SVM Regression

Load packages and data.

```
library(e1071)
library(MASS)
df <- Boston[]
```

Divide into train, test, validate

```
set.seed(1234)
spec <- c(train=.6, test=.2, validate=.2)
i <- sample(cut(1:nrow(df),
                nrow(df)*cumsum(c(0,spec)), labels=names(spec)))
train <- df[i=="train",]
test <- df[i=="test",]
vald <- df[i=="validate",]
```

Try linear regression

```
lm1 <- lm(medv~., data=train)
pred <- predict(lm1, newdata=test)
cor_lm1 <- cor(pred, test$medv)
mse_lm1 <- mean((pred-test$medv)^2)
```

Try a linear kernel

```
svm1 <- svm(medv~., data=train, kernel="linear", cost=10, scale=TRUE)
summary(svm1)
```

```
##
## Call:
## svm(formula = medv ~ ., data = train, kernel = "linear", cost = 10,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  linear
##        cost:  10
##       gamma:  0.07692308
##     epsilon:  0.1
##
##
## Number of Support Vectors:  226
```

```
pred <- predict(svm1, newdata=test)
cor_svm1 <- cor(pred, test$medv)
```

```r
mse_svm1 <- mean((pred - test$medv)^2)
```

Tune

```r
tune_svm1 <- tune(svm, medv~., data=vald, kernel="linear",
                  ranges=list(cost=c(0.001, 0.01, 0.1, 1, 5, 10, 100)))
summary(tune_svm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##  0.01
##
## - best performance: 40.52242
##
## - Detailed performance results:
##    cost    error dispersion
## 1 1e-03 70.78493   46.59835
## 2 1e-02 40.52242   34.84187
## 3 1e-01 41.12805   53.45971
## 4 1e+00 46.79177   68.15944
## 5 5e+00 47.21344   69.51412
## 6 1e+01 47.25101   69.51757
## 7 1e+02 47.31055   69.48502
```

Evaluate on best linear svm

Since our validation set is small, only about 100 observations, we probably did not get hyperparameters that generalize to the full data set.

```r
pred <- predict(tune_svm1$best.model, newdata=test)
cor_svm1_tune <- cor(pred, test$medv)
mse_svm1_tune <- mean((pred - test$medv)^2)
```

Try a polynomial kernel

```r
svm2 <- svm(medv~., data=train, kernel="polynomial", cost=10, scale=TRUE)
summary(svm2)
```

```
##
## Call:
## svm(formula = medv ~ ., data = train, kernel = "polynomial", cost = 10,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
```

```
##       gamma:  0.07692308
##      coef.0:  0
##     epsilon:  0.1
##
##
## Number of Support Vectors:  232
```

```
pred <- predict(svm2, newdata=test)
cor_svm2 <- cor(pred, test$medv)
mse_svm2 <- mean((pred - test$medv)^2)
```

Try a radial kernel

```
svm3 <- svm(medv~., data=train, kernel="radial", cost=10, gamma=1, scale=TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = medv ~ ., data = train, kernel = "radial", cost = 10,
##     gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  10
##       gamma:  1
##     epsilon:  0.1
##
##
## Number of Support Vectors:  227
```

```
pred <- predict(svm3, newdata=test)
cor_svm3 <- cor(pred, test$medv)
mse_svm3 <- mean((pred - test$medv)^2)
```

Tune hyperperameters

```
set.seed(1234)
tune.out <- tune(svm, medv~., data=vald, kernel="radial",
                 ranges=list(cost=c(0.1,1,10,100,1000),
                             gamma=c(0.5,1,2,3,4)))
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##    10   0.5
##
## - best performance: 58.02112
##
```

```
## - Detailed performance results:
##     cost gamma     error dispersion
## 1  1e-01   0.5  91.15533   72.10177
## 2  1e+00   0.5  62.00383   61.26624
## 3  1e+01   0.5  58.02112   55.07826
## 4  1e+02   0.5  58.02112   55.07826
## 5  1e+03   0.5  58.02112   55.07826
## 6  1e-01   1.0  96.65288   71.81208
## 7  1e+00   1.0  73.76509   63.26715
## 8  1e+01   1.0  71.34190   57.52486
## 9  1e+02   1.0  71.34190   57.52486
## 10 1e+03   1.0  71.34190   57.52486
## 11 1e-01   2.0  99.93523   71.68509
## 12 1e+00   2.0  86.30956   65.19410
## 13 1e+01   2.0  83.60352   57.93906
## 14 1e+02   2.0  83.60352   57.93906
## 15 1e+03   2.0  83.60352   57.93906
## 16 1e-01   3.0 101.18045   71.97354
## 17 1e+00   3.0  92.47475   66.46731
## 18 1e+01   3.0  90.16279   58.87913
## 19 1e+02   3.0  90.16279   58.87913
## 20 1e+03   3.0  90.16279   58.87913
## 21 1e-01   4.0 101.77521   72.13671
## 22 1e+00   4.0  95.87078   67.20413
## 23 1e+01   4.0  93.97494   59.60295
## 24 1e+02   4.0  93.97494   59.60295
## 25 1e+03   4.0  93.97494   59.60295
```

```r
svm4 <- svm(medv~., data=train, kernel="radial", cost=100, gamma=0.5, scale=TRUE)
summary(svm4)
```

```
##
## Call:
## svm(formula = medv ~ ., data = train, kernel = "radial", cost = 100,
##     gamma = 0.5, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  100
##       gamma:  0.5
##     epsilon:  0.1
##
##
## Number of Support Vectors:  215
```

```r
pred <- predict(svm4, newdata=test)
cor_svm4 <- cor(pred, test$medv)
mse_svm4 <- mean((pred - test$medv)^2)
```