Statistical Leraning/ Lab 4

Sascha Strobl

5/16/2018

Support Vector Machine classificationa and hierarchical clustering

Load dataset and remove two columns:

```
setwd("~/Documents/CGUClasses/ZOLDClasses/Statistical Learning/Lab4");
GH = read.table("glendalehousing.txt", header = T);
drops <- c("Address","LotArea.1")
GH1=GH[ , !(names(GH) %in% drops)]
```

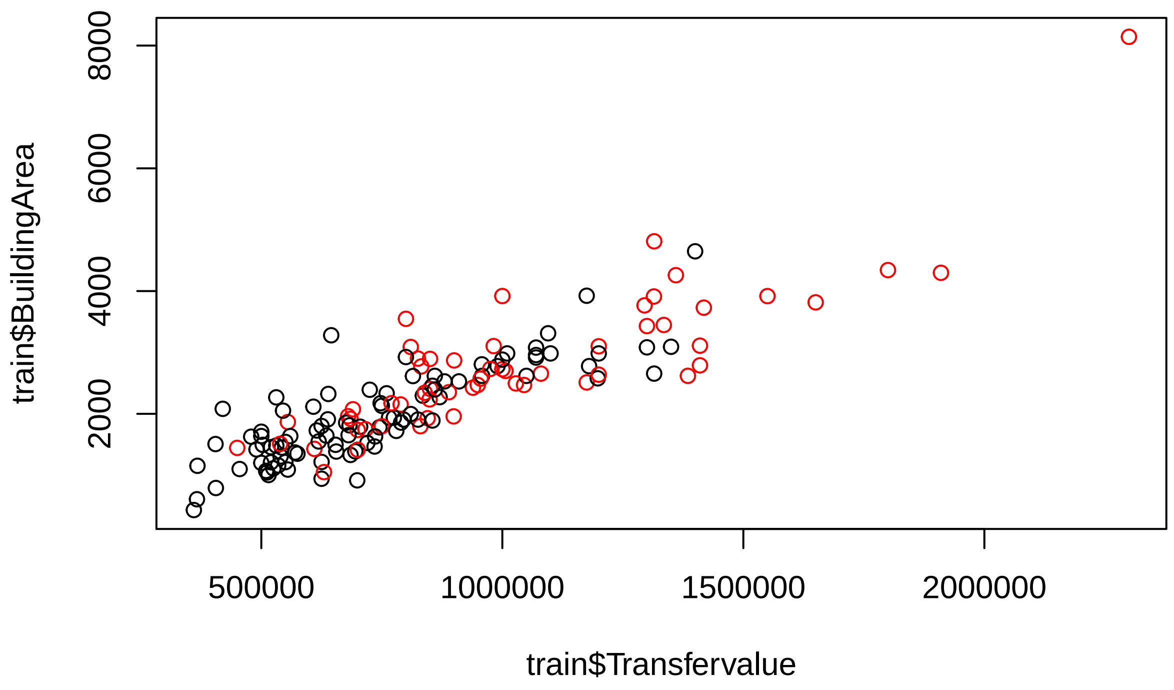
Get dimensions of dataset:

```
dim(GH1)
```

```
## [1] 173  12
```

Run Support Vector Machine classification with three different kernels (linear, polynomial, radial):

```
keeps <- c("Transfervalue", "BuildingArea", "Pool")
GH2=GH1[keeps]
train=GH2[1:150,]
test=GH2[151:173,]
train$Pool[train$Pool == 0] <- -1
test$Pool[test$Pool == 0] <- -1
train$Pool=as.factor(train$Pool);
test$Pool=as.factor(test$Pool);
plot(x=train$Transfervalue, y=train$BuildingArea, col=factor(train$Pool));
```



```r
library(e1071);
#Linear
set.seed (1);
tune.out=tune(svm,Pool~.,data=train,kernel="linear",ranges=list(cost=c(0.001,0.01,0.1,1,5,10,100)));
summary (tune.out);
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.3333333
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.3733333 0.11417985
## 2 1e-02 0.3533333 0.09962894
## 3 1e-01 0.3400000 0.10159226
## 4 1e+00 0.3333333 0.09938080
## 5 5e+00 0.3333333 0.09938080
## 6 1e+01 0.3333333 0.09938080
## 7 1e+02 0.3333333 0.09938080
```

