Robust Support Vector Machines

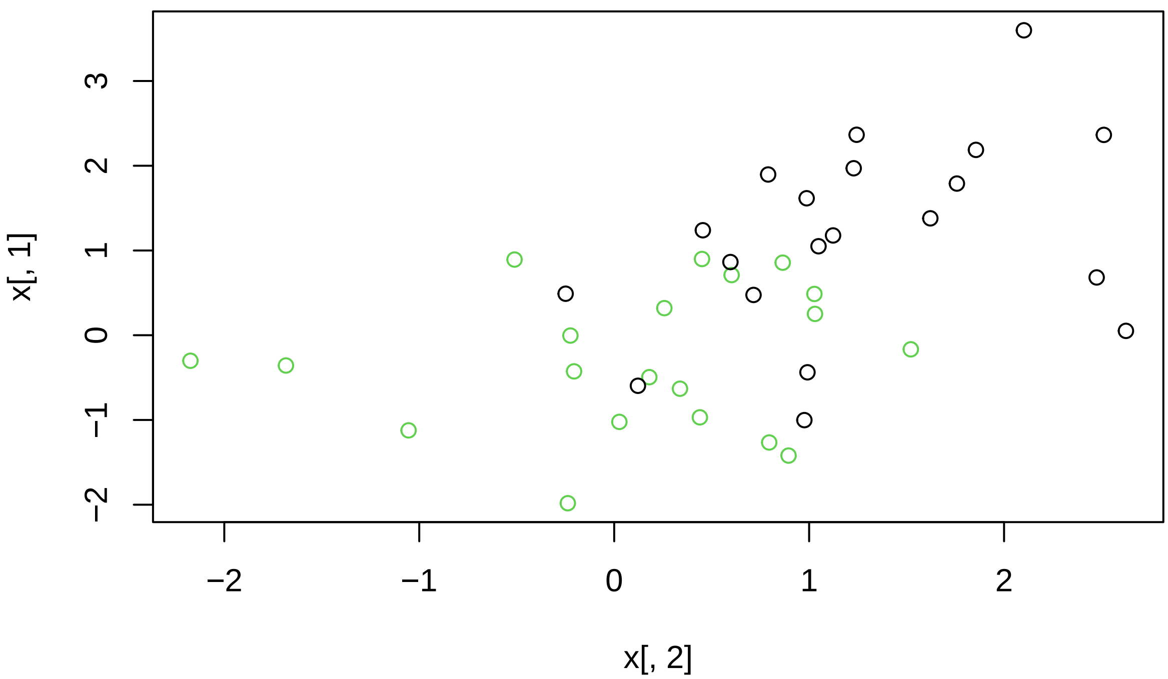
Zhu Wang*

February 14, 2022

The CC-family contains functions of composite of concave and convex functions. The CC-estimators are derived from minimizing loss functions in the CC-family by the iteratively reweighted convex optimization (IRCO), an extension of the iteratively reweighted least squares (IRLS). The IRCO reduces the weight of the observation that leads to a large loss; it also provides weights to help identify outliers. The CC-estimators include robust support vector machines. See Wang (2020).

Support vector machine classification

```
library("mpath")
library("e1071")
set.seed(1900)
x <- matrix(rnorm(40*2), ncol=2)
y <- c(rep(-1, 20), rep(1, 20))
x[y==1,] <- x[y==1, ] + 1
plot(x[,2],x[,1], col=(2-y))
```



