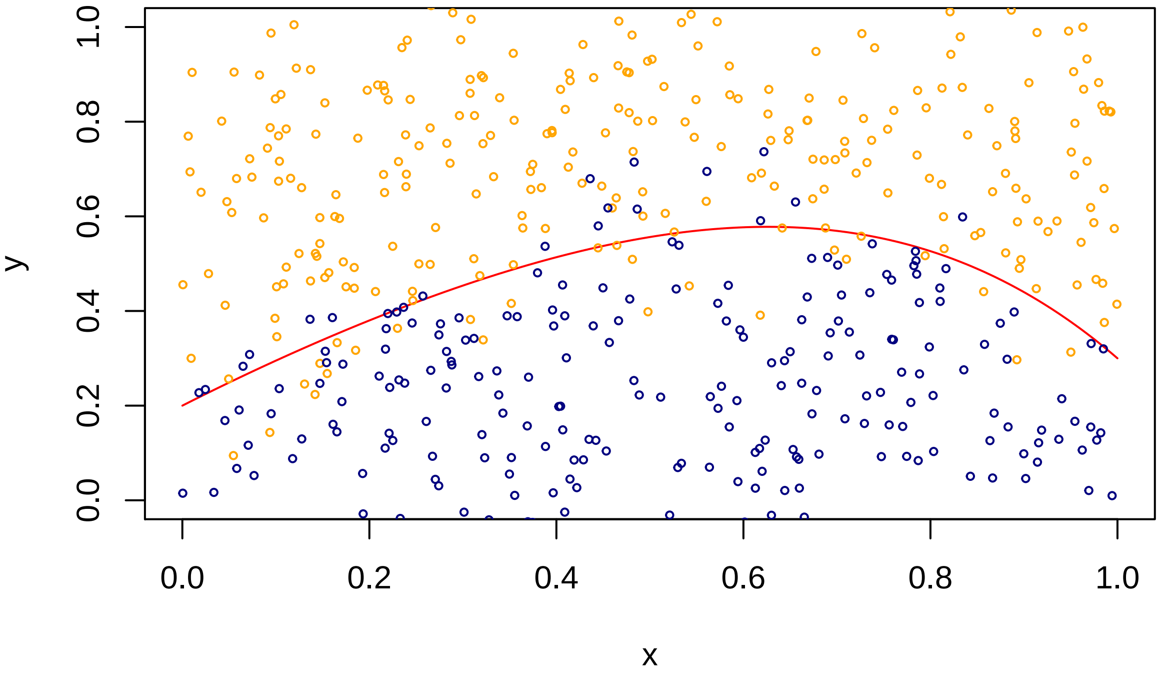
improved_classification.Rmd

Improving global classification algorithm

Simulating data

Assume a true decision boundary in a unit square with functional form: $y = 0.2 + x + 0.5x^2 + 0.1x^3 - 0.5x^4$

```
f<-function(x){
  return(0.2 + x - 0.5*x^2 + 0.1*x^3 - 0.5*x^4)
}
xv = seq(0,1,0.001)
yv = f(xv)

dx = runif(500)
dy = runif(500)

boundry = f(dx)
label = (dy>boundry)+0

x_value = dx
y_value = dy + rnorm(length(dy),sd=0.1)
training_data = cbind(y_value, x_value, label)
plot(yv~xv,xlim=c(0,1), ylim=c(0,1),t = "l",col="red",xlab="x", ylab="y")
points(y_value[label==1]~x_value[label==1],cex=0.5,col="orange")
points(y_value[label==0]~x_value[label==0],cex=0.5,col="navyblue")
```

