CUNY School of Professional Studies

SPS.CUNY.EDU

Lecture 10
2020 Spring Data-622
Support Vector Machine SVM

Raman Kannan

Instructor Email Address: Raman.Kannan@sps.cuny.edu

Acknowledgements:
Generous support from IBM Power Systems Academic Initiative
IBM PSAI provides computing infrastructure for free

Review

Our focus has been classification.
Using Bayesian Formulation
- We saw linear **parametric** classifiers
  - Discriminative classifier
    - Conditional Likelihood P(class|data)
    - Logistic Regression
  - Generative classifier
    - Joint Distribution – P(data,class)
    - Fisher Discriminant *LDA/QDA*
    - Naive Bayes
- Instance based – **non-parametric**, any kind of separation linear/non-linear
  - KNN – using euclidean distance metric
- In this module we will look at SVM
  - Another Linear discriminative Classifier
  - Latest – new type of learning –
  - Vapnik-Chervonenkis (VC) Dimension

Script for Algorithms

Develop the Intuition
Understand the assumptions
Develop the mathematics
Run the algorithms
Learn to interpret the result/output
Predict using the model
Learn to determine the performance
Distinguish training/testing error
Differentiate between overfitting/underfitting
Techniques to improve performance

Script remains the same even though mechanism is different

Intuition (VC Dimension)

2.2 *Vapnik-Chervonenkis (VC) Dimension* ALP 27

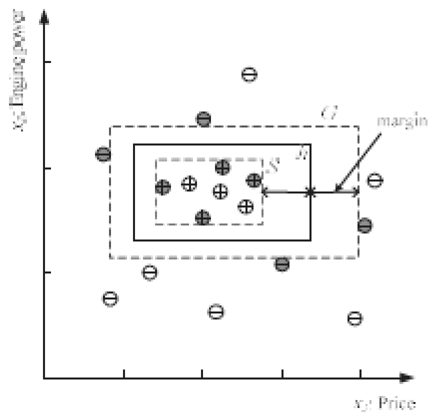


Figure 2.5 We choose the hypothesis with the largest margin, for best separation. The shaded instances are those that define (or support) the margin; other instances can be removed without affecting h.

2.2 Vapnik-Chervonenkis (VC) Dimension

Let us say we have a dataset containing N points. These N points can be labeled in 2^N ways as positive and negative. Therefore, 2^N different learning problems can be defined by N data points. If for any of these problems, we can find a hypothesis $h \in \mathcal{H}$ that separates the positive examples from the negative, then we say $\mathcal{H}$ *shatters* N points. That is, any learning problem definable by N examples can be learned with no error by a hypothesis drawn from $\mathcal{H}$. The maximum number of points that can be shattered by $\mathcal{H}$ is called the *Vapnik-Chervonenkis* (VC) *dimension* of $\mathcal{H}$, is denoted as $VC(\mathcal{H})$, and measures the *capacity* of $\mathcal{H}$.

VC DIMENSION

If there are N points, and with two possible labels, then there are 2^N possible Learners/algorithms – not all of which may be useful.
So what is:VC dim = the maximum number of points that can be separated in all possible ways by that set of functions.

Class Separation

When we compute NaiveBayes, we simply compared the probabilities and took the highest probability – we stopped short of computing exact probabilities. P(c1|x)> P(c2|x) or the ratio P(C1|x)/P(c2|x) > 1.

Similarly,in SVM, all we need is the boundary separating the two classes.

