Table of Contents

Support Vector Machine 1
Aim: 1
Theory: 1
SVM with fisheriris 1
SVM with Exor gate 2
Conclusion: 4

Support Vector Machine

Name: Ventrapragada Sai Shravani

PRN:17070123120

Batch:Entc(2017-21) G-3

```
clc;
clear all;
close all;
```

Aim:

Implementation of Support Vector Machine for FisherIris and EXOR gate.

Theory:

The worth of a classifier is not in how well it seperates the training data.

SVMS try to find all such data that correctly classify the training data and among all such lines pick the one that has the greatest distance to the points closest to it.

For data that isn't linearly sepereable we can project data to a space where it is almost linealy seperable.

An important aspect of SVMs is that all the mathematical machinery that it uses, the exact projection of the number of dimensions doesn't show up. You can write all of it in terms of dot products between various data points represented as vectors.

SVM with fisheriris

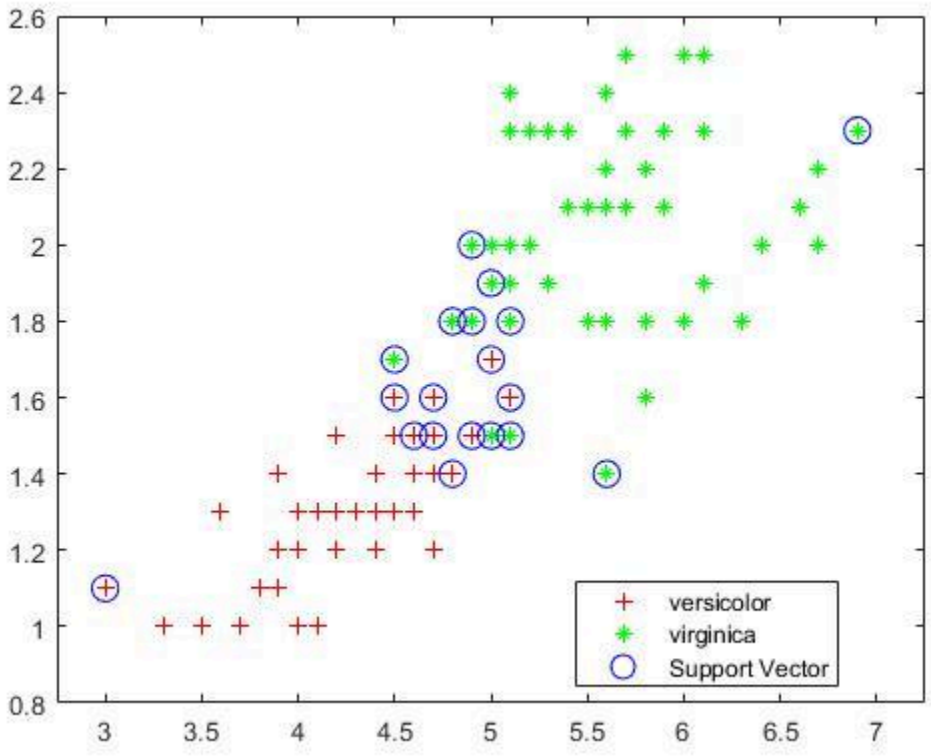
```
load fisheriris
inds = ~strcmp(species,'setosa');
X = meas(inds,3:4);
Y = species(inds);
svmModel = fitcsvm(X,Y,'KernelFunction','RBF')

Sv = svmModel.SupportVectors;
figure
gscatter(X(:,1),X(:,2),Y,'rg','+*');
hold on
plot(Sv(:,1),Sv(:,2),'bo','MarkerSize',10)
```

```
axis([2.75 7.25 0.8 2.6])
legend('versicolor','virginica','Support Vector')
hold off
```

```
svmModel =

  ClassificationSVM
             ResponseName: 'Y'
    CategoricalPredictors: []
               ClassNames: {'versicolor'  'virginica'}
           ScoreTransform: 'none'
          NumObservations: 100
                    Alpha: [23×1 double]
                     Bias: 0.1507
         KernelParameters: [1×1 struct]
           BoxConstraints: [100×1 double]
          ConvergenceInfo: [1×1 struct]
          IsSupportVector: [100×1 logical]
                   Solver: 'SMO'
```



