

EE583 Pattern Recognition HW3

Kutay Uğurlu 2232841

November 28, 2021

1 Question 1

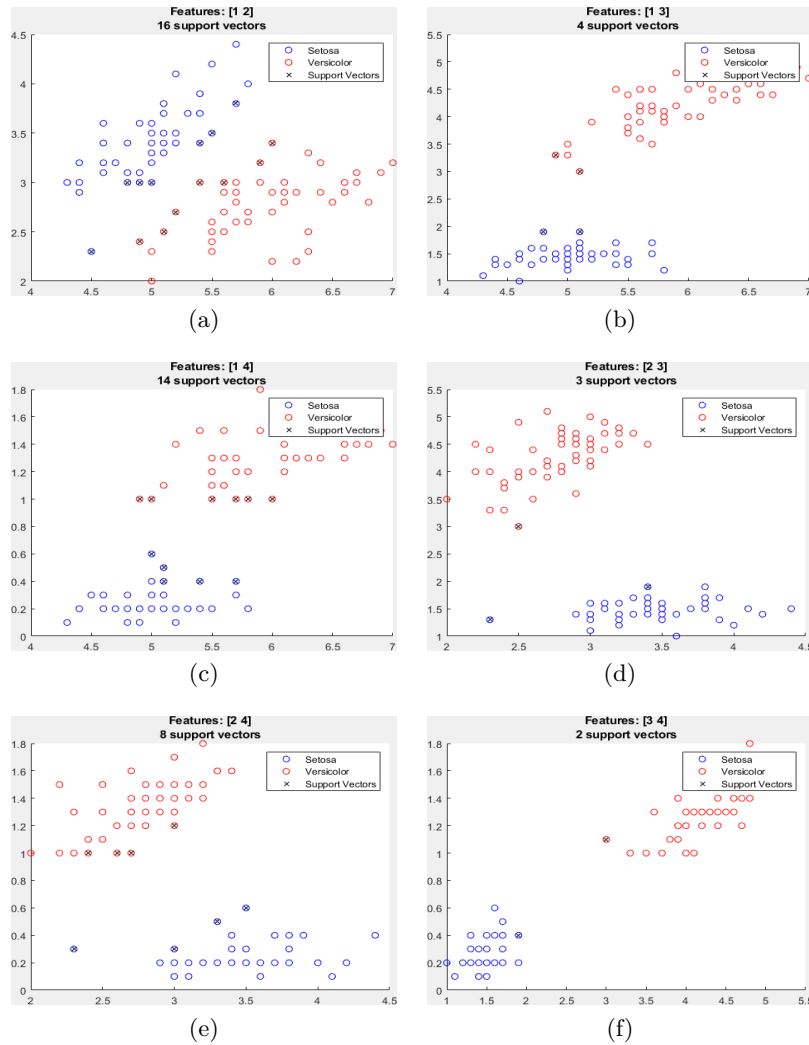


Figure 1: Support Vectors for different feature pairs

The number of support vectors was minimum for the features 3 & 4. One can notice the opposite relationship between the linear separability of the feature space and the number of support vectors. However, it is important to note that one can achieve the same number of minimum vectors for other features by setting the BoxConstraint parameter manually. I observed that setting it to 15 resulted in 2 support vectors for two cases.

2 Question 2

4 Features

kfold loss = 0

LOOCV loss = 0

1 Feature

kfold loss = 0.1900

LOOCV loss = 0.1700

Using four features, having higher dimensions, have resulted in SVM Model linearly separating the feature space. Hence, for both Cross Validation Experiments, the loss rate was 0. On the other hand, when I decreased the number of utilized dimensions to 1, LOOCV loss was slightly less than the 10-Fold CV loss. This may be attributed to the fact that LOOCV utilizes more training data than kfold, resulting in higher more accurate test set predictions, considering that the number of training data is not so large that it causes overfitting.