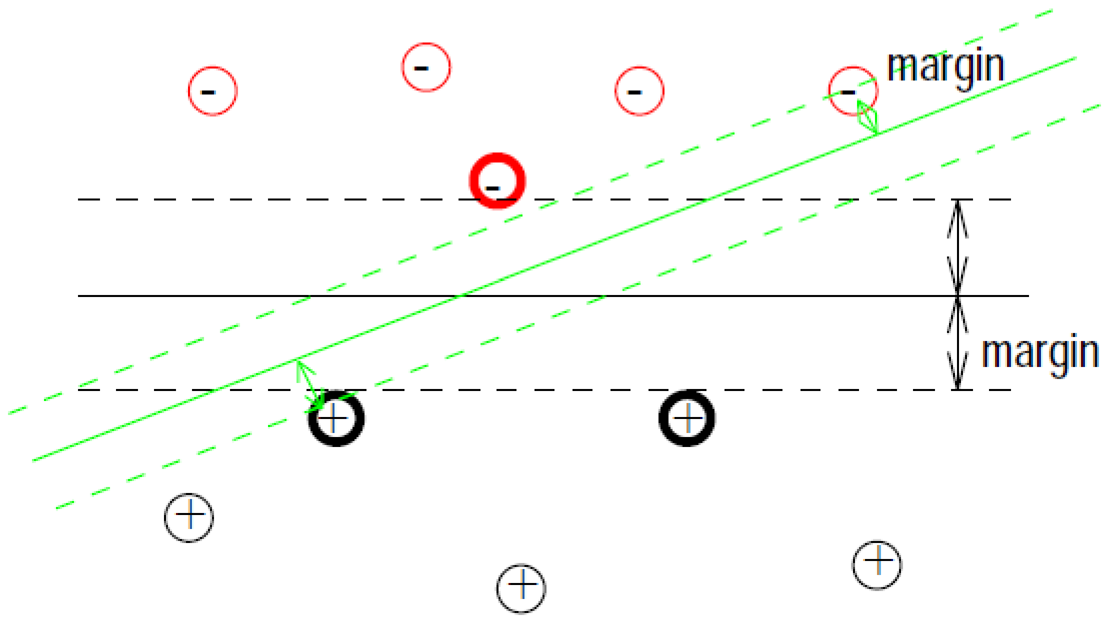
Let us say there are two classes C1 and C2 and what is the probability on that boundary?

1. It is a discriminant-based method and uses Vapnik's principle to never solve a more complex problem as a first step before the actual problem (Vapnik 1995). For example, in classification, when the task is to learn the discriminant, it is not necessary to estimate where the class densities $p(\boldsymbol{x}|C_i)$ or the exact posterior probability values $P(C_i|\boldsymbol{x})$; we only need to estimate where the class boundaries lie, that is, $\boldsymbol{x}$ where $P(C_i|\boldsymbol{x}) = P(C_j|\boldsymbol{x})$. Similarly, for outlier detection, we do not need to estimate the full density $p(\boldsymbol{x})$; we only need to find the boundary separating those $\boldsymbol{x}$ that have low $p(\boldsymbol{x})$, that is, $\boldsymbol{x}$ where $p(\boldsymbol{x}) < \theta$, for some threshold $\theta \in (0, 1)$.

SVM Basics:margin/vector

3 Support Vector Machines: basics

[Boser, Guyon, Vapnik '92],[Cortes & Vapnik '95]



Nice properties: convex, theoretically motivated, nonlinear with kernels..

http://www.cs.columbia.edu/~kathy/cs4701/documents/jason_svm_tutorial.pdf

SVM Basics:margin/vector

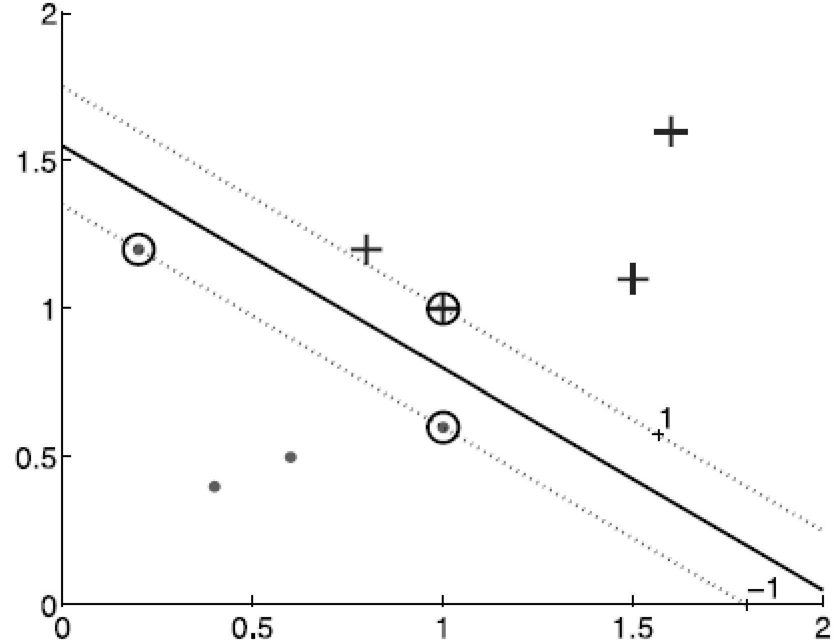


Figure 13.1 For a two-class problem where the instances of the classes are shown by plus signs and dots, the thick line is the boundary and the dashed lines define the margins on either side. Circled instances are the support vectors.