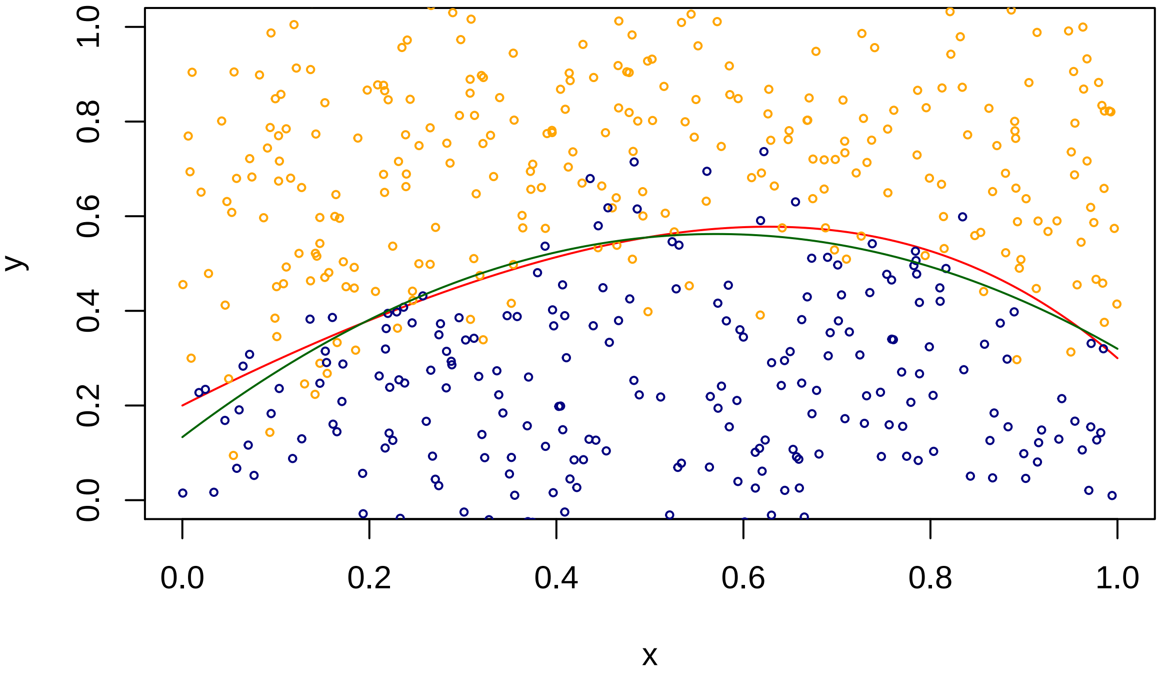


Improved global classifier 1

Here we simply introduce polynomial terms into the probit regression model

1. To start, add a quadratic term

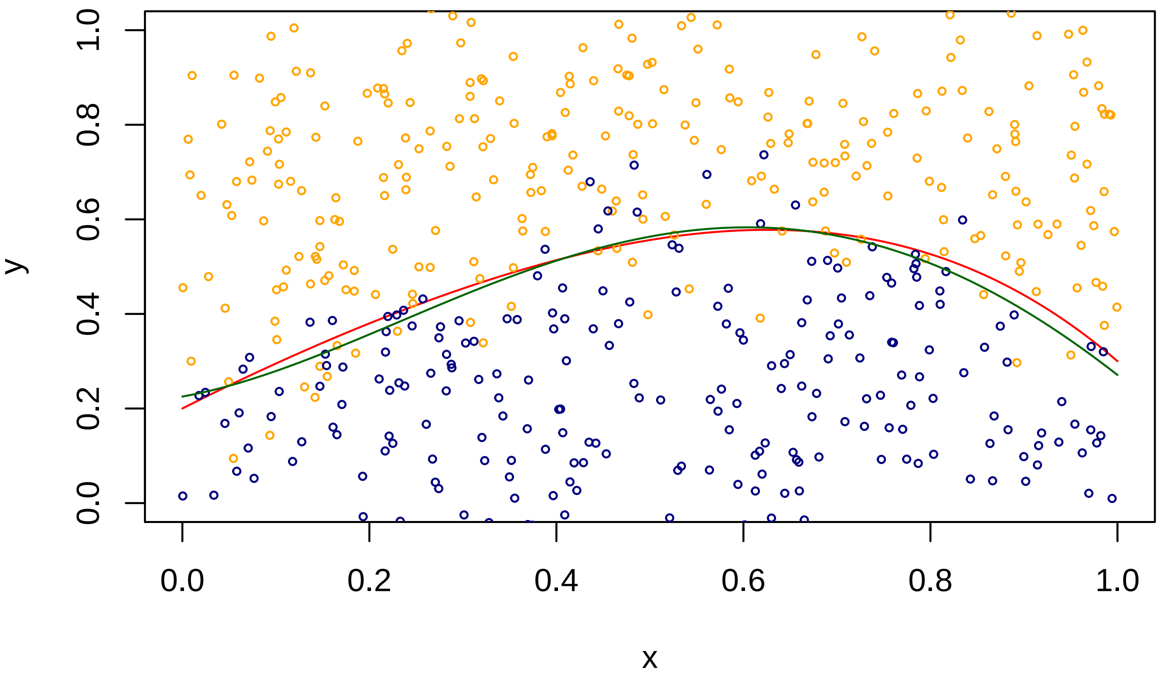
```
x1 = x_value
x2 = x_value^2
fit = glm(label~x1+x2+y_value,family=binomial(link="probit"))
c = -(fit$coef/fit$coef[4])[1:3]
yv2 = c[1] + c[2]*xv + c[3]*xv^2
plot(yv~xv,xlim=c(0,1), ylim=c(0,1),t = "l",col="red",xlab="x", ylab="y")
points(y_value[label==1]~x_value[label==1],cex=0.5,col="orange")
points(y_value[label==0]~x_value[label==0],cex=0.5,col="navyblue")
lines(yv2~xv, col = "darkgreen")
```



This is a signficant improvement from the linear case.

