Group Assignment 9

Student ID:474084,476397,474869,457942,473928

(a)

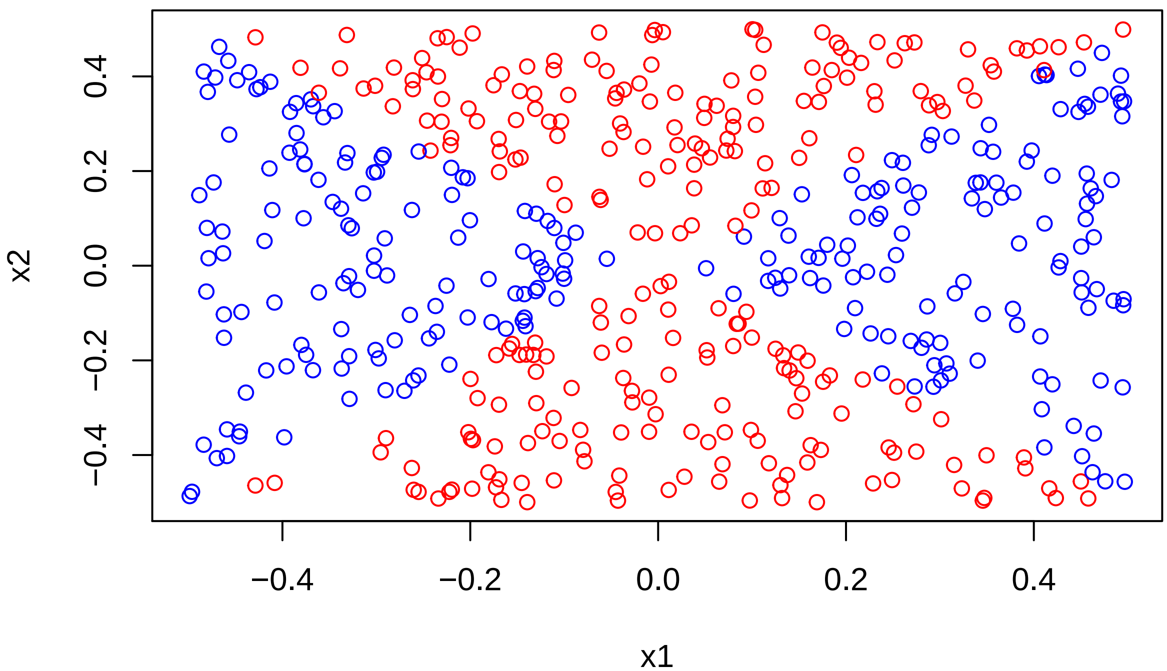
```
set.seed(100)
x1=runif (500) -0.5

x2=runif (500) -0.5

y=1*(x1^2-x2^2 > 0)
```

