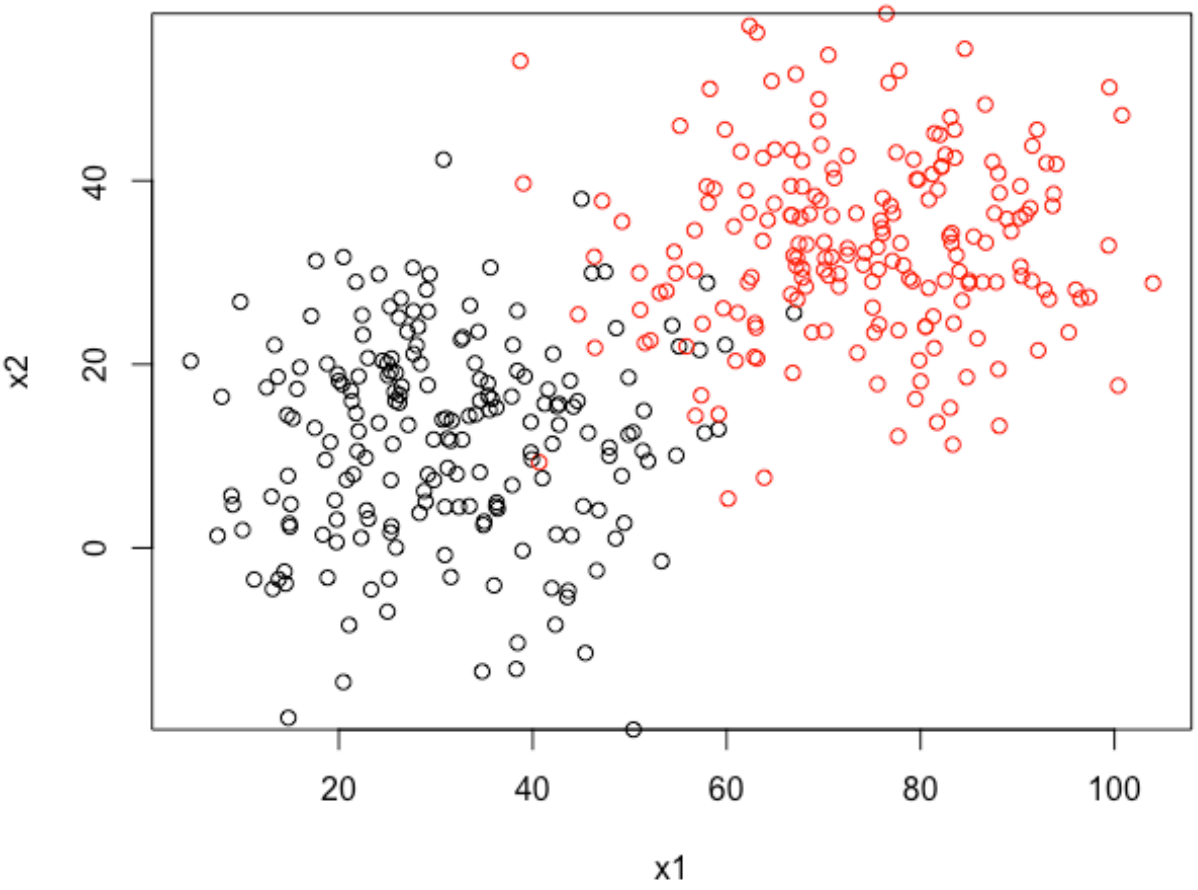
# 1. scatter diagram

the scatter diagram showed as follows,we can see there are two classes in this dataset, we can draw primary conclusion when x1,x2 are both small, it belong to class 1, and when x1,x2 are both large, it belong to class 2. We can use SVM analysis to find the hyperplane.