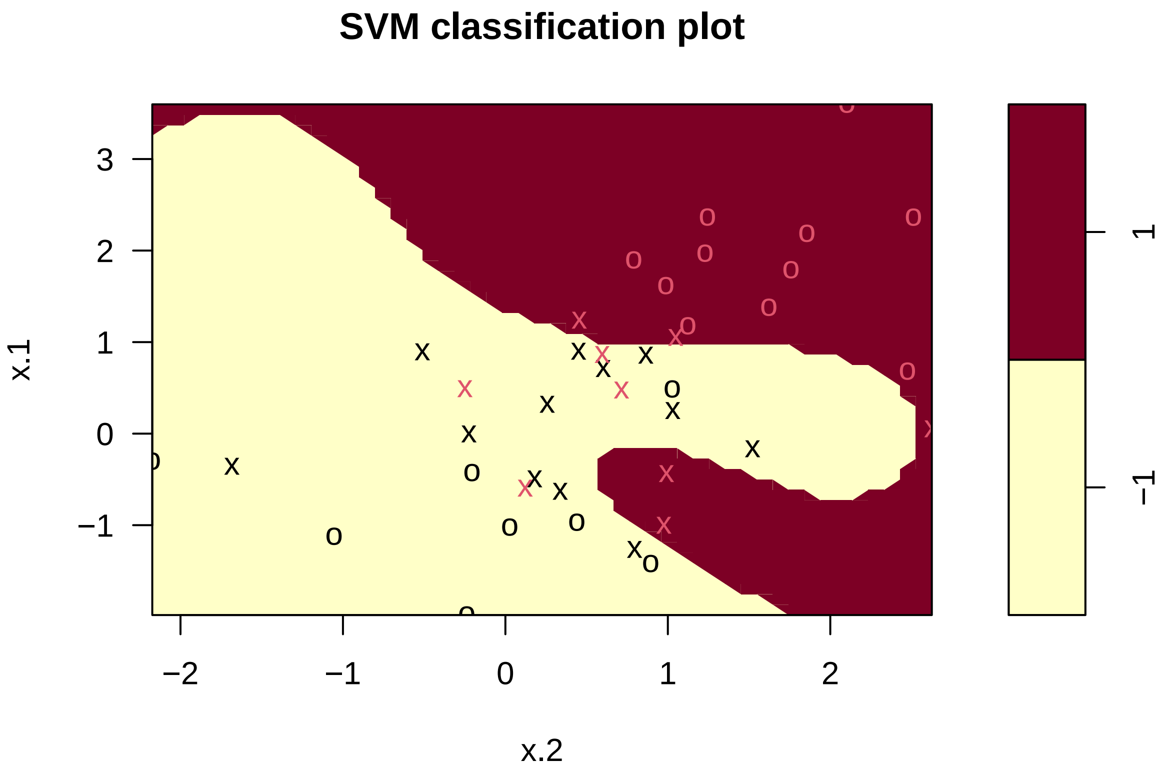
Use the radial kernel SVM for classification.

*UT Health San Antonio, zhuwang@gmail.com

```r
dat <- data.frame(x=x, y=as.factor(y))
svm.model <- svm(y~., data=dat, cost=100, type="C-classification")
summary(svm.model)
```

```
##
## Call:
## svm(formula = y ~ ., data = dat, cost = 100, type = "C-classification")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  21
##
##  ( 12 9 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```r
plot(svm.model, dat)
```