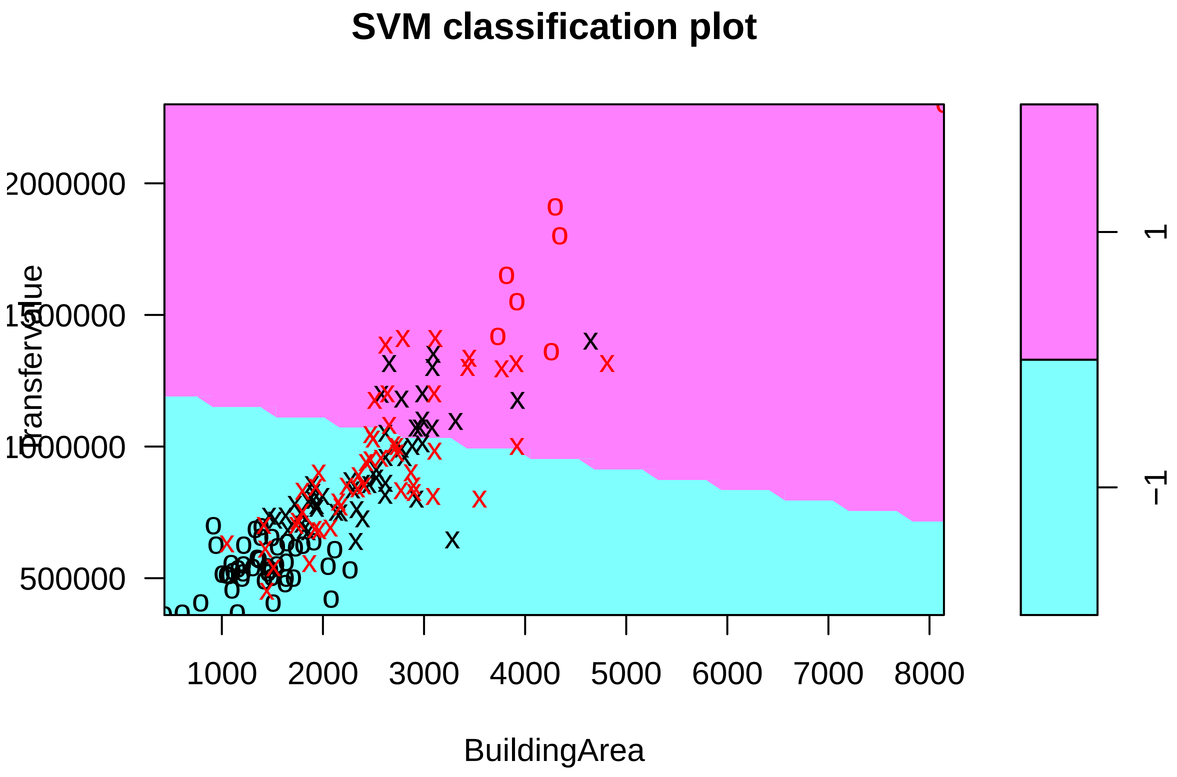
```r
bestmod =tune.out$best.model;
summary(bestmod);
```

```
##
## Call:
## best.tune(method = svm, train.x = Pool ~ ., data = train, ranges = list(cost = c(0.001,
##     0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  1
##       gamma:  0.5
##
## Number of Support Vectors:  99
##
##  ( 50 49 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```r
ypred=predict (bestmod ,test);
table(predict =ypred , truth= test$Pool);
```

```
##         truth
## predict -1  1
##      -1 13  6
##      1   2  2
```

```
plot(bestmod,train)
```



```
#Polynomial
set.seed (1);
tune.out=tune(svm,Pool~.,data=train,kernel="polynomial",ranges=list(cost=c(0.001,0.01,0.1,1,5,10,100)))
summary (tune.out);
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##   0.1
##
## - best performance: 0.2933333
##
## - Detailed performance results:
##     cost     error dispersion
## 1 1e-03 0.3666667 0.10540926
## 2 1e-02 0.3333333 0.10423146
## 3 1e-01 0.2933333 0.11417985
## 4 1e+00 0.3066667 0.09532271
## 5 5e+00 0.3133333 0.09962894
## 6 1e+01 0.3200000 0.11243654
## 7 1e+02 0.3200000 0.11243654
```