# 2. SVM Analysis

- scale the features range [0,1],code as follows

```
range01 <- function(x){(x-min(x))/(max(x)-min(x))}
df[1] = range01(df[1])
df[2] = range01(df[2])
```

- Apply the SVM method using RBF kernel

we first divide the data set into train dataset and test data set. we choose 75% of the dataset as train data using sample function and the 25% left be the test data. And then fit SVM method using RBF kernel. From the result, we can see the number of supported Vectors is 48 when using defalut values of gamma 0.5 and cost 1.

```
set.seed(123)
sample <- sample.int(n = nrow(df), size = floor(.75*nrow(df)), replace = F)
df.train = df[sample,]
df.test = df[-sample,]

svmfit = svm(l~.,data = df.train,type ="C-classification")
plot(svmfit,data = df.train)
```

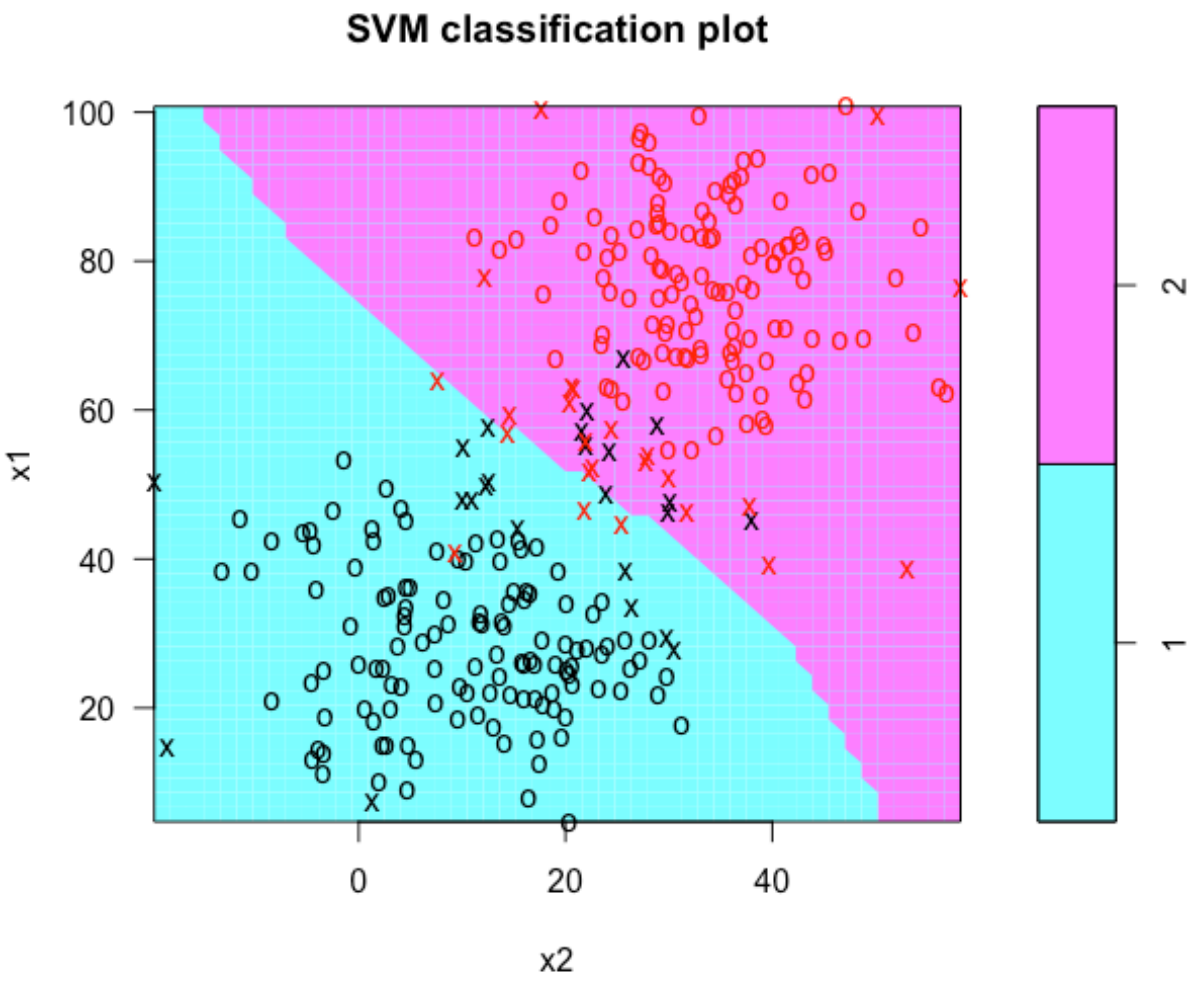
```
> svmfit = svm(l~.,data = df.train,type ="C-classification")
> plot(svmfit,data = df.train)
> svmfit

Call:
svm(formula = l ~ ., data = df.train, type = "C-classification")


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1
      gamma:  0.5

Number of Support Vectors:  48

> set.seed(123)
```