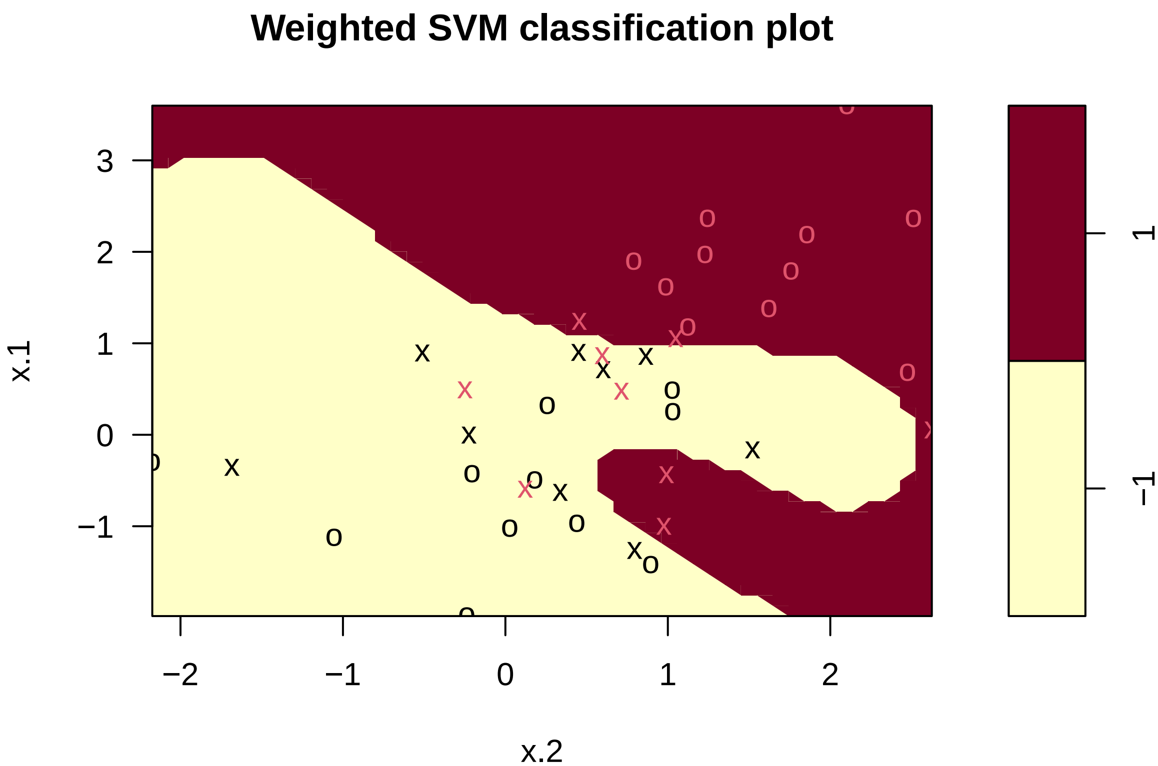


Robust radial kernel SVM for classification.

```r
ccsvm.model <- ccsvm(y ~ ., data = dat, cost = 100, type="C-classification", cfun="acave",
                     s=1)
summary(ccsvm.model)
```

```
##
## Call:
## ccsvm.formula(formula = y ~ ., data = dat, cost = 100, type = "C-classification",
##     cfun = "acave", s = 1)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  100
##
## Number of Support Vectors:  18
##
##  ( 9 9 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```
plot(ccsvm.model, dat)
```



Add 15% outliers to the training data, and fit robust SVM, selecting tuning parameters with the cross-validation method.

```
n <- length(y)
nout <- n*0.15
id <- sample(n)[1:nout]
```

```r
cat("id=", id)
```

```
## id= 16 39 30 17 40 25
```

```r
y[id] <- -y[id]
dat2 <- data.frame(x=x, y=as.factor(y))
ccsvm.opt <- cv.ccsvm(y ~ ., data=dat2, type="C-classification", s=1, cfun="acave",
                      n.cores=2, balance=FALSE)
ccsvm.opt$cost
```

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

```r
ccsvm.opt$gamma
```

```
## [1] 0.125
```

```r
ccsvm.opt$s
```

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

To evaluate prediction, we simulate test data with no outliers.

```r
xtest <- matrix(rnorm(20*2), ncol=2)
ytest <- sample(c(-1,1), 20, rep=TRUE)
xtest[ytest==1, ] <- xtest[ytest==1, ] + 1
testdat <- data.frame(x=xtest, y=as.factor(ytest))
```

Fit a robust SVM model again, with tuning parameters selected by cross-validation, then evaluate prediction accuracy with test data, with 85% accuracy.

```r
ccsvm.model1 <- ccsvm(y ~ ., data = dat2, cost = ccsvm.opt$cost, gamma=ccsvm.opt$gamma,
                      s=ccsvm.opt$s, cfun="acave", type="C-classification")
summary(ccsvm.model1)
```

