package:ggplot2':
##
##     alpha
```

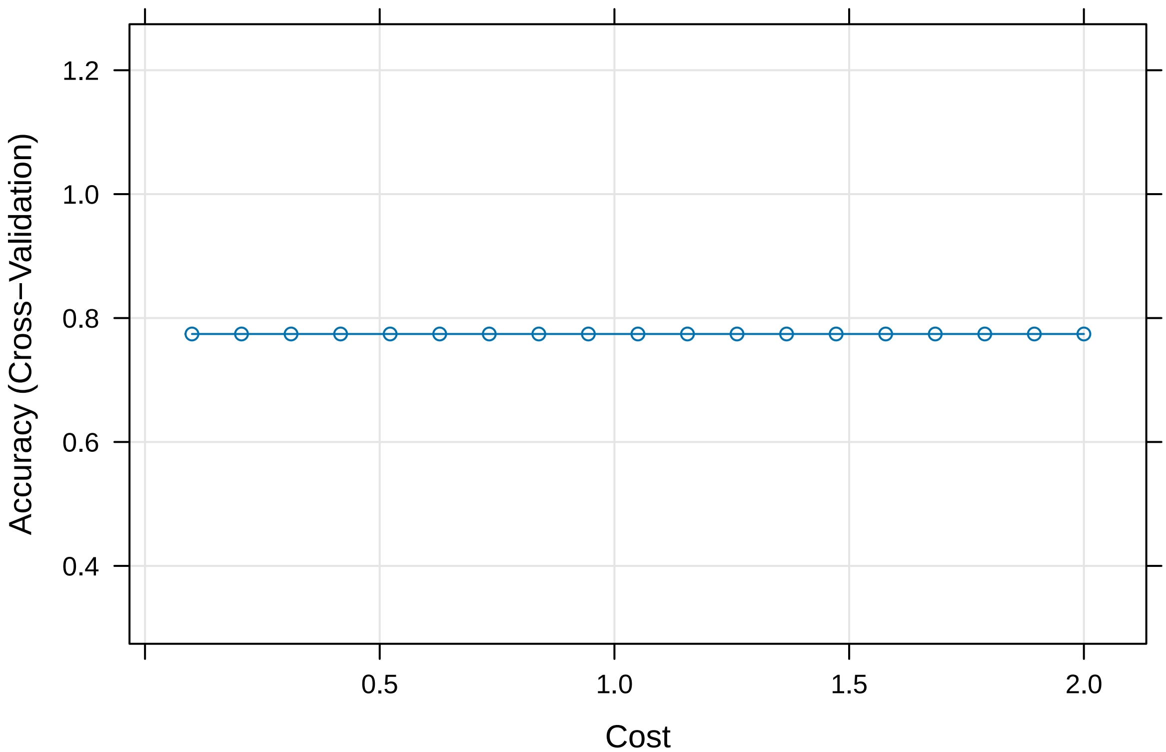
```
set.seed(123)
model <- train(Survived~., data=train_data, method="svmLinear", trControl = trainControl("cv",number = 
predicted_class <- model %>% predict(test_data)

confusionMatrix(factor(predicted_class),factor(test_data$Survived),positive='1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 93 23
##          1 13 49
##
##                Accuracy : 0.7978
##                  95% CI : (0.7312, 0.8541)
##     No Information Rate : 0.5955
##     P-Value [Acc > NIR] : 7.457e-09
##
##                   Kappa : 0.5706
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.6806
##             Specificity : 0.8774
##          Pos Pred Value : 0.7903
##          Neg Pred Value : 0.8017
##              Prevalence : 0.4045
##          Detection Rate : 0.2753
##    Detection Prevalence : 0.3483
##       Balanced Accuracy : 0.7790
##
##        'Positive' Class : 1
##
```

```
set.seed(123)
model <- train(Survived~.,data=train_data,method="svmLinear",trControl=trainControl("cv",number=10),tun

plot(model) #plot model accuracy vs. different values of Cost
```




```
model$bestTune #the best tuning parameter that maximizes model accuracy
```

```
##     C
## 1 0.1
```

```
model <- train(Survived~.,data=train_data,method="svmLinear",trControl=trainControl("cv",number=10),tun

predicted_class <- model %>% predict(test_data)
confusionMatrix(factor(predicted_class),factor(test_data$Survived),positive='1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 93 23
##          1 13 49
##
##                Accuracy : 0.7978
##                  95% CI : (0.7312, 0.8541)
##     No Information Rate : 0.5955
##     P-Value [Acc > NIR] : 7.457e-09
##
##                   Kappa : 0.5706
##
##  Mcnemar's Test P-Value : 0.1336
##
##             Sensitivity : 0.6806
##             Specificity : 0.8774
##          Pos Pred Value : 0.7903
##          Neg Pred Value : 0.8017
##              Prevalence : 0.4045
```

```
##          Detection Rate : 0.2753
##    Detection Prevalence : 0.3483
##       Balanced Accuracy : 0.7790
##
##        'Positive' Class : 1
##
```

```r
set.seed(123)
model <- train(Survived~.,data=train_data,method="svmRadial",trControl=trainControl("cv",number=10),tun
model$bestTune # Print the best tuning parameter sigma and C that maximizes model accuracy
```

```
##       sigma C
## 3 0.1420266 1
```

```r
predicted_class <- model %>% predict(test_data)

confusionMatrix(factor(predicted_class),factor(test_data$Survived),positive='1')
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 90 17
##          1 16 55
##
##                Accuracy : 0.8146
##                  95% CI : (0.7496, 0.8688)
##     No Information Rate : 0.5955
##     P-Value [Acc > NIR] : 3.229e-10
##
##                   Kappa : 0.6143
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.7639
##             Specificity : 0.8491
##          Pos Pred Value : 0.7746
##          Neg Pred Value : 0.8411
##              Prevalence : 0.4045
##          Detection Rate : 0.3090
##    Detection Prevalence : 0.3989
##       Balanced Accuracy : 0.8065
##
##        'Positive' Class : 1
##
```

```r
set.seed(123)
model <- train(Survived~., data=train_data, method="svmPoly",trControl=trainControl("cv",number=10),tun
model$bestTune # Print the best tuning parameter sigma and C that maximizes model accuracy
```

```
##    degree scale    C
## 25      2   0.1 0.25
```

```r
predicted_class <- model %>% predict(test_data)

confusionMatrix(factor(predicted_class),factor(test_data$Survived),positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 93 21
##          1 13 51
##
##                Accuracy : 0.809
##                  95% CI : (0.7434, 0.8639)
##     No Information Rate : 0.5955
##     P-Value [Acc > NIR] : 9.523e-10
##
##                   Kappa : 0.5963
##
##  Mcnemar's Test P-Value : 0.2299
##
##             Sensitivity : 0.7083
##             Specificity : 0.8774
##          Pos Pred Value : 0.7969
##          Neg Pred Value : 0.8158
##              Prevalence : 0.4045
##          Detection Rate : 0.2865
##    Detection Prevalence : 0.3596
##       Balanced Accuracy : 0.7928
##
##        'Positive' Class : 1
##
```