Support Vector Machines in R

Contents

Introduction to Support Vector Machines

- Support Vector Machines (SVM's) are a relatively new learning method generally used for classification problem.
- Although the first paper dates way back to early 1960's it is only in 1992-1995 that this powerful method was universally adopted as a mainstream machine learning paradigm

The basic idea is to find a hyper plane which separates the d-dimensional data perfectly into its classes. However, since training data is often not linearly separable, SVM's introduce the notion of a "Kernel-induced Feature Space" which casts the data into a higher dimensional space where the data is separable.

What is a Hyper Plane

In two dimensions, a hyper plane is defined by the equation:

$$\boxed{W_1X_1 + W_2X_2 + b = 0}$$

This is nothing but **equation of line.**

The above equation can be easily extended to the p-dimensional setting:

$$\boxed{W_1X_1 + W_2X_2 + \cdots + W_pX_p + b = 0}$$

In short,

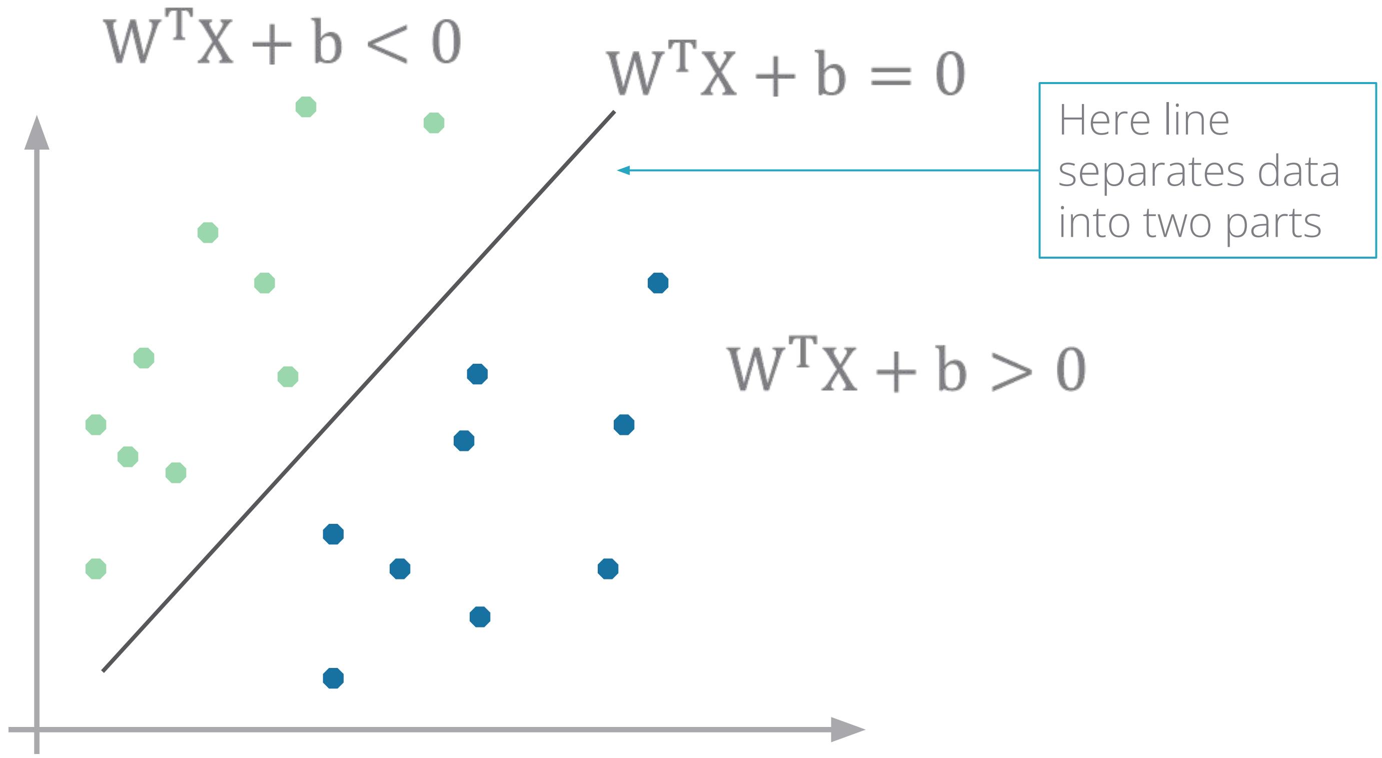
$$\boxed{\mathbf{W}^{\mathrm{T}}\mathbf{X} + \mathbf{b} = \mathbf{0}}$$

In p > 3 dimensions, it can be hard to visualize a hyper planes.