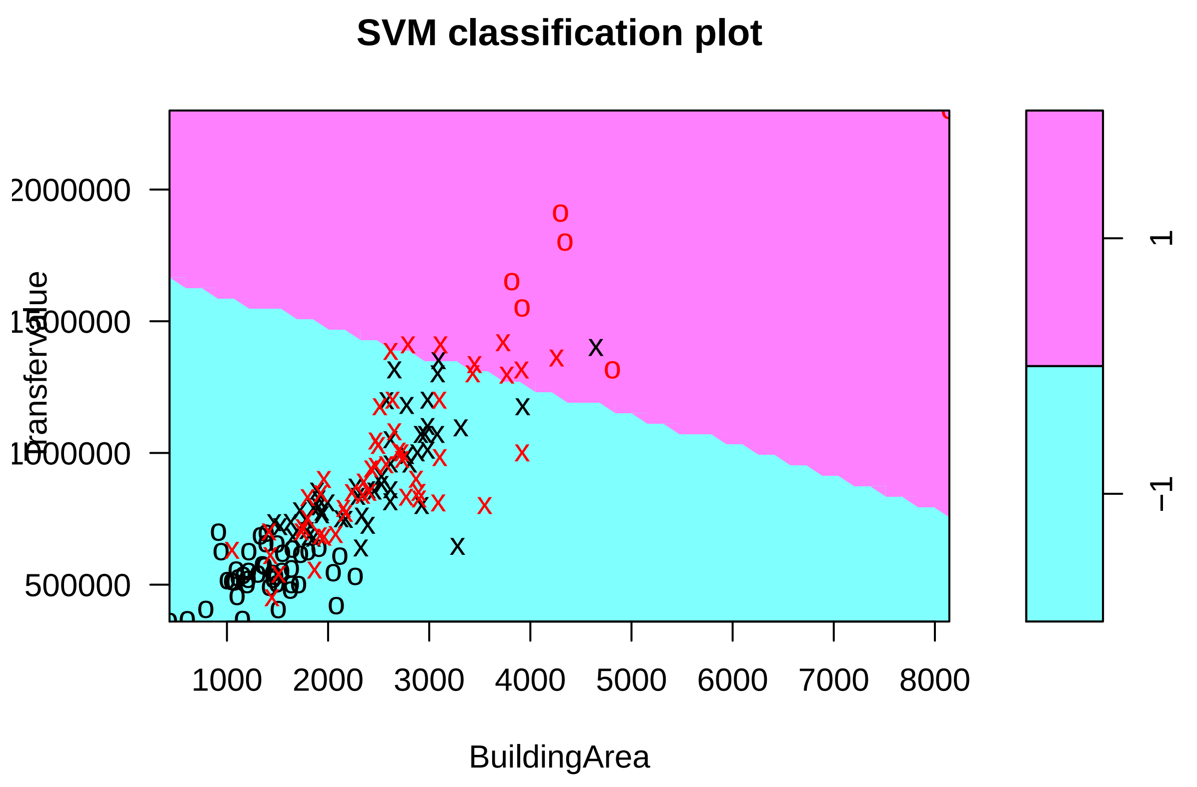
```
bestmod =tune.out$best.model;
summary(bestmod);
```

```
##
## Call:
## best.tune(method = svm, train.x = Pool ~ ., data = train, ranges = list(cost = c(0.001,
##     0.01, 0.1, 1, 5, 10, 100)), kernel = "polynomial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  0.1
##      degree:  3
##       gamma:  0.5
##      coef.0:  0
##
## Number of Support Vectors:  100
##
##  ( 50 50 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```
ypred=predict (bestmod ,test);
table(predict =ypred , truth= test$Pool);
```

```
##        truth
## predict -1  1
##      -1 15  6
##      1   0  2
```

```
plot(bestmod,train)
```



```r
#Radial
set.seed (1);
tune.out=tune(svm,Pool~.,data=train,kernel="radial",ranges
              =list(cost=c(0.001,0.01,0.1,1,5,10,100)));
summary (tune.out);
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     1
##
## - best performance: 0.3466667
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.3733333  0.1141798
## 2 1e-02 0.3733333  0.1141798
## 3 1e-01 0.3800000  0.1090928
## 4 1e+00 0.3466667  0.1079552
## 5 5e+00 0.3533333  0.1177987
## 6 1e+01 0.3733333  0.1003697
## 7 1e+02 0.3533333  0.1297671
```

```r
bestmod =tune.out$best.model;
summary(bestmod);
```