- SVM tune to find the best parameters.

  we use package `e1071` to do grid search in conjunction with 5 folds cross validation to tune the SVM models with gamma range from $2^{-15}$ ~ $2^{3}$ and cost range from $2^{-5}$ ~$2^{15}$ to run to get the summary of it. We can see when gamma is 0.125 and cost 8 is the best parameters. We also can get information about the error( accuray) about every parameters combinations in the performace field of the result, which we will used for the 3D plot.

```
tunedResult = tune.svm(factor(l)~.,data = df.train,gamma =2^(seq(-15,3,2)),
cost = 2^(seq(-5,15,2)),tunecontrol=tune.control(cross=5))
```

```
> summary(tunedResult)

Parameter tuning of ‘svm’:

- sampling method: 5-fold cross validation

- best parameters:
 gamma cost
 0.125    2

- best performance: 0.04745763

- Detailed performance results:
           gamma       cost      error  dispersion
1   3.051758e-05 3.1250e-02 0.54915254 0.039019879
2   1.220703e-04 3.1250e-02 0.54915254 0.039019879
3   4.882812e-04 3.1250e-02 0.54915254 0.039019879
4   1.953125e-03 3.1250e-02 0.54915254 0.039019879
5   7.812500e-03 3.1250e-02 0.54915254 0.039019879
6   3.125000e-02 3.1250e-02 0.08135593 0.052787836
7   1.250000e-01 3.1250e-02 0.05762712 0.025704663
8   5.000000e-01 3.1250e-02 0.05762712 0.025704663
9   2.000000e+00 3.1250e-02 0.06101695 0.030789665
10  8.000000e+00 3.1250e-02 0.51525424 0.094063979
11  3.051758e-05 1.2500e-01 0.54915254 0.039019879
12  1.220703e-04 1.2500e-01 0.54915254 0.039019879
13  4.882812e-04 1.2500e-01 0.54915254 0.039019879
14  1.953125e-03 1.2500e-01 0.54915254 0.039019879
15  7.812500e-03 1.2500e-01 0.05423729 0.022098991
16  3.125000e-02 1.2500e-01 0.05762712 0.025704663
17  1.250000e-01 1.2500e-01 0.05762712 0.025704663
18  5.000000e-01 1.2500e-01 0.05762712 0.025704663
```

- we can redo svmfit with the best parameters and do accuraryaccuracy computation on the train data and test data on this defalut model and get s 0.9525424 and 0.969697 on the tain dataset and test dataset perspectively

