Lecture 8: Classifiers

Intro to Data Science for Public Policy, Spring 2016

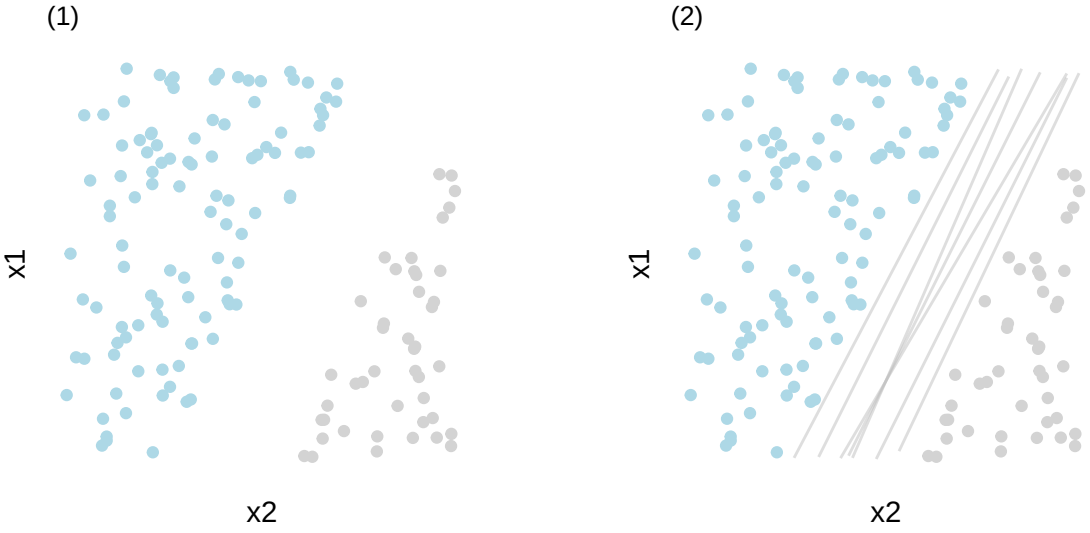
by Jeff Chen & Dan Hammer, Georgetown University McCourt School of Public Policy

Contents

Building on Lecture 7, this section lightly introduces three more classification algorithms: Generalized Linear Models, Support Vector Machines, and Artificial Neural Networks.

Support vector machines

SVMs cannot be interpretted. They are
You're given a three feature data set. Two features are continuous inputs and the third is a target containing labels of two groups. Upon plotting the points in two dimensional space and color coding for the target, you notice that there is a clear line of separation: A straight line can partition one group from the other (Figure 1). You think and realize that multiple lines could do the job: there are almost infinite lines (Figure 2) that could serve as the boundary between the groups. But which is the best? There should in theory be one line that optimally describes the separation between the groups.



If we are to assume a straight line is appropriate, we can find a line that maximizes the distance between the groups. As seen in Figure 3, the dashed grey lines and the solid purple lines are known as hyperplanes, but are simply lines in two dimensional space. H1 and H2 are hyperplanes that are defined by a set of "support vectors" – points that serve as control or reference points for the location of the hyperplane (see Figure 4). The elegance of this method is that not all points in a dataset are used to define H1 and H2: only select points on or near the hyperplanes are required to define the plane. These planes are defined using simple

linear equations shown in dot-product form:

$$w^T x - b = +1$$

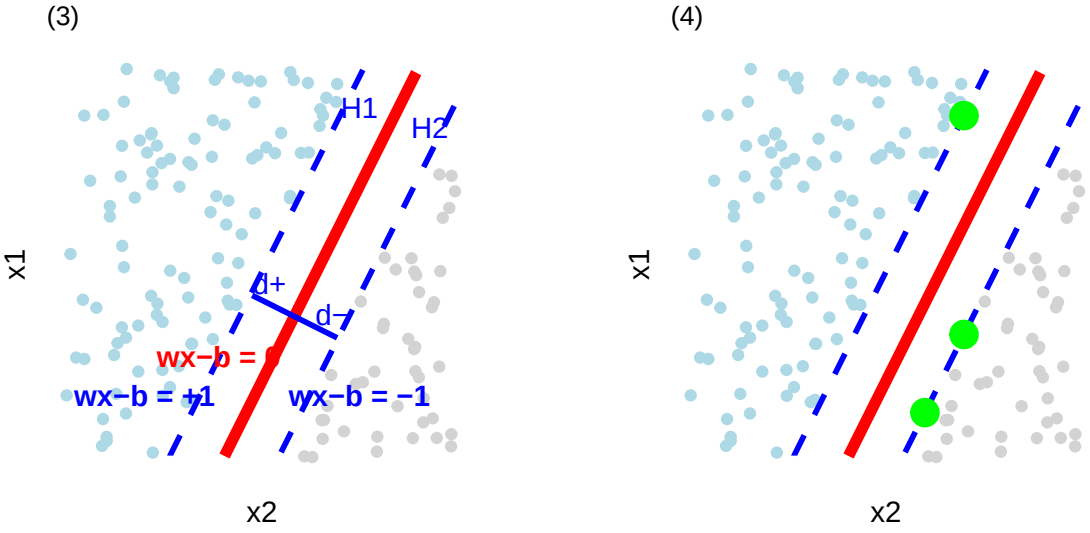
and

$$w^T x - b = -1$$

for H2, where w is a weight that needs to be calibrated. H1 and H2 primarily serve as the boundaries of what is known as the *margin*, or the space that maximally separates the two classes that are linearly separable The optimal hyperplane or *decision boundary* is defined as

$$w^T x - b = 0$$

and sits at a distance of $d+$ from H1 and $d-$ from H2.



[Soft margin versus hard margin]

[Higher dimensions]

[Pseudo code]

[Math]

[Cost and Gamma]

Start with line $mx + b = y$

Generalized case for plane $wt + b = y$

Classification for two classes $w^T + b > y$ $w^T + b < y$ $w^T + b$ are parameters of plane

Where $y = 0$, that's when we have no idea if point is positive or neg labels are $y \in -1, +1$

boundaries are thus -1 and 1. Above the absolutes are scored as 1 and everything in between is iffy $w^T + b = +1$ $w^T + b = 0$ $w^T + b = -1$ Lines are parallel to one another

To optimize: we want to find the maximum distance between H1 and H2. This can be done by finding the distance of the line that is perpendicular to H1 and H2 since they are parallel. The following equations are the points at which the perpendicular line intersects at two points $w_1^T + b = 1$ $w_2^T + b = -1$

Subtract the two equations: $w^T(x_1 - x_2) = 2$ Using a vector calculus trick, we can $x_1 - x_2 = \frac{2}{||w||}$ where $||w||$ is the normalized w vector (length of w) This is the definition of the margin $= \frac{2}{||w||}$

Optimization - maximize $= \frac{2}{||w||}$ - easier to minimize $\frac{1}{2}||w||^2$ — this is known as a quadratic programming problem Or maximize another equation: sum of all points i, minus product alphas, labels, values $w(\alpha) =$

$\sum_i \alpha_i - \frac{1}{2} \sum_i \alpha_1 \alpha_0 y_1 y_0 x_1^T x_0$ subject to $\alpha_i \geq 0$ (non-negatives), $\sum_i \alpha_i y_i = 0$ (sum of alpha and y are equal to zero)

Once the equation is maximized, $w = \sum_i \alpha_i y_i x_i$ can be extracted Many α_i are zero, non-zeros are the support vectors or basically data points Only a few X's matter

$x_i^T x_j$ is the dot product that indicates relationship and only cases that move together are used

Applying SVMs

```
library(e1071)
health <- read.csv("data/lecture8.csv")

#Create index of randomized booleans of the same length as the health data set
  set.seed(100)
  rand <- runif(nrow(health))
  rand <- rand > 0.5

#Create train test sets
  train <- health[rand == T, ]
  test <- health[rand == F, ]

#Check SVM
  svm.fit <- svm(coverage ~ ., data=train)
  summary(svm.fit)

#Tune SVM
  tune <- tune.svm(coverage ~. ,
                   data = train,
                   kernel="radial",
                   cost=10^(-1:2), gamma=c(.5,1,2))

  print(tune)

#Re-train
  svm.fit2 <- svm(coverage ~ ., data=train, kernel="polynomial", cost=1, gamma=1)
  summary(svm.fit2)

#Prediction
  svm.pred <- predict(svm.fit2, test)
```

Estimate the AUC

```
#plotROC
  library(plotROC)
  library(ggplot2)

#Predict values for train set
  pred.svm.train <- predict(svm.fit2, train, type='prob')

#Predict values for test set
  pred.svm.test <- predict(svm.fit2, test, type='prob')


#Set up ROC inputs
```

```
  input.svm <- rbind(data.frame(model = "train", d = train$coverage, m = pred.svm.train),
                     data.frame(model = "test", d = test$coverage,  m = pred.svm.test))

#Graph all three ROCs
  roc.svm <- ggplot(input.svm, aes(d = d, model = model, m = m, colour = model)) +
             geom_roc(show.legend = TRUE) + style_roc()  + ggtitle("ROC: SVM")

#AUC
  calc_auc(roc.svm)
```

Artificial Neural Networks

Applications of classifiers

Appropriate uses of classification techniques

[text goes here]

```
#
```

Scoring

[text goes here]

```
#
```

prediction and prioritization

[text goes here]

```
#
```

Propensity score matching

[text goes here]

```
#
```

Exercise Data

- [Labor and wage analysis]