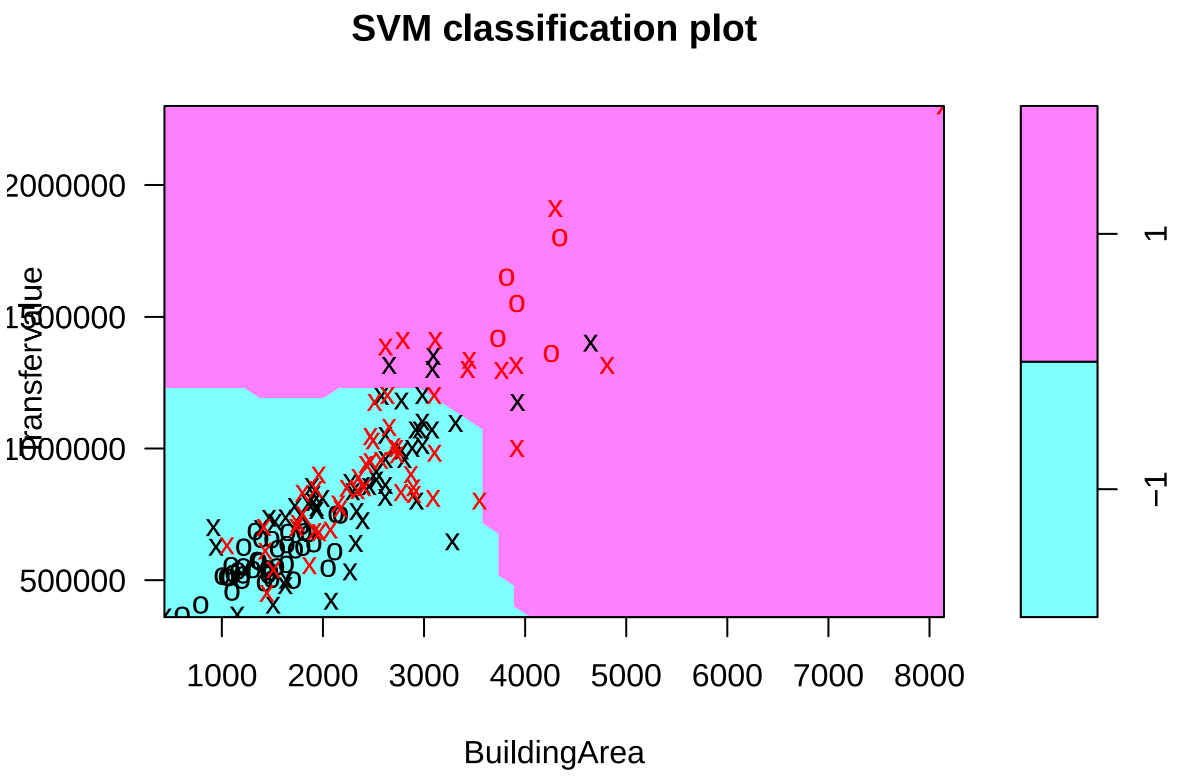
```
##
```

```
## Call:
## best.tune(method = svm, train.x = Pool ~ ., data = train, ranges = list(cost = c(0.001,
##     0.01, 0.1, 1, 5, 10, 100)), kernel = "radial")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##       gamma:  0.5
##
## Number of Support Vectors:  105
##
##  ( 54 51 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```r
ypred=predict (bestmod ,test);
table(predict =ypred , truth= test$Pool);
```