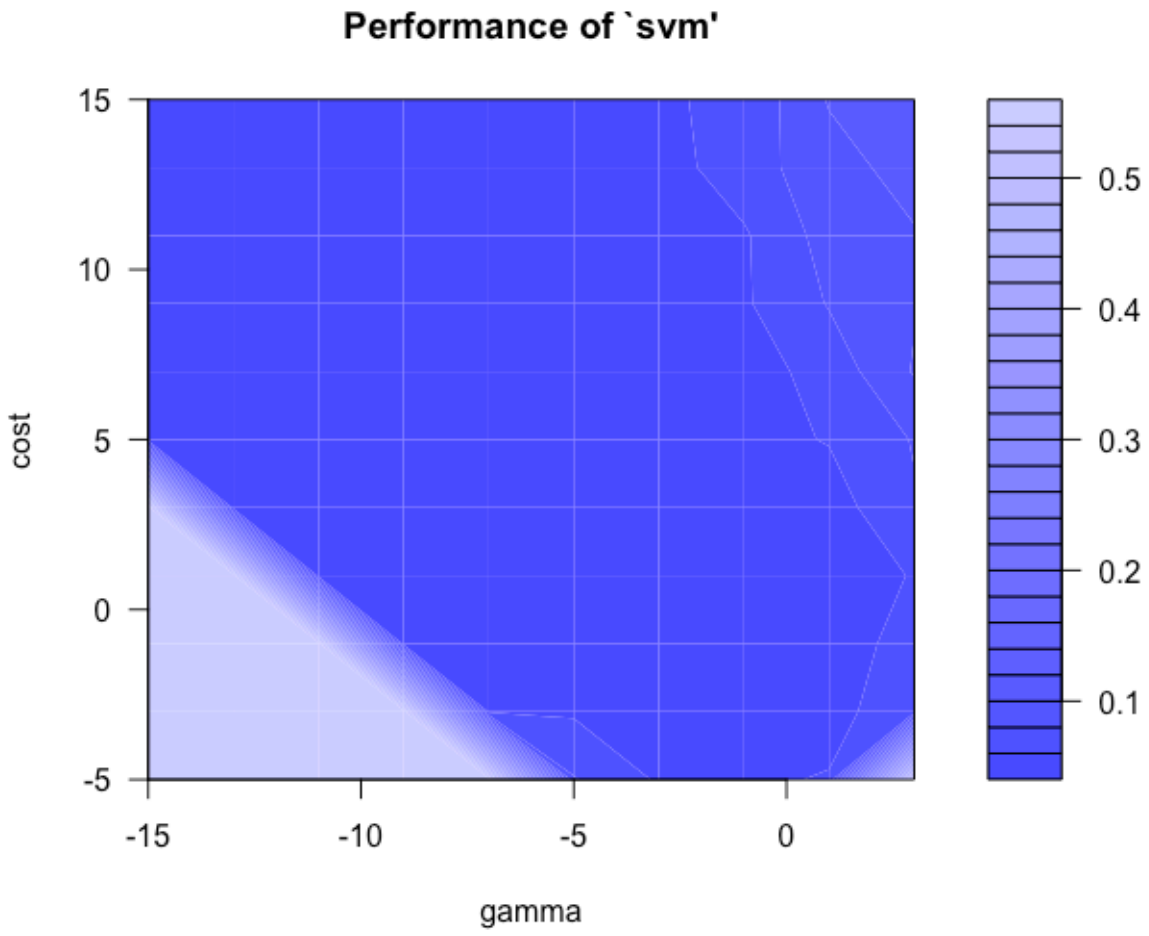
```
svmfit = svm(factor(l)~.,data = df.train,gamma=0.125,cost =8)

pred_train = predict(svmfit,df.train)
mean(pred_train== df.train$l)

pred_test = predict(svmfit,df.test[,-3],type = "class")
plot(pred_test)
tab = table(pred_test,df.test[,3])
mean(pred_test == df.test[,3])
```

