Support Vector Machines in R

Data preparation

- Install packages “e1071”, “rpart”, and “mlbench”
  - > library(e1071)
  - > library(rpart)
  - > data(Glass, package=“mlbench”)
  - > Glass

- Split data into a train and test set
  - > set.seed(1234)
  - > ind = sample(2, nrow(Glass), replace=TRUE, prob=c(0.67, 0.33)
  - > trainset = Glass[ind==1,]
  - > testset = Glass[ind==2,]

Support vector machines (SVM)

- SVM
  - > svm.model = svm(Type ~ ., data = trainset, cost = 100, gamm a = 1)
  - > svm.pred = predict(svm.model, testset[,-10])

- Compare SVM to regression trees as implemented in rpart()
  - > rpart.model = rpart(Type ~ ., data = trainset)
  - > rpart.pred = predict(rpart.model, testset[,-10], type="class")

Confusion matrix

- Compute SVM confusion matrix
  - > svm.table = table(pred = svm.pred, true = testset[,10])

```
      true
pred   1  2  3  5  6  7
   1  14  5  4  0  0  0
   2   9 19  0  5  3  4
   3   0  2  1  0  0  0
   5   0  0  0  0  0  0
   6   0  0  0  0  1  0
   7   0  0  0  0  0  4
```

- Compute rpart confusion matrix
  - > rpart.table = table(pred = rpart.pred, true = testset[,10])

```
      true
pred   1  2  3  5  6  7
   1  19 14  5  0  0  1
   2   3 11  0  0  3  0
   3   0  0  0  0  0  0
   5   0  1  0  4  0  0
   6   0  0  0  0  1  1
   7   1  0  0  1  0  6
```

Performance evaluation

- Accuracy rates and the kappa indices
- > tab = (svm.table, rpart.table)
- > classAgreement(tab)

	method	Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
Accuracy	svm	0.61	0.64	0.68	0.67	0.69	0.77
	rpart	0.46	0.49	0.54	0.53	0.56	0.66
Kappa	svm	0.58	0.62	0.66	0.66	0.7	0.73
	rpart	0.42	0.49	0.53	0.53	0.58	0.61

Non-linear ε-Regression

```
> library(e1071)
> library(rpart)
> data(Ozone, package="mlbench")
> ## split data into a train and test set
> index     <- 1:nrow(Ozone)
> testindex <- sample(index, trunc(length(index)/3))
> testset   <- na.omit(Ozone[testindex,-3])
> trainset  <- na.omit(Ozone[-testindex,-3])

> ## svm
> svm.model <- svm(V4 ~ ., data = trainset, cost = 1000, gamma = 0.0001)
> svm.pred  <- predict(svm.model, testset[,-3])
> crossprod(svm.pred - testset[,3]) / length(testindex)


 > ## rpart
 > rpart.model <- rpart(V4 ~ ., data = trainset)
 > rpart.pred  <- predict(rpart.model, testset[,-3])
 > crossprod(rpart.pred - testset[,3]) / length(testindex)
```