Separating a Hyper Plane

- Binary classification can be viewed as the task of separating classes in feature space:

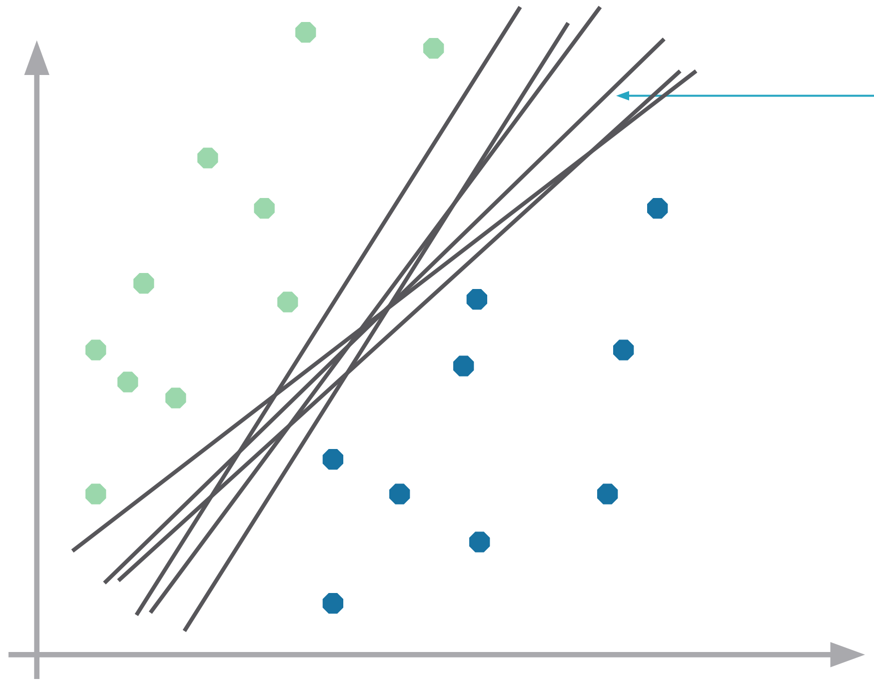
Fig. 01: Binary Classification



Linear Separators

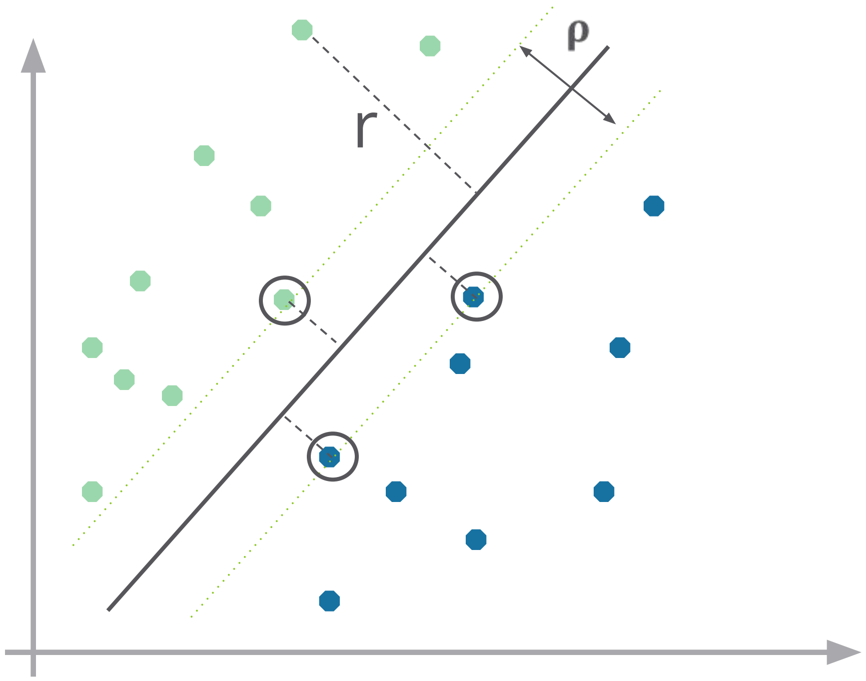
The objective in SVM is to find optimum separator