ALP: 13th Chapter

SVM: default kernel

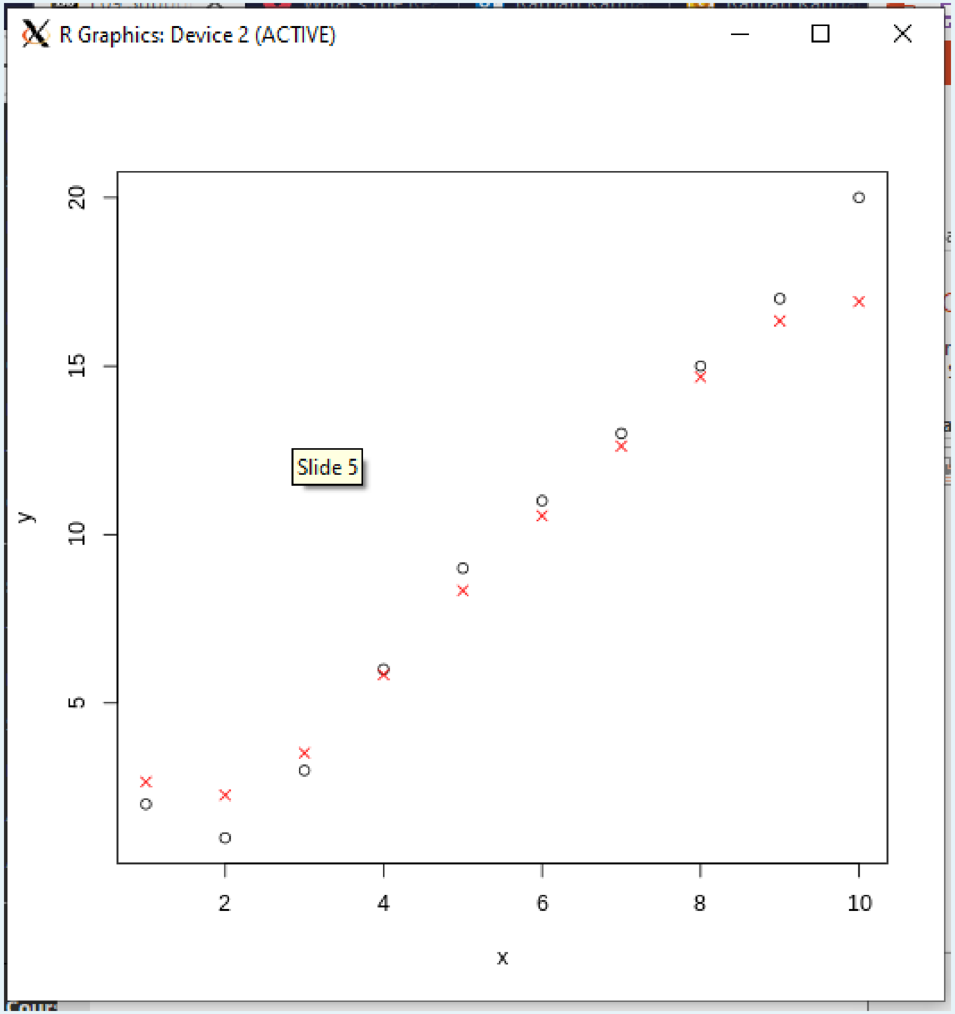
path<-"download ddata.txt into your local dir and change this path"
dataxy<-read.csv(paste(path,"/","ddata.txt",sep=""),head=TRUE,sep=",")
model<-svm(y~x,dataxy)
dataxy$predicted<-predict(model,dataxy)
plot(dataxy[c(1,2)])
points(dataxy$x,dataxy$predicted,col="red",pch=4)

```
> dataxy[,c(1,2)]
    x  y
1   1  2
2   2  1
3   3  3
4   4  6
5   5  9
6   6 11
7   7 13
8   8 15
9   9 17
10 10 20
```

```
> summary(ddata.svm.model)

Call:
svm(formula = y ~ x, data = dataxy)


Parameters:
   SVM-Type:  eps-regression
 SVM-Kernel:  radial
       cost:  1
      gamma:  1
    epsilon:  0.1


Number of Support Vectors:  5
```

```
> dataxy
    x  y predicted
1   1  2  2.659755
2   2  1  2.268953
3   3  3  3.517689
4   4  6  5.840469
5   5  9  8.339256
6   6 11 10.558571
7   7 13 12.623260
8   8 15 14.674331
9   9 17 16.339257
10 10 20 16.910680
```