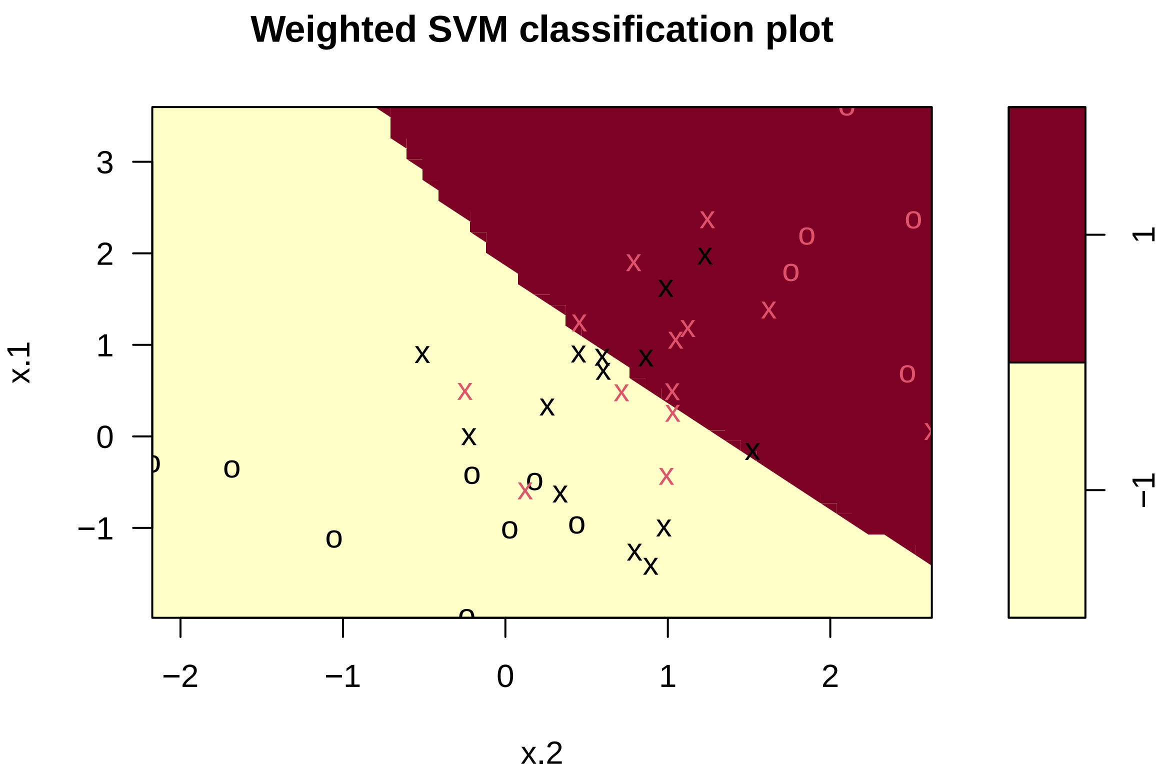
```
##
## Call:
## ccsvm.formula(formula = y ~ ., data = dat2, cost = ccsvm.opt$cost,
##     gamma = ccsvm.opt$gamma, s = ccsvm.opt$s, cfun = "acave", type = "C-classification")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  27
##
##  ( 14 13 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```r
table(predict=predict(ccsvm.model1, xtest), truth=testdat$y)
```

```
##        truth
## predict -1  1
```

```
##      -1  7  2
##      1   1 10
```

```r
plot(ccsvm.model1, dat2)
```



Develop a SVM model with training data and evaluate with the test data. The prediction accuracy is 80%.