Fig. 02: Linear Separators



Which of the linear separators is optimal?

Classification Margin



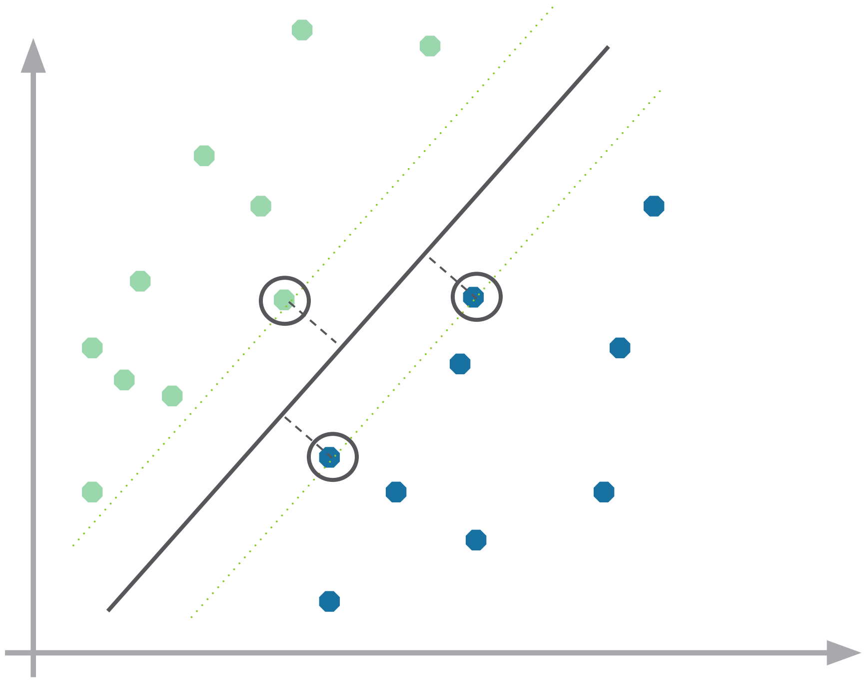
- Distance from case $\mathbf{x}_i$ to the separator is

$$r = \frac{w^T x_i + b}{\| w \|}$$

Here $\| w \|$ is length of a vector given by sqrt(sum(W^2))

- **Cases closest to the hyper plane are Support Vectors**
- **Margin ρ of the separator is the distance between support vectors**

Maximum Margin Classification



- The objective is now to maximize the margin ρ of the separator
- The focus is on 'Support Vectors'
- Other cases are not considered in the algorithm

Mathematical Approach to Linear SVM

Let training set be separated by a hyper plane with margin ρ. Then for each training observation

$$\begin{array}{ll} w^T x_i + b \leq -\rho/2 & \text{if } y_i = -1 \\ w^T x_i + b \geq \rho/2 & \text{if } y_i = 1 \end{array} \leftrightarrow \boxed{y_i(w^T x_i + b) \geq \rho/2}$$

For every support vector x_s the above inequality is an equality

After rescaling w and b by $\rho/2$ in the equality, we obtain that distance between each x_s and the hyper plane is

$$r = \frac{y_i(w^T x_s + b)}{\| w \|} = \frac{1}{\| w \|}$$

Margin can be expressed through (

$$\boxed{\rho = 2r = \frac{2}{\| w \|}}$$

Mathematical Approach to Linear SVM

Quadratic Optimisation problem is:

Find w and b such that

$\rho = \frac{2}{\|w\|}$ is maximised

and

$$y_i(w^T x_i + b) \geq 1$$

which can be reformulated as:

Find w and b such that

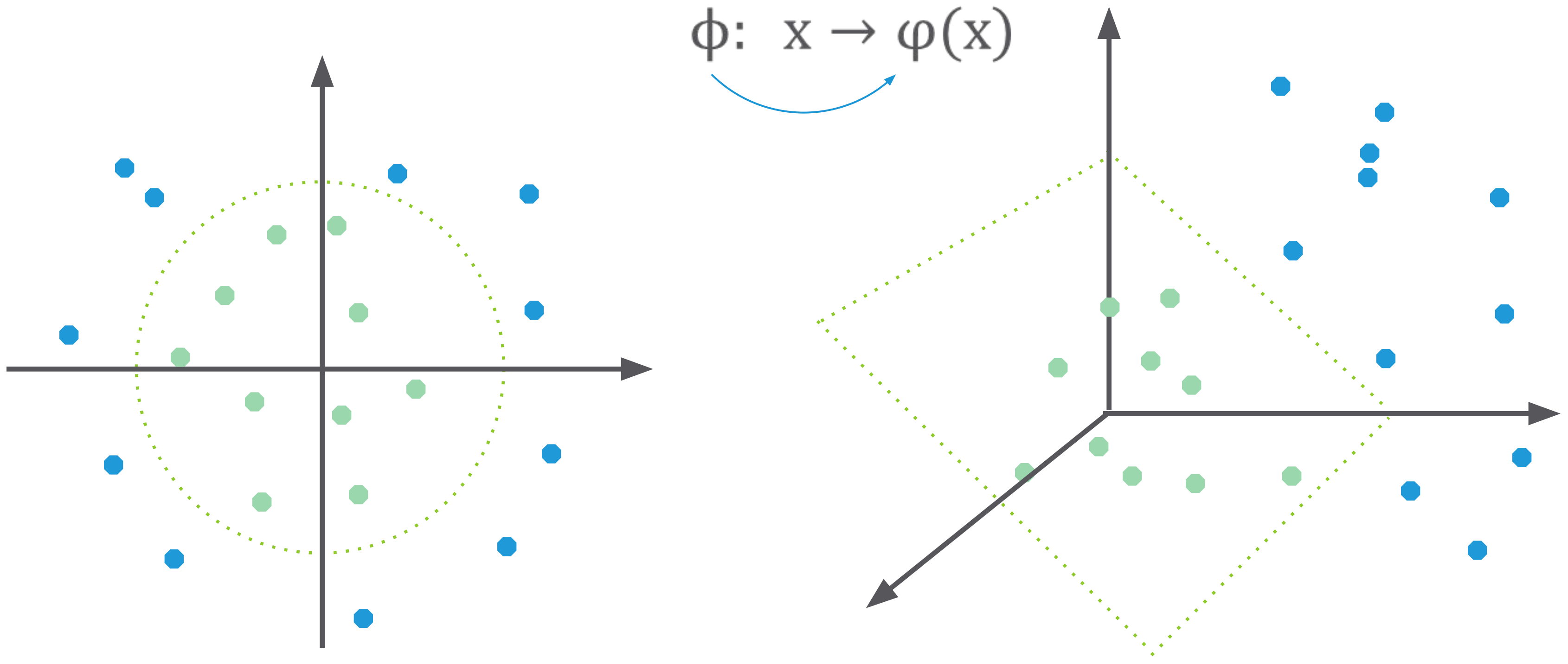
$\phi(w) = w^T w$ is minimised

and

$$y_i(w^T x_i + b) \geq 1$$

Non-Linear SVMs – Feature Spaces

General idea: The original feature space can always be mapped to some higher-dimensional feature space where the training set is separable



The "Kernel Trick"