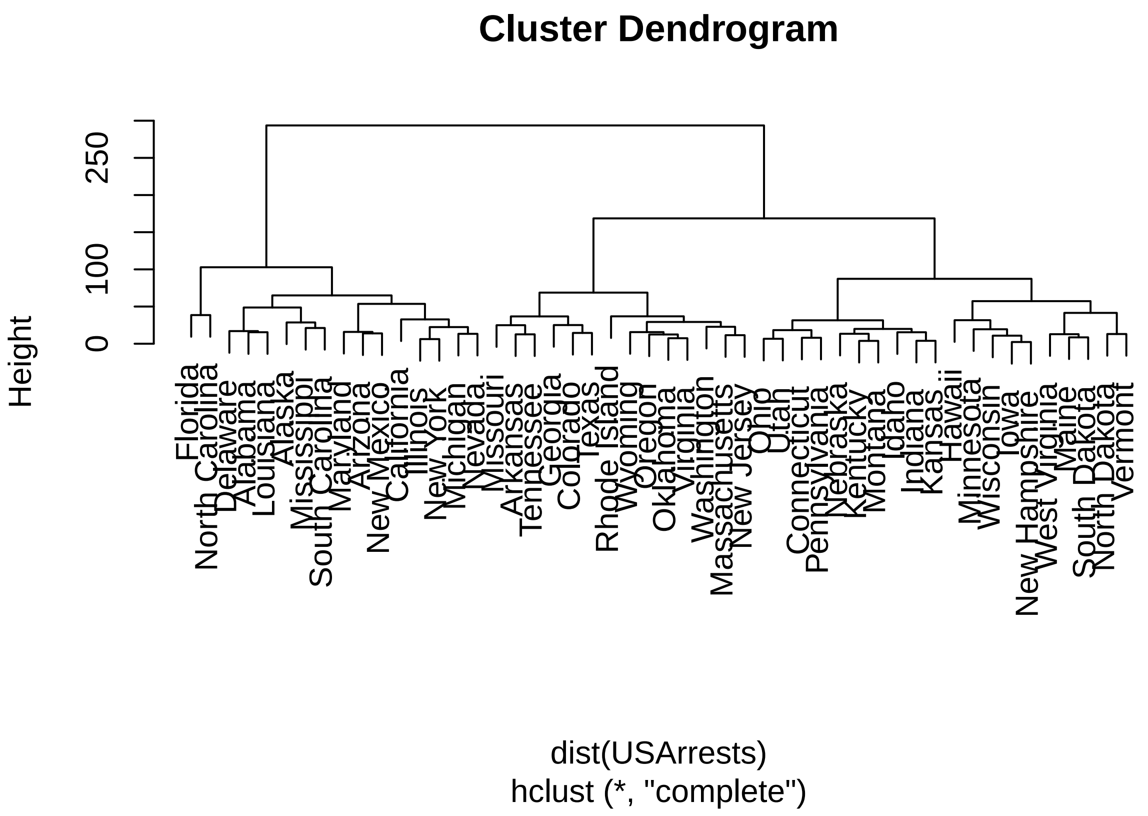
```
##        truth
## predict -1  1
##      -1 15  6
##      1   0  2
```

```r
plot(bestmod,train)
```