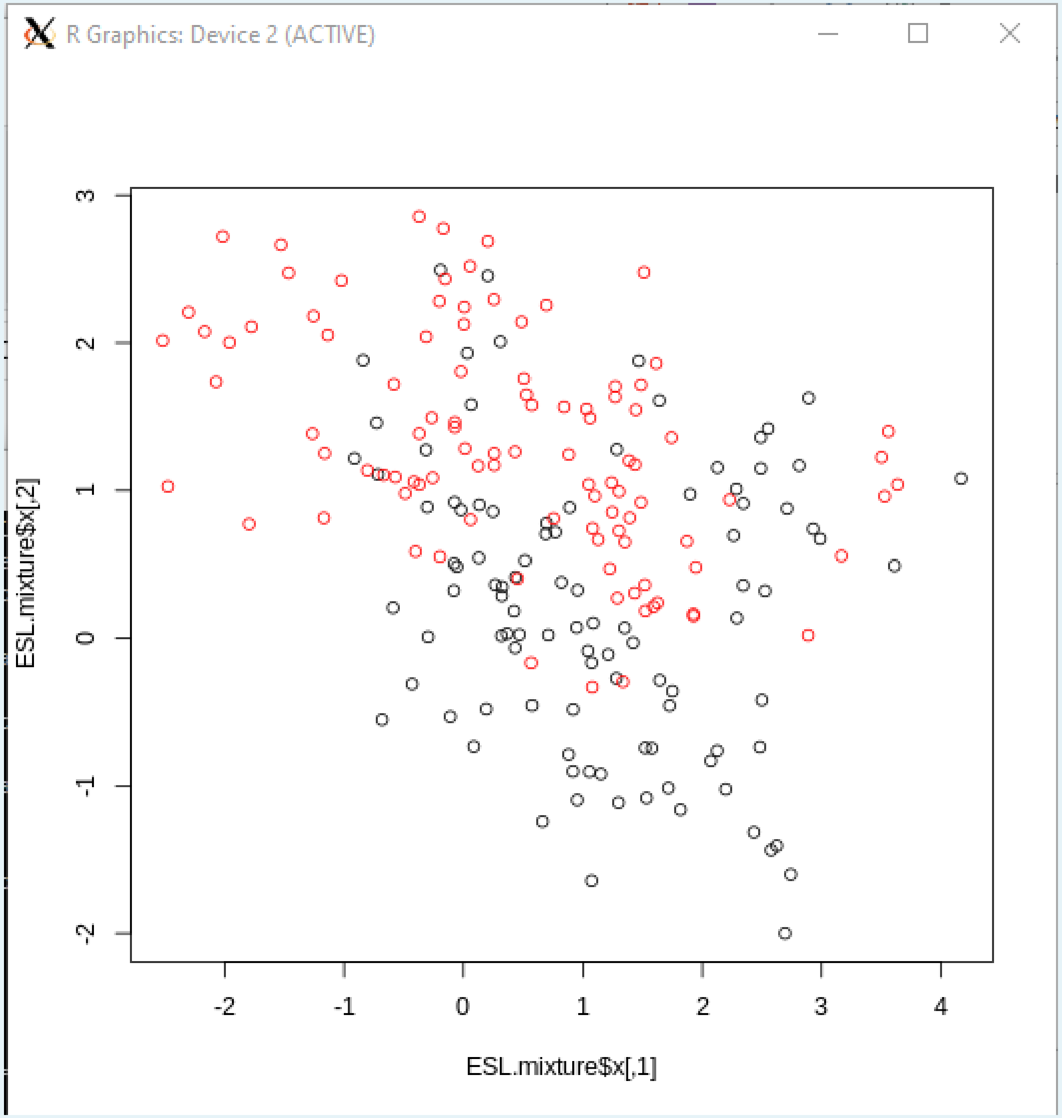
ESL: default Kernel



ESL: Radial Basis Kernel

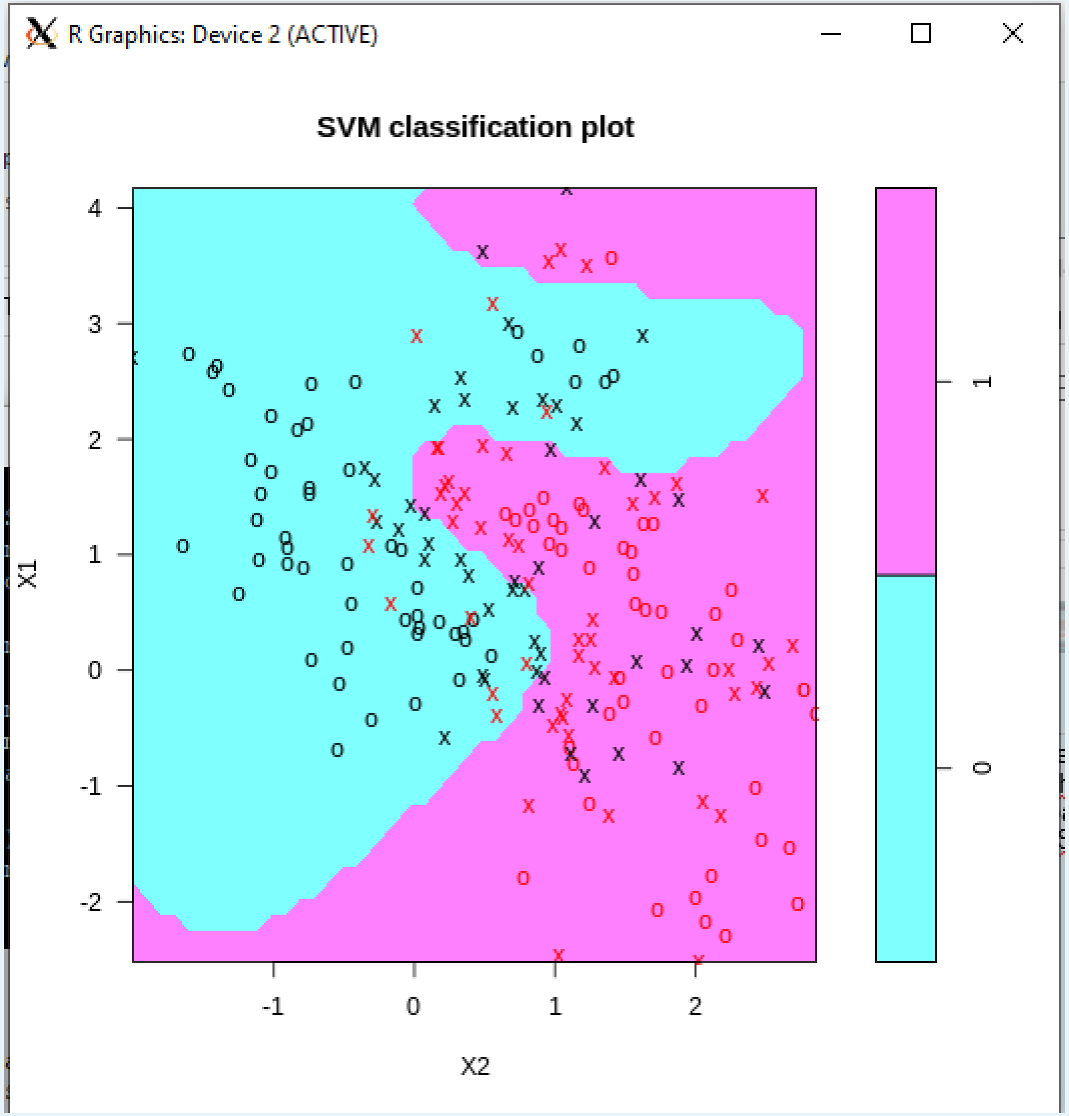
```
require(e1071)
eslpath<-'~/ML/ESL.SVM/‘
load(paste(eslpath,"ESL.mixture.rda",sep=""))
plot(ESL.mixture$x,col=ESL.mixture$y+1)
data_svm<-data.frame(y=factor(ESL.mixture$y), ESL.mixture$x)
data.svm<-svm(factor(y)~.,data=data_svm,scale=FALSE,kernel="radial",cost=5)
plot(data.svm,data_svm)
```

EDA



R Graphics: Device 2 (ACTIVE)
ESL.mixture$x[,1]
ESL.mixture$x[,2]
-2
-1
0
1
2
3
4