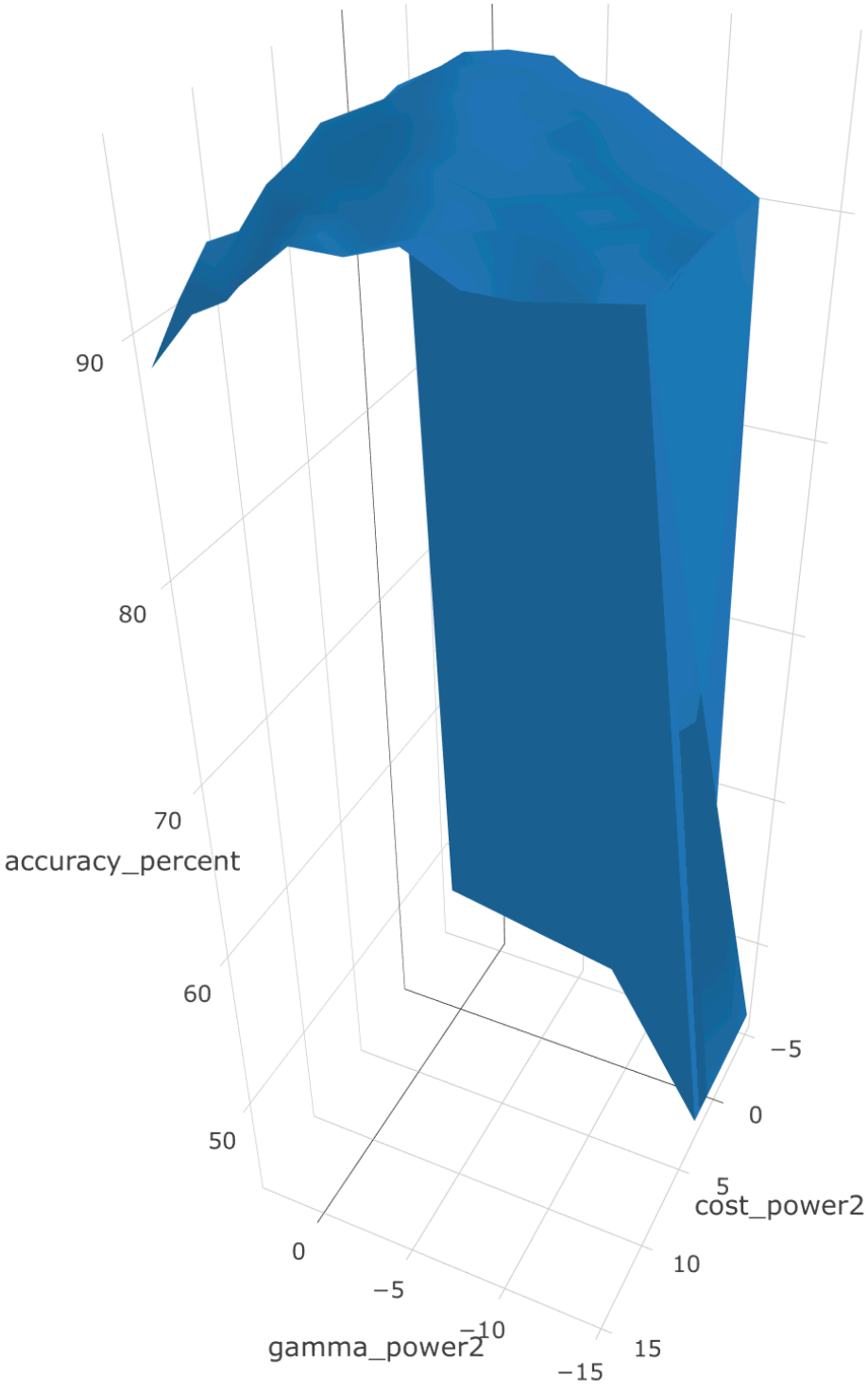
- Plot results

  Firstly, the performane contour is a way to graph SVM tune,shown below.