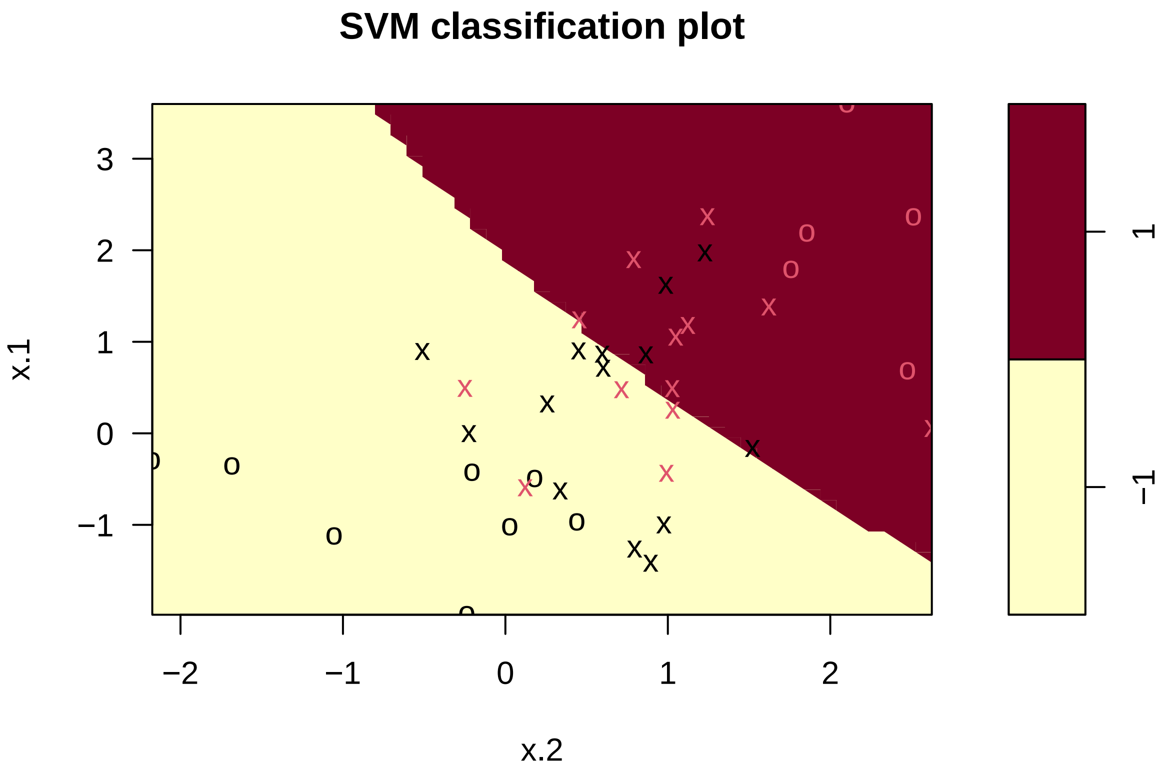
```r
svm.model1 <- svm(y~., data=dat2, cost=ccsvm.opt$cost, gamma=ccsvm.opt$gamma,
                  type="C-classification")
summary(svm.model1)
```

```
##
## Call:
## svm(formula = y ~ ., data = dat2, cost = ccsvm.opt$cost, gamma = ccsvm.opt$gamma,
##     type = "C-classification")
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  27
##
##  ( 14 13 )
##
##
## Number of Classes:  2
##
## Levels:
##  -1 1
```

```r
table(predict=predict(svm.model1, testdat), truth=testdat$y)
```

```
##         truth
## predict -1 1
##      -1  7 3
##      1   1 9
```

```r
plot(svm.model1, dat2)
```



In robust SVM with function `ccsvm`, argument `cfun` can be chosen from `"hcave"`, `"acave"`, `"bcave"`, `"ccave"`, `"dcave"`, `"gcave"`, `"tcave"`, `"ecave"`, for a variety of concave functions.

Support vector machine regression

We predict median value of owner-occupied homes in suburbs of Boston. The data can be obtained from the UCI machine learning data repository. There are 506 observations and 13 predictors.

```r
urlname <- "https://archive.ics.uci.edu/ml/"
filename <- "machine-learning-databases/housing/housing.data"
dat <- read.table(paste0(urlname, filename), sep="", header=FALSE)
n <- dim(dat)[1]
p <- dim(dat)[2]
cat("n=",n,"p=", p, "\n")
```

```
## n= 506 p= 14
```

Randomly split the data into 90% of samples for training and 10% of samples as test data.

```r
set.seed(129)
trid <- sample(n)[1:(n*0.9)]
```

```r
traindat <- dat[trid, ]
testdat <- dat[-trid, ]
```

Fit the robust radial kernel CCSVM model with truncated ϵ-insensitive loss, i.e., `cfun="tcave"` in function `ccsvm`. Root mean squared error on test data is reported. A comprehensive robust CCSVM analysis with other types of `cfun` can be found in Wang (2020).

```r
ccsvm.model <- ccsvm(x=traindat[,-p], y=traindat[,p], cost = 2^3, gamma=2^(-4),
                     epsilon=2^(-4), s=5, cfun="tcave")
summary(ccsvm.model)
```

```
##
## Call:
## ccsvm.matrix(x = x, y = ..1, cost = ..2, gamma = ..3, epsilon = ..4,
##     s = 5, cfun = "tcave")
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  8
##       gamma:  0.0625
##     epsilon:  0.0625
##
##
## Number of Support Vectors:  351
```

```r
ccsvm.predict <- predict(ccsvm.model, testdat[,-p])
mse1 <- mean((testdat[,p] - ccsvm.predict)^2)
cat("RMSE with robust SVM", sqrt(mse1))
```

```
## RMSE with robust SVM 2.758136
```