SVM

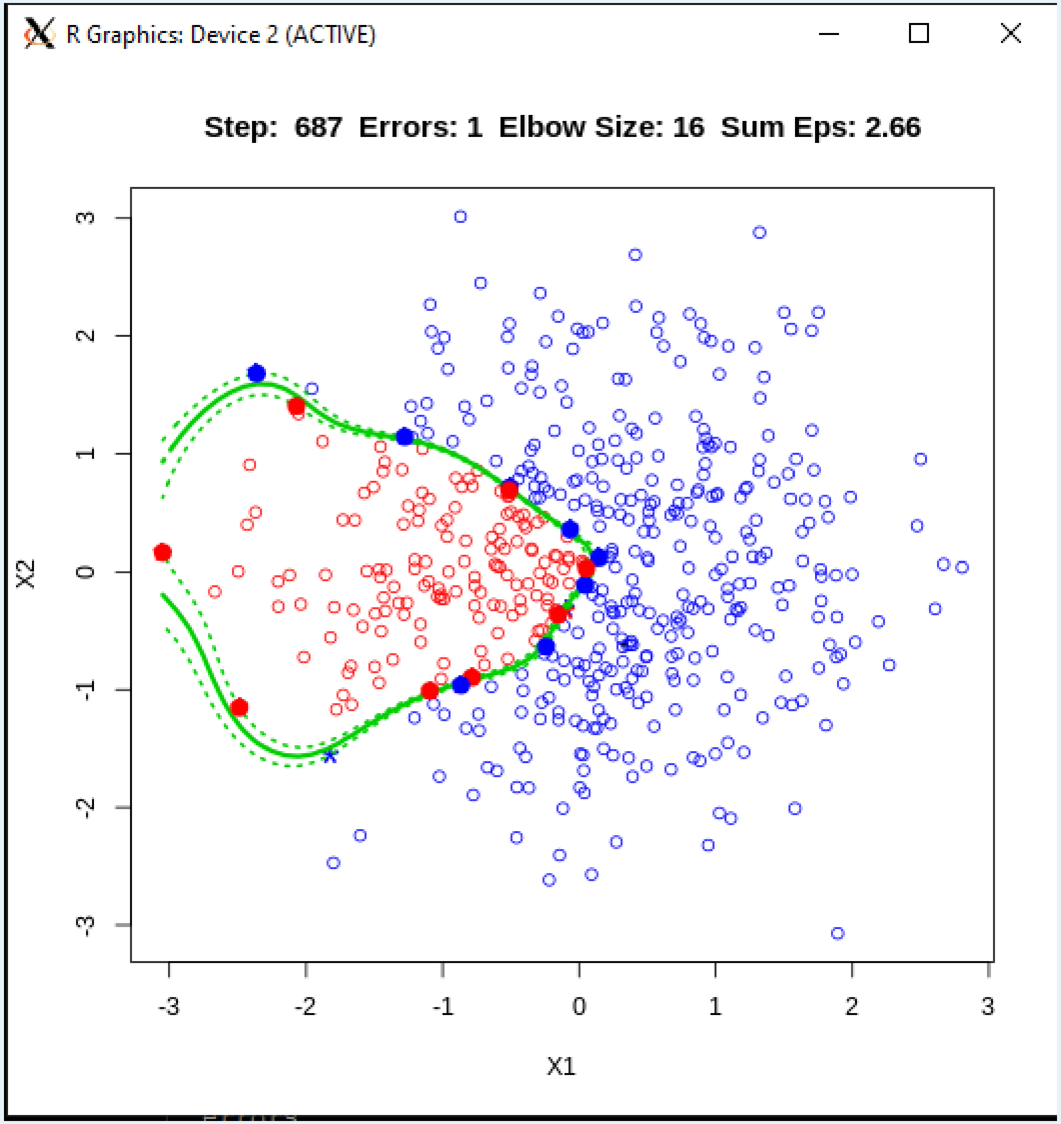


R Graphics: Device 2 (ACTIVE)
SVM classification plot
X1
X2
4
3
2
1
0
-1
-2
-1
0
1
2

How about this ?

SVM can come up with such boundaries by default.

We will see this example two/three weeks from now…



Lab Segment

Please review the document titled
script concepts for svm with math

You can also experiment with the svmpath R script.

References

http://research.microsoft.com/en-us/um/people/cburges/papers/svmtutorial.pdf
http://www.cs.columbia.edu/~kathy/cs4701/documents/jason_svm_tutorial.pdf
13th Chapter from Alpaydin book.