3 Question 3

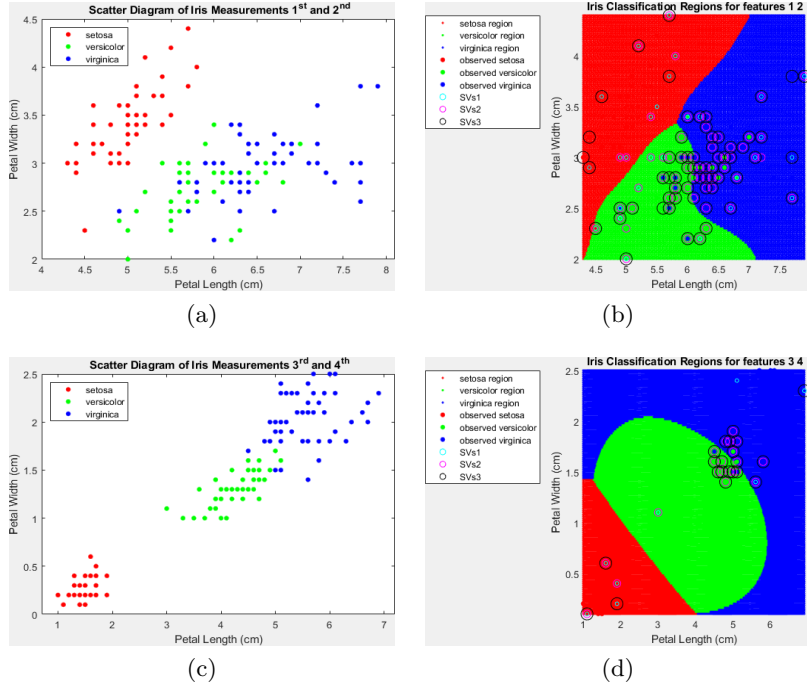


Figure 2: Support Vectors for different feature pairs

Changing feature pairs that are utilized in SVM classification resulted in different distributions. Features 3 and 4 provides a more linearly separable feature space than 1 and 2. The resultant numbers of support vectors 196 and 58 also indicates the same result. In addition, this is also observable in scatter plots.


4 Question 4

The objective function, as a function of BoxConstraint and KernelScale, is optimized using Bayesian optimization. Hence, these parameters determines the cross validation loss and selecting them is a crucial part of optimizing an SVM classifier, keeping in mind that rbf as kernel function is fixed. In addition, I have tried polynomial kernel function, and observed that optimized cross validation loss is different from the one that rbf kernel function is able to achieve. Therefore,

- Box Constraint
- Kernel Scale
- Kernel Function

are the hyperparameters that can be optimized in an SVM to achieve better cross validation success, in terms of loss.

5 APPENDIX

The code given in this section is shared @.

5.1 Q1

```

1 %% HW3_Q1
2 load fisheriris
3 inds = ~strcmp(species, 'virginica');
4
5 features = 1:4;
6 all_feature_pairs = nchoosek(features, 2);
7 counter = zeros(size(all_feature_pairs, 1), 1);
8
9 for i = 1:size(all_feature_pairs, 1)
10     temp_features = all_feature_pairs(i, :);
11     X = meas(inds, temp_features);
12     y = species(inds);
13     setosa_idx = strcmp(y, 'setosa');
14     versicolor_idx = strcmp(y, 'versicolor');
15     SVMModel = fitcsvm(X, y);
16     % n_support_vectors = sum(SVMModel.IsSupportVector);
17     n_support_vectors = size(SVMModel.SupportVectors, 1);
18     idx = SVMModel.IsSupportVector;
19     figure
20     scatter(X(setosa_idx, 1), X(setosa_idx, 2), 'bo')
21     hold on
22     scatter(X(versicolor_idx, 1), X(versicolor_idx, 2), 'ro')
23     hold on
24     scatter(X(idx, 1), X(idx, 2), 'kx')
25     title(['Features: ', mat2str(temp_features)], [num2str(
26         n_support_vectors), ' support vectors'])
27     counter(i) = n_support_vectors;
28     legend('Setosa', 'Versicolor', 'Support Vectors')
29 end
30 min_support_vectors = min(counter);
31 min_SV_features = all_feature_pairs(find(counter ==
32     min_support_vectors), :);
33
34 %% Save all
35 chdir('..')
36 addpath('export_fig')
37 chdir('HW3')

```

```
38 figHandles = findall(0, 'Type', 'figure');
39
40 for i = 1:numel(figHandles)
41     export_fig(['Q1_', num2str(i)], '-png', figHandles(i), '-
        append')
42 end
43
44 close all
```

5.2 Q2

```

1 %% HW2 Q2
2 %% All features
3 load fisheriris
4 inds = ~strcmp(species, 'virginica');
5 features = 1:4;
6 X = meas(inds, features);
7 Y = species(inds);
8 SVMModel = fitcsvm(X, Y, 'Standardize', true, 'KernelFunction', 'RBF'
    , ...
    'KernelScale', 'auto');
9
10
11 CVSVMModel = crossval(SVMModel, 'Kfold', 10, 'Leaveout', 'off');
12 kfold_loss = kfoldLoss(CVSVMModel)
13
14 SVMModel = fitcsvm(X, Y, 'Standardize', true, 'KernelFunction', 'RBF'
    , ...
    'KernelScale', 'auto');
15
16
17 CVSVMModel_xval = crossval(SVMModel, 'Leaveout', 'on');
18 leave_one_out_loss = kfoldLoss(CVSVMModel_xval)
19
20 %% Just 2nd Feature
21 inds = ~strcmp(species, 'virginica');
22 features = 2;
23 X = meas(inds, features);
24 