Fit the radial kernel SVM model. The RMSE is larger than the robust SVM, and the model has a larger number of support vectors as well. See the figure below for a comparison.

```r
svm.model <- svm(x=traindat[,-p], y=traindat[,p], cost=2^3, gamma=2^(-4), epsilon=2^(-4))
summary(svm.model)
```

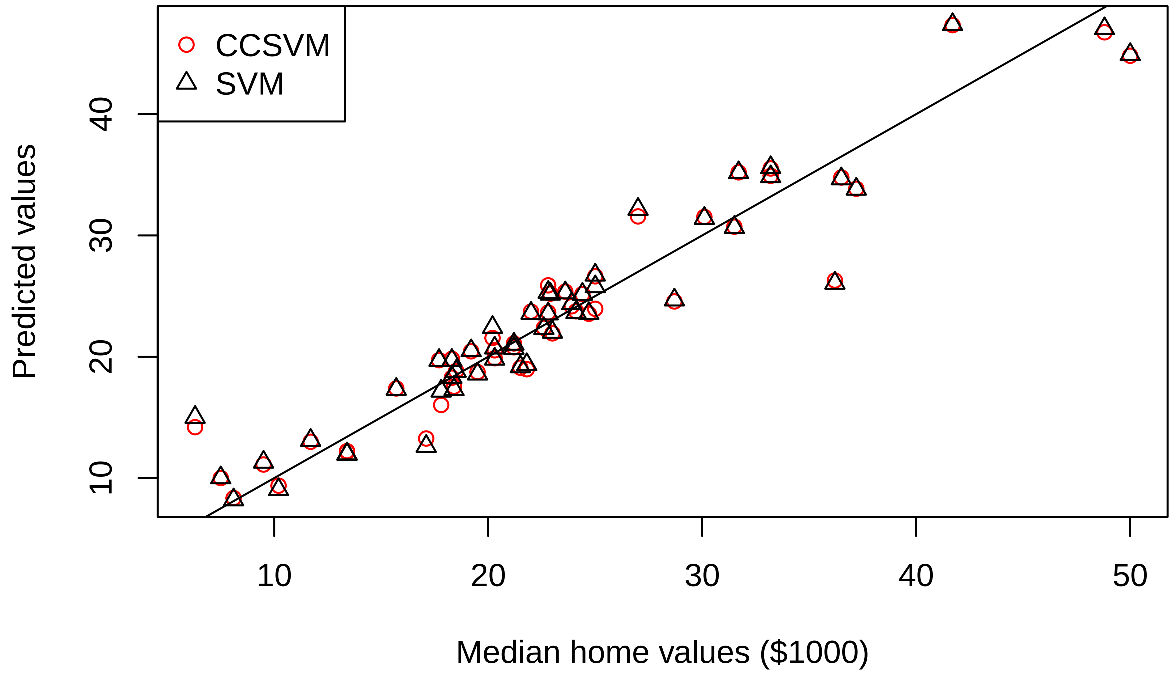
```
##
## Call:
## svm.default(x = traindat[, -p], y = traindat[, p], gamma = 2^(-4),
##     cost = 2^3, epsilon = 2^(-4))
##
##
## Parameters:
##    SVM-Type:  eps-regression
##  SVM-Kernel:  radial
##        cost:  8
##       gamma:  0.0625
##     epsilon:  0.0625
##
##
## Number of Support Vectors:  361
```

```r
svm.predict <- predict(svm.model, testdat[,-p])
mse2 <- mean((testdat[,p] - svm.predict)^2)
```

```r
cat("RMSE with SVM", sqrt(mse2))
```

```
## RMSE with SVM 2.840434
```

```r
plot(testdat[,p], ccsvm.predict, col="red", pch=1, ylab="Predicted values",
     xlab="Median home values ($1000)")
points(testdat[,p], svm.predict, col="black", pch=2)
legend("topleft", c("CCSVM", "SVM"), col=c("red", "black"), pch=c(1, 2))
abline(coef=c(0, 1))
```



Reference

Wang, Zhu. 2020. "Unified Robust Estimation." *arXiv E-Prints*, October, arXiv:2010.02848. http://arxiv.org/abs/2010.02848.