(b)

```
plot(x=x1,y=x2,col=ifelse(y==0, "red", "blue"),xlab="x1",ylab="x2")
```



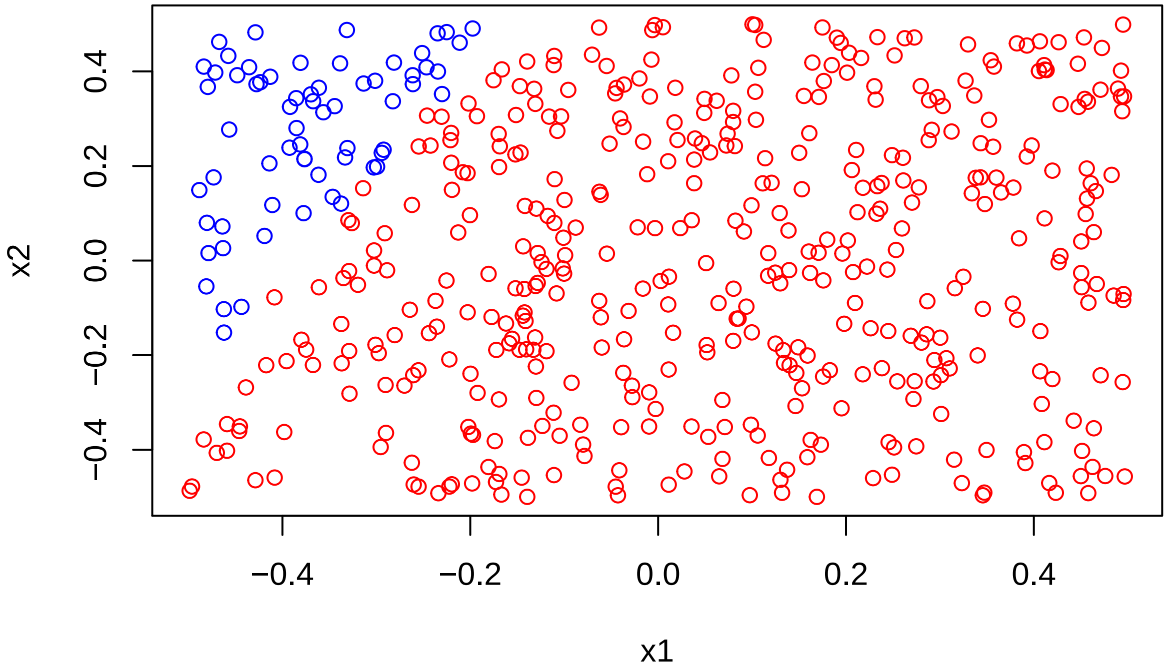
(c)

```
glm.fit=glm(y~x1+x2,family=binomial)
glm.fit
```

```
##
## Call:  glm(formula = y ~ x1 + x2, family = binomial)
##
## Coefficients:
## (Intercept)           x1           x2
##    -0.05600     -0.14615      0.06528
##
## Degrees of Freedom: 499 Total (i.e. Null);  497 Residual
## Null Deviance:       692.8
## Residual Deviance: 692.5     AIC: 698.5
```

(d)

```r
cdata=data.frame(cbind(x1,x2))
pred.probs=predict(glm.fit,cdata,type="response")
glm.pred=ifelse(pred.probs>0.5,1,0)
plot(x1,x2, col=ifelse(glm.pred==0,"red","blue"))
```



(e)

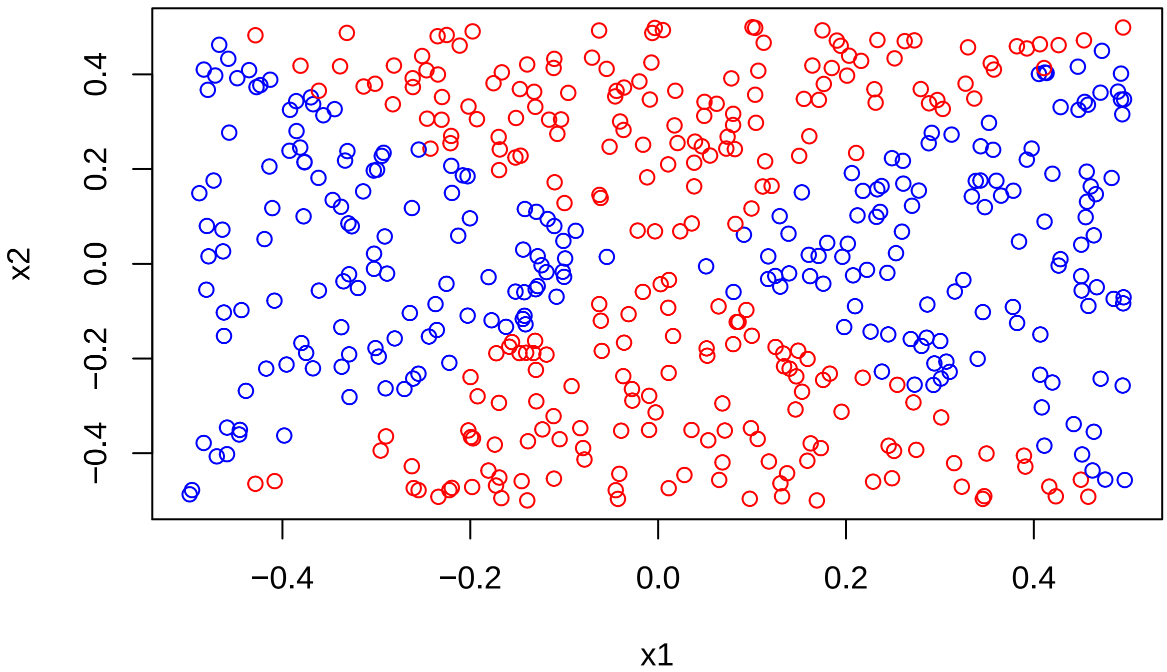
```r
glm.fit2=glm(y~I(x1^2)+I(x2^2),family=binomial)
```

```
## Warning: glm.fit: algorithm did not converge

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

(f)