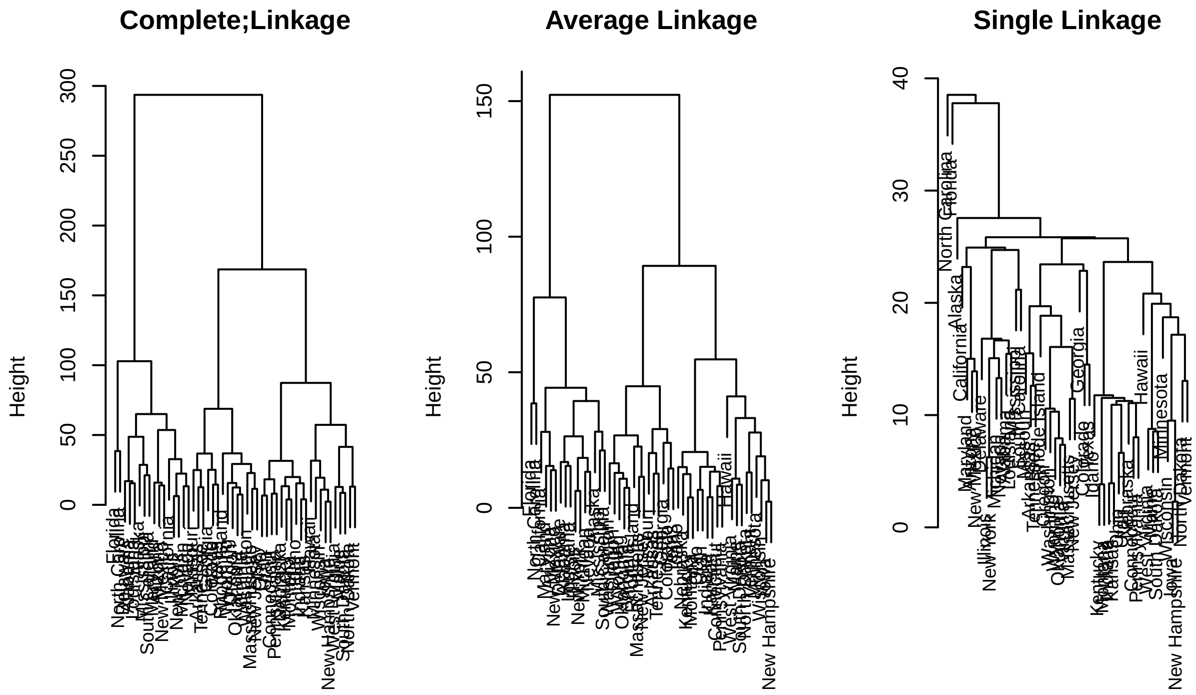


Hierachical Clustering on the USArrests dataset:

```
data(USArrests);
clusters <- hclust(dist(USArrests))
plot(clusters)
```



```
x=USArrests;
hc.complete=hclust(dist(x),method ="complete");
hc.average=hclust(dist(x),method ="average");
hc.single=hclust(dist(x),method ="single");
par(mfrow =c(1,3));
plot(hc.complete,main="Complete;Linkage",xlab="",sub="",cex=.9);
plot(hc.average,main ="Average Linkage",xlab="",sub="",cex=.9);
plot(hc.single,main ="Single Linkage",xlab="",sub="",cex=.9);
```



```
cutree(hc.complete,2);
```

```
##        Alabama         Alaska        Arizona       Arkansas     California
##              1              1              1              2              1
##       Colorado    Connecticut       Delaware        Florida        Georgia
##              2              2              1              1              2
##         Hawaii          Idaho       Illinois        Indiana           Iowa
##              2              2              1              2              2
##         Kansas       Kentucky      Louisiana          Maine       Maryland
##              2              2              1              2              1
##  Massachusetts       Michigan      Minnesota    Mississippi       Missouri
##              2              1              2              1              2
##        Montana       Nebraska         Nevada  New Hampshire     New Jersey
##              2              2              1              2              2
##     New Mexico       New York North Carolina   North Dakota           Ohio
##              1              1              1              2              2
##       Oklahoma         Oregon   Pennsylvania   Rhode Island South Carolina
##              2              2              2              2              1
##   South Dakota      Tennessee          Texas           Utah        Vermont
##              2              2              2              2              2
##       Virginia     Washington  West Virginia      Wisconsin        Wyoming
##              2              2              2              2              2
```

```
xsc=scale (x);
```