The linear classifier relies on inner product between vectors

$$K(x_i, x_j) = x_i^T x_j$$

If every data point is mapped into high-dimensional space via some transformation $\phi\colon\ x \rightarrow \varphi(x)$

then the inner product becomes

$$K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$$

A kernel function is a function that is equivalent to an inner product in some feature space

The "Kernel Trick"

Example:

2-dimensional vector $x = [x_1 \quad x_2]$;

Let $K(x_i, x_j) = (1 + x_i^T x_j)^2$

Need to show that $K(x_i, x_j) = \varphi(x_i)^T \varphi(x_j)$:

$K(x_i, x_j) = (1 + x_i^T x_j)^2$

$= 1 + x_{i1}^2 x_{j1}^2 + 2x_{i1}x_{j1}x_{i2}x_{j2} + x_{i2}^2 x_{j2}^2 + 2x_{i1}x_{j1} + 2x_{i2}x_{j2}$

$= [1 \quad x_{i1}^2 \; \sqrt{2}x_{i1}x_{i2} \quad x_{i2}^2 \; \sqrt{2}x_{i1} \quad \sqrt{2}x_{i2}] \; T \; [1$

$x_{j1}^2 \; \sqrt{2}x_{j1}x_{j2} \quad x_{j2}^2 \; \sqrt{2}x_{j1} \quad \sqrt{2}x_{j2}]$

$= \varphi(x_i)^T \varphi(x_j)$ where $\varphi(x) = [1 \; x_1^2 \sqrt{2}x_1 x_2 \; x_2^2 \sqrt{2}x_1 \sqrt{2}x_2]$

Thus, a kernel function implicitly maps data to a high-dimensional space (Without the need to compute each $\varphi(x)$ explicitly)

Examples of Kernel Functions

Linear

$$K(x_i, x_j) = x_i{}^T x_j$$

Mapping ϕ

$x \rightarrow \varphi(x)$ where $\varphi(x)$ is x itself

Polynomial of power ρ

$$K(x_i, x_j) = (1 + x_i{}^T x_j)^\rho$$

Gaussian (Radial basis function)

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$$

Case Study – Predicting Loan Defaulters

Background

- The bank possesses demographic and transactional data of its loan customers. If the bank has a robust model to predict defaulters it can undertake better resource allocation.

Objective

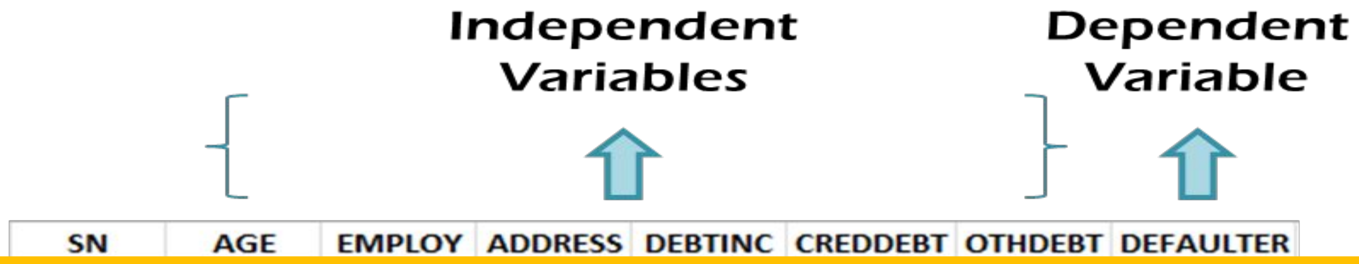
- To predict whether the customer applying for the loan will be a defaulter

Available Information

- Sample size is 700
- Age group, Years at current address, Years at current employer, Debt to Income Ratio, Credit Card Debts, Other Debts are the independent variables
- Defaulter (=1 if defaulter, 0 otherwise) is the dependent variable

