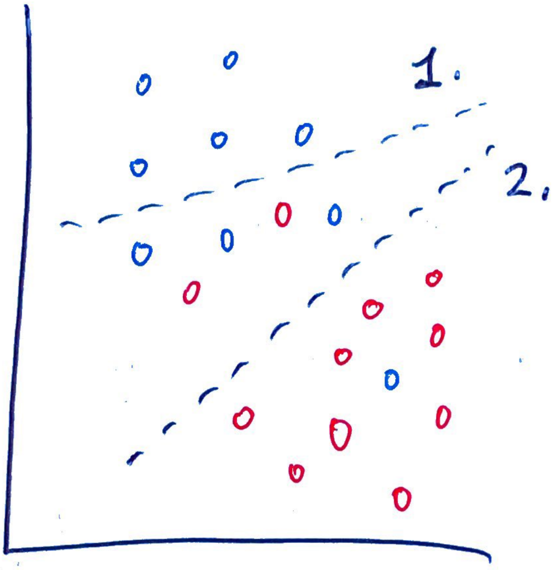
Support Vector Machines



Find the "best" hyperplane that separates the data. (Maximize the boundary) Includes a penalty term for misclassification. Use multiple SVMs for multiple groups.

% Set up

```
>> groupData = readtable("groupData.csv")
>> groupData.group = categorical(...
        groupData.group)
>> cvpt = cvpartition(groupData.group,...
        "HoldOut", 0.35)
>> dataTrain = groupData(training(cvpt),:)
>> dataTest = groupData(test(cvpt),:)
```

% Construct an SVM classifier

fitcecoc

>> mdlSVM = ~~fitcsvm~~(dataTrain, "group")

multiclass SVM → error-correcting output codes

```
>> errSVM = loss(mdlSVM, dataTest)
0.1588
```

Plot the results

```
>> label = predict(mdl, dataTest)

>> gscatter(groupData.x, groupData.y, ...
                    groupData.group)

% now plot the predicted labels on top
  of the original data

>> hold on

>> gscatter(dataTest.x, dataTest.y, ...
          label, [ ], "O", 15)
```

plot predicted labels

default colors

little o's

marker size

Concentric Data

```
% Setup
>> points = readtable("points.csv")

>> cv = cvpartition(points.group, ...
                    "HoldOut", 0.38)

>> trainPoints = points(training(cv),:)

>> testPoints = points(test(cv),:)

>> mdlSVM = fitcsvm(trainPoints, "group")

err = loss(mdlSVM, testPoints)        % 0.2616

% Try Gaussian Kernel function

>> mdlSVM = fitcsvm(trainPoints, "group", ...
       "KernelFunction", "gaussian")
err = loss(mdlSVM, testPoints)        % 0.01
```

Calculate error

```
% training error

errTrain = resubLoss(mdl)

% test error

errTest = loss(mdl, hdTest)
    0.31
```

Change the kernel function to "gaussian"

```
>> mdl = fitcsvm(hdTrain, "HeartDisease", ...
        "KernelFunction", "gaussian")
```

```
errTest = loss(mdl, hdTest)
    0.30
```

Heart Disease (numeric data)

```
% set up
>> heartData = readtable("heartDataNum,csv")
>> heartData.HeartDisease = categorical(...
        heartData,HeartDisease)

>> pt = cvpartition(heartData.HeartDisease,...
        "HoldOut", 0.3)

>> hdTrain = heartData(training(pt),:)
>> hdTest = heartData(test(pt),:)
```

Create an SVM model

```
>> mdl = fitcsvm(hdTrain, "HeartDisease")
```

Heart Disease Analysis

```
% setup

>> heartData = readtable("heartDataAll.csv")

% convert categorical
>> heartData = convertvars(heartData, ...
        12:22, "categorical")

>> pt = cvpartition(heartData, HeartDis...
            "HoldOut", 0.3)

>> hdTrain = heartData(training(pt), :)
>> hdTest = heartData(test(pt), :)
```

SVM

```
>> mdl = fitcsvm(hdTrain, "HeartDisease")
```