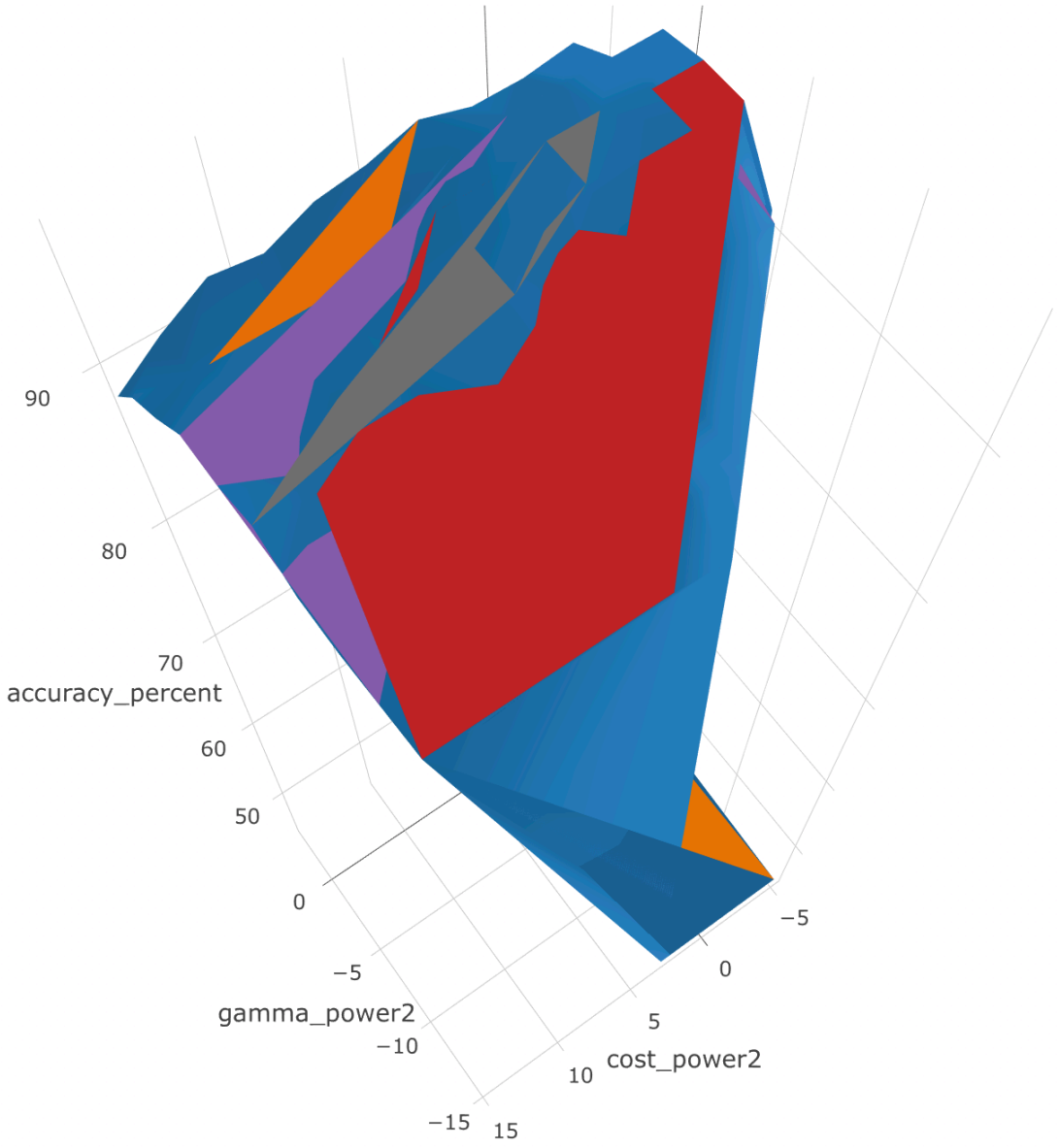



  Also, we do the 3D surface, x axis is to represent gammar parameter , which is gammar = $2^x$ and , y axis is to represent cost parameter , which is cost = $2^y$ , and z represent the accuracy .

  We first draw the (gammar,cost,accuracy ) as flowers,




And then, we use `add_trace` to trace the groups with same z value,as shown below




the code is as follows

```
gamma_power2 = tunedResult$performances$gamma
cost_power2 =tunedResult$performances$cost
accuracy_percent = tunedResult$performances$z
p <- plot_ly(x = ~gamma_power2, y = ~cost_power2, z = ~accuracy_percent,
type = 'mesh3d',xlab = "gammar",ylab="cost")

svmperformance = tunedResult$performances
uni = unique(svmperformance$z)
n = length(uni)
for(i in 1:n){
  sub1 = svmperformance[svmperformance$z == uni[i],]
  p = p %>% add_trace(x = ~sub1$gamma, y = ~sub1$cost,  z=~sub1$z)
}
```

```
print(p)
```

# Discussion

In this project, we learn how to use SVM for classification. We firstly do scatter diagram and draw primary conclusion when x1,x2 are both small, it belong to class 1, and when x1,x2 are both large, it belong to class 2. And then,we begin to do SVM analysis to find the hyperplane.Firstly, we need to partition the dataset into train and test data. And for tain data, we apply SVM method using RBF kernel, and use grid search in conjunction with 5 folds cross validation to tune the SVM models to find the best gammar and cost for SVM models. In this case, the best parameters are gamma 0.125 and cost 8 , we also use this best parameter to redo SVM fit, and use the test data to test the accuracy and gets 0.969697. For the SVM tune result, we graph performance contours and 3D plots, which use colors to indicate regions with the same accuracy. We implemented this project in R language, where `e1071` package is developed for SVM analysis.