2. Let's go all the way to x^4

```
x1 = x_value
x2 = x_value^2
x3 = x_value^3
x4 = x_value^4
fit = glm(label~x1+x2+x3+x4+y_value,family=binomial(link="probit"))
c = -(fit$coef/fit$coef[6])[1:5]
yv3 = c[1] + c[2]*xv + c[3]*xv^2 + c[4]*xv^3 + c[5]*xv^4

plot(yv~xv,xlim=c(0,1), ylim=c(0,1),t = "l",col="red",xlab="x", ylab="y")
points(y_value[label==1]~x_value[label==1],cex=0.5,col="orange")
points(y_value[label==0]~x_value[label==0],cex=0.5,col="navyblue")
lines(yv3~xv, col = "darkgreen")
```



The results looks better than the previous model, but not significantly better.

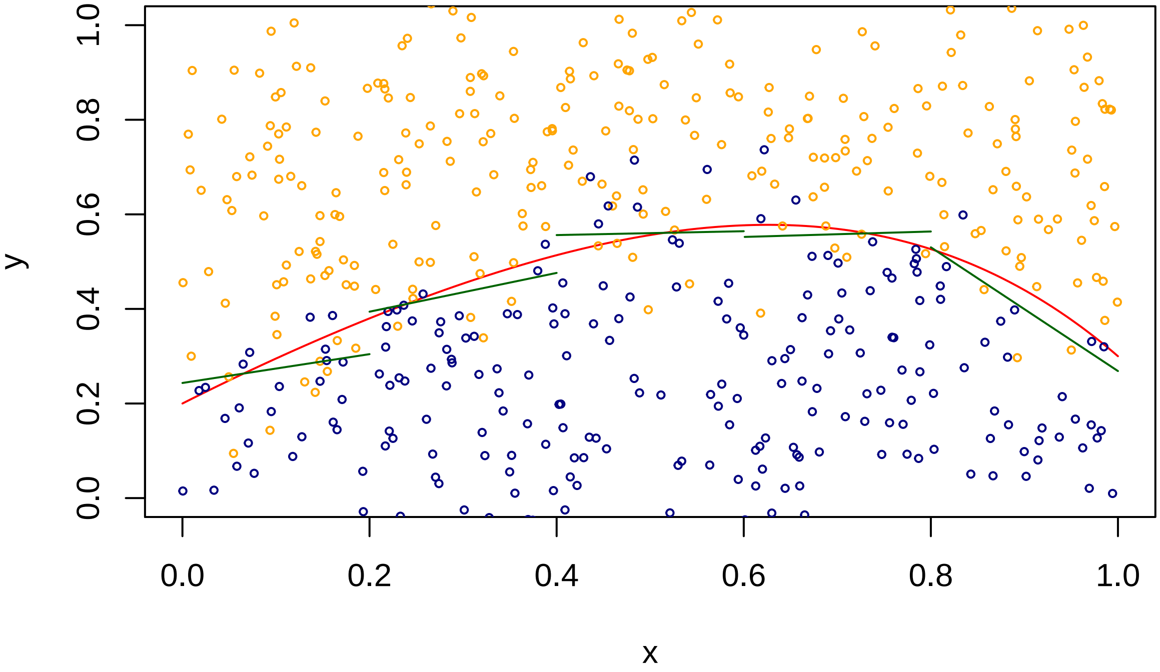
Improved global classifier 2

Here we try another idea with piece-wise linear classifer, i.e., for different x region, we fit different probit model

```
lb = seq(0,0.8,0.2)
rb = seq(0.2,1.0,0.2)

get_line<-function(outcome,x,y){
  fit = glm(outcome~x+y,family=binomial(link="probit"))
  c = -(fit$coef/fit$coef[3])[1:2]
  return(c)
}



coef = t(sapply(1:length(lb), function(x) get_line(label[x_value>=lb[x]&x_value<=rb[x]], x_value[x_valu
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
plot(yv~xv,xlim=c(0,1), ylim=c(0,1),t = "l",col="red",xlab="x", ylab="y")
points(y_value[label==1]~x_value[label==1],cex=0.5,col="orange")
points(y_value[label==0]~x_value[label==0],cex=0.5,col="navyblue")
pl<-function(mu, beta, x){
  y = mu+beta*x
  lines(y~x,col="darkgreen")
}
null_out= sapply(1:length(rb), function(x) pl(coef[x,1],coef[x,2], xv[xv>=lb[x]&xv<=rb[x]]))
```



y
0.0
0.2
0.4
0.6
0.8
1.0
x
0.0
0.2
0.4
0.6
0.8
1.0