Data Snapshot

BANK LOAN



Column	Description	Type	Measurement	Possible Values
SN	Serial Number	-	-	-
AGE	Age Groups	Integer	1(<28 years),2(28-40 years),3(>40 years)	3
EMPLOY	Number of years customer working at current employer	Integer	-	Positive value
ADDRESS	Number of years customer staying at current address	Integer	-	Positive value
DEBTINC	Debt to Income Ratio	Continuous	-	Positive value
CREDDEBT	Credit to Debit Ratio	Continuous	-	Positive value
OTHDEBT	Other Debt	Continuous	-	Positive value
DEFAULTER	Whether customer defaulted on loan	Integer	1(Defaulter), 0(Non-Defaulter)	2

SVM in R

```
# Importing and Readying the Data
```

```
bankloan<-read.csv("BANK LOAN.csv",header=T)

bankloan$AGE<-as.factor(bankloan$AGE)
```

as.factor() changes age from an integer to a factor variable.

```
str(bankloan)
```

str() is used to check if the conversion to factor has taken place and if all other variable formats are appropriate, before moving to SVM modeling.

```
# Output
```

```
'data.frame':	700 obs. of  8 variables:
 $ SN      : int  1 2 3 4 5 6 7 8 9 10 ...
 $ AGE     : Factor w/ 3 levels "1","2","3": 3 1 2 3 1 3 2 3 1 2 ...
 $ EMPLOY  : int  17 10 15 15 2 5 20 12 3 0 ...
 $ ADDRESS : int  12 6 14 14 0 5 9 11 4 13 ...
 $ DEBTINC : num  9.3 17.3 5.5 2.9 17.3 10.2 30.6 3.6 24.4 19.7 ...
 $ CREDDEBT : num  11.36 1.36 0.86 2.66 1.79 ...
 $ OTHDEBT  : num  5.01 4 2.17 0.82 3.06 ...
 $ DEFAULTER: int  1 0 0 0 1 0 0 0 1 0 ...
```

SVM in R

```
# SVM Using Package "e1071"

install.packages("e1071")
library(e1071)

model<-svm(formula=DEFAULTER~AGE+EMPLOY+ADDRESS+
           DEBTINC+CREDDEBT+OTHDEBT,data=bankloan,
           type="C",probability=TRUE,kernel="linear")

model
```

- **svm()** trains a support vector machine.
- **formula=** gives the model to be fit.
- **data=** specifies the data object.
- **type=** specifies whether SVM is used for classification or regression or novelty detection. Default for **type=** is "C“.
- **probability=** logical for indicating whether model should allow for probability predictions.
- **kernel=** specifies the kernel used in training and predicting. Here, we have kept kernel as linear.

SVM in R

```
# Output
```

```
> model

Call:
svm(formula = DEFAULTER ~ AGE + EMPLOY + ADDRESS + DEBTINC + CREDDEBT + OTHDEBT,
 data = bankloan, type = "C", probability = TRUE,
    kernel = "linear")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  linear
       cost:  1

Number of Support Vectors:  312
```

Predictions Based on SVM

```
# Predictions
```

```
pred1<-predict(model,bankloan,probability=TRUE)
```

- **predict()** returns predicted probabilities based on the model results and historical data.
- First argument is the **svm()** model object while the second argument is original dataset.
- **probability=TRUE** returns raw probabilities. This argument is valid only when **type="probability“** is specified in **svm()**.

```
pred2<-attr(pred1,"probabilities")[,1]
```

- **attr()**, from base R, is used get or set specific attributes of an object. Here, we want to get the predicted probabilities obtained by the **svm()** model.
- First argument is the name of the object whose attributes we want to extract.
- Second argument is the character string specifying which attribute is to be accessed. Check **pred1** to know the exact name, which is **"probabilities"**.