```r
dat=data.frame(cbind(x1,x2))
pred.probs2=predict(glm.fit2,newdata=dat,type="response")
glm.pred2=ifelse(pred.probs2>0.5,1,0)
plot(x1,x2,col=ifelse(glm.pred2==0,"red","blue"))
```




```r
table(y,glm.pred2)
```

```
##    glm.pred2
## y     0   1
##   0 257   0
##   1   0 243
```

(g)

```r
library(e1071)
data.x=data.frame(x=cbind(x1,x2),y=as.factor(y))
set.seed(100)
tune.out=tune(svm,y~.,data=data.x,kernel="linear",
              ranges=list(cost=c(0.001,0.01,0.1,1,5,10,100)))
tune.out
```

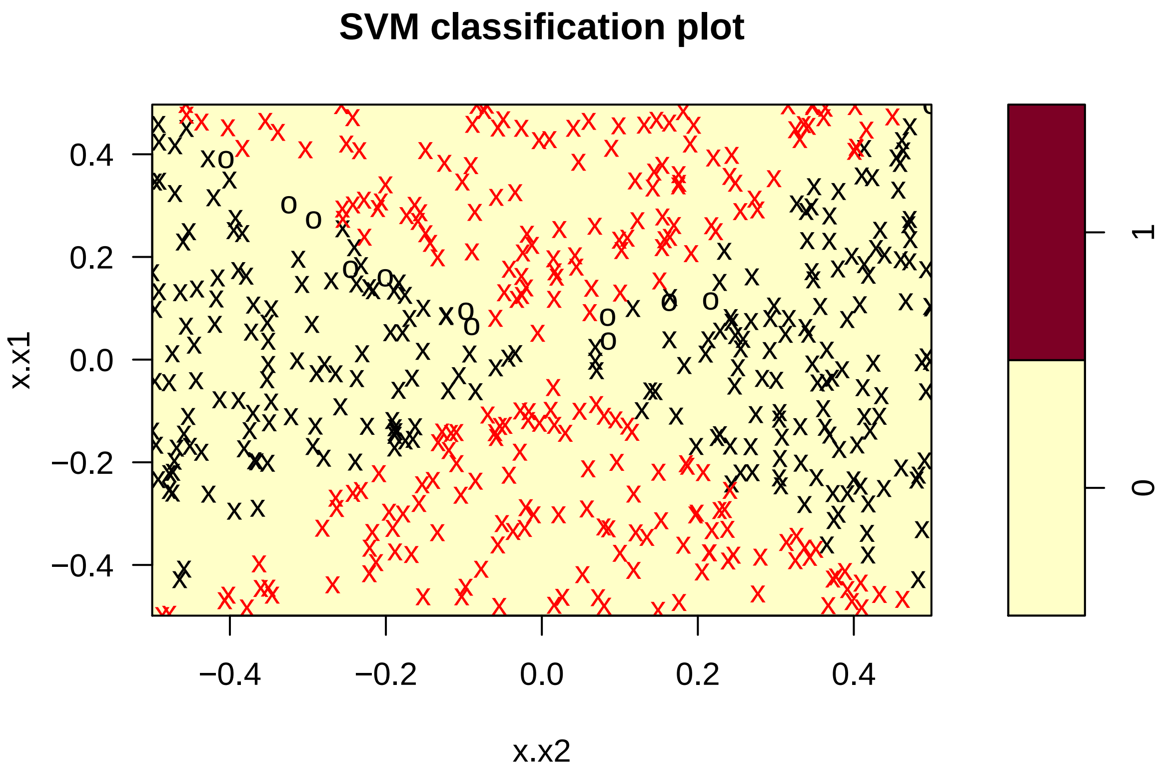
```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##   cost
##  0.001
##
## - best performance: 0.522
```