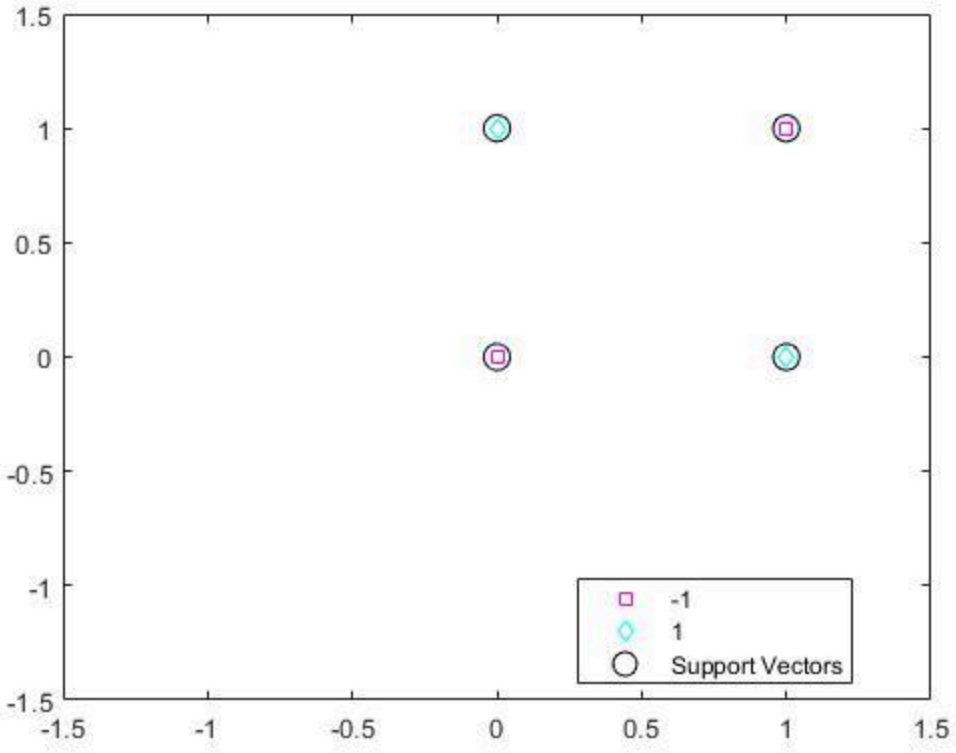
SVM with Exor gate

```
x=[0 0; 0 1; 1 0; 1 1];
```

```
y=[0; 1;  1; 0];
SVMModel = fitcsvm(x,y,'KernelFunction','RBF',...
    'KernelScale','auto')
sv = SVMModel.SupportVectors;
figure
gscatter(x(:,1),x(:,2),y,'mc','sd');
hold on
plot(sv(:,1),sv(:,2),'ko','MarkerSize',10)
axis([-1.5 1.5 -1.5 1.5])
legend('-1','1','Support Vectors');
hold off
```

```
SVMModel =

  ClassificationSVM
             ResponseName: 'Y'
    CategoricalPredictors: []
               ClassNames: [0 1]
          ScoreTransform: 'none'
         NumObservations: 4
                   Alpha: [4×1 double]
                    Bias: 0
        KernelParameters: [1×1 struct]
          BoxConstraints: [4×1 double]
         ConvergenceInfo: [1×1 struct]
         IsSupportVector: [4×1 logical]
                  Solver: 'SMO'
```



Conclusion:

In the above experiment we implement a MATLAB code for Support vector machine. I learnt Kernel function of Radial basis function Because support vector machines employing the kernel trick do not scale well to large numbers of training samples or large numbers of features in the input space, several approximations to the RBF kernel (and similar kernels) have been introduced.

We used fitcsvm to perform the SVM function, and also plotted the scatter plot using gscatter function. We used legend and axis and a few marker effects for better presentation.

Published with MATLAB® R2020a