ROC Curve and Area Under ROC Curve

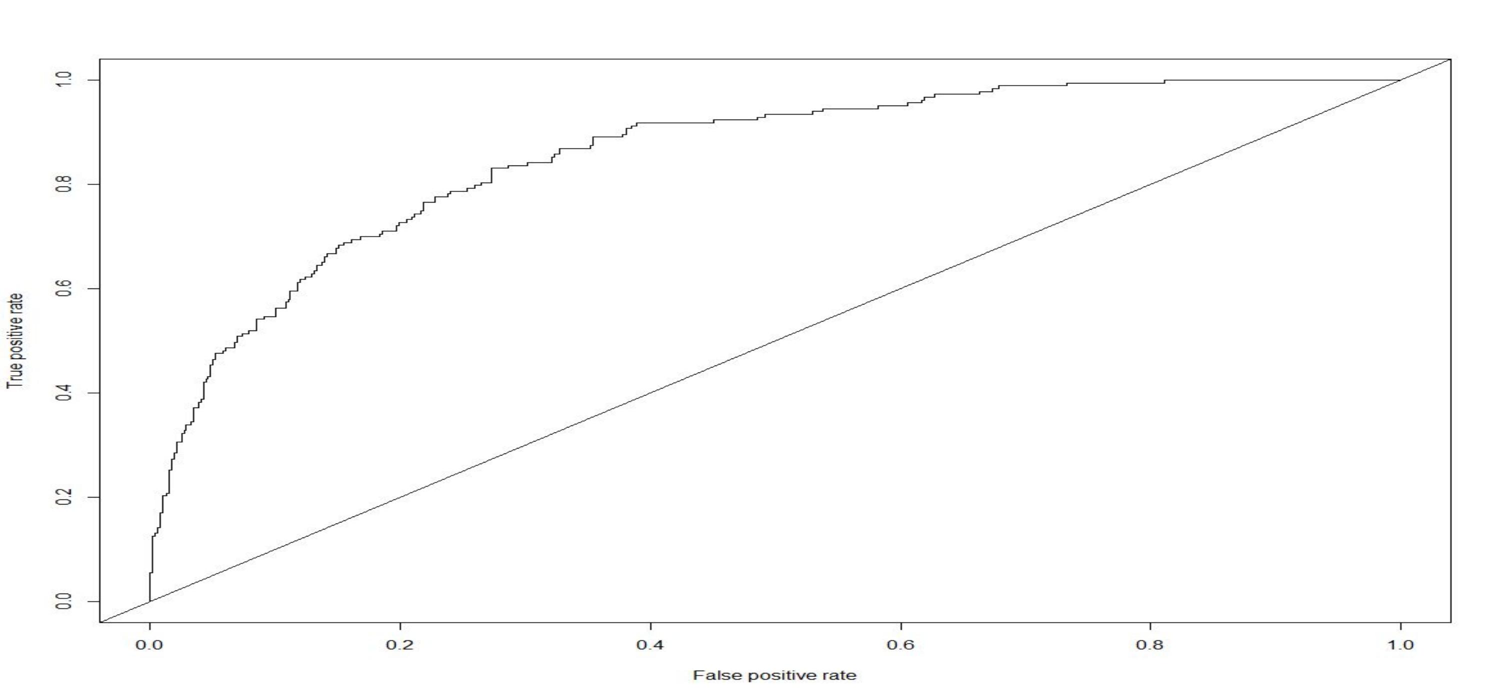
```
#ROC Curve

install.packages("ROCR")
library(ROCR)

pred<-prediction(pred2,bankloan$DEFAULTER)

perf<-performance(pred,"tpr","fpr")

plot(perf)

abline(0,1)
```

prediction() creates object of class prediction, required for ROC curve. **performance()** calculates predictor evaluations. Using **measure="tpr", measure="fpr"** we can plot an ROC Curve.

abline()
adds a straight line to the plot.

ROC Curve and Area Under ROC Curve

```
# Output
```




```
# Area Under ROC Curve
auc<-performance(pred,"auc")
auc@y.values
[[1]]
[1] 0.855577
```

“**auc**” in performance() calculates Area Under ROC Curve.

Quick Recap

Support Vector Machines	• SVMs find a hyper plane which separates the d-dimensional data perfectly into its classes • Since training data is often not linearly separable, SVM's introduce the notion of a “Kernel-induced Feature Space" which casts the data into a higher dimensional space where the data is separable
SVM in R	• Package **`"e1071"`** has **`svm()`** that trains a support vector machine • The function takes arguments to specify whether **`svm()`** is to be used for classification or regression; if probabilities are to be returned and which kernel to use for training and predicting