```r
bestmod=tune.out$best.model
summary(bestmod)
```

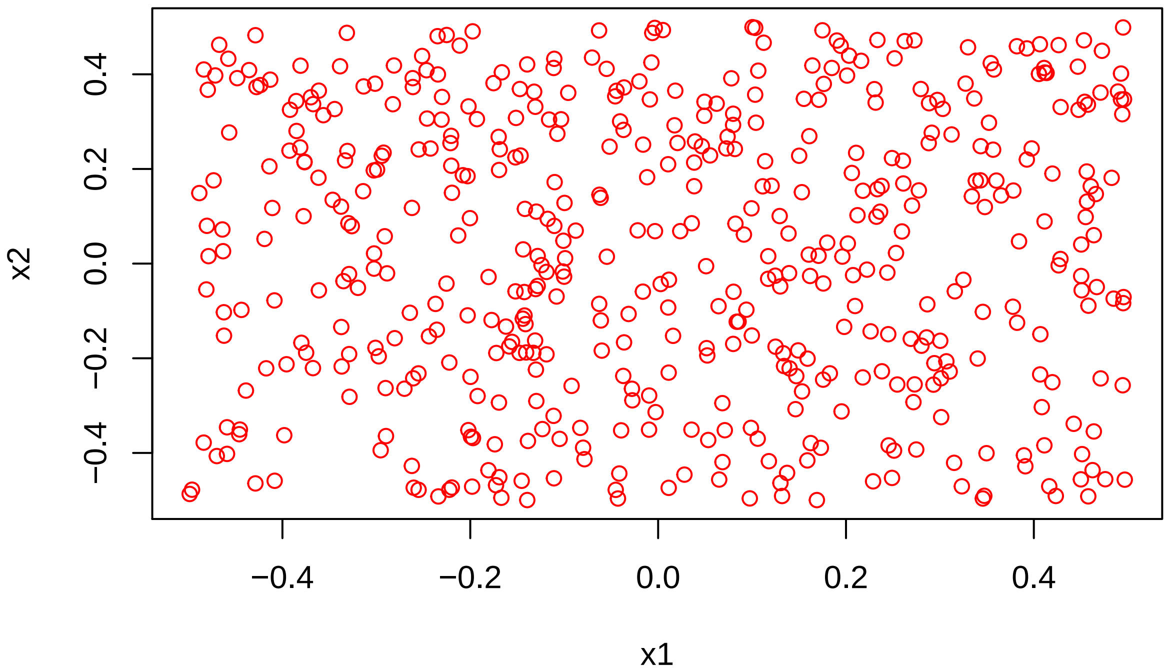
```
##
## Call:
## best.tune(method = svm, train.x = y ~ ., data = data.x, ranges = list(cost = c(0.001,
##     0.01, 0.1, 1, 5, 10, 100)), kernel = "linear")
##
```

```
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  0.001
##
## Number of Support Vectors:  488
##
##  ( 245 243 )
##
##
## Number of Classes:  2
##
## Levels:
##  0 1
```

```
plot(bestmod,data.x)
```




```
svm.pred=predict(bestmod,data.x)
plot(x1,x2,col=ifelse(svm.pred==0,"red","blue"))
```

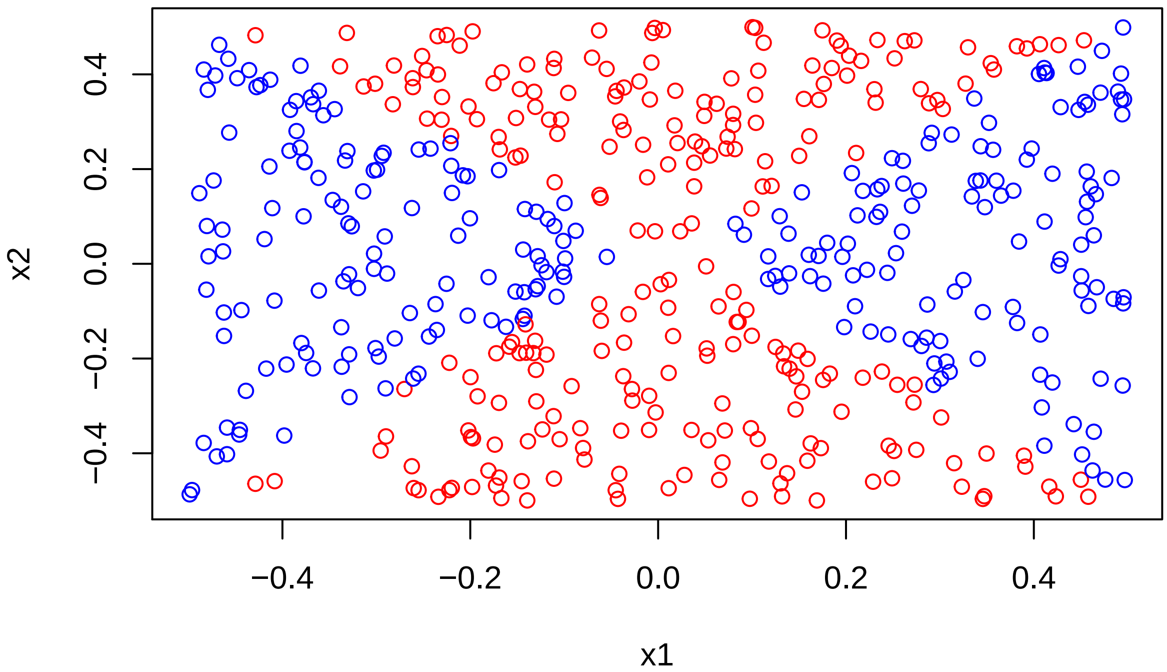


(h)

```r
set.seed(1)
tune.out=tune(svm,y~.,data=data.x,kernel="polynomial",
              ranges=list(cost=c(0.1,1,10,100,1000),degree=c(0.5,1,2,3,4)))
ypred=predict(tune.out$best.model,data.x)
plot(x1,x2,col=ifelse(ypred==0,"red","blue"))
table(y,ypred)
```

```
##    ypred
## y     0   1
##   0 247  10
##   1   7 236
```

```r
set.seed(1)
tune.out=tune(svm,y~.,data=data.x,kernel="polynomial",
              ranges=list(cost=c(0.1,1,10,100,1000),degree=c(0.5,1,2,3,4)))
ypred=predict(tune.out$best.model,data.x)
plot(x1,x2,col=ifelse(ypred==0,"red","blue"))
```




```r
table(y,ypred)
```

```
##    ypred
## y     0   1
##   0 247  10
##   1   7 236
```

(i) According to the chuncks above, we can see that logistic method with transformed x and SVM method can give a good result of prediction for the non-linear decision boundary. And the error rates are low. When compared with each other, logistic approach needs us to transform predictor x into different forms, and SVM needs us to tune the cost of funtion. And the tuning of cost is easier than transformation.Besides, we only predict on one data set, which means we need more test data to tune out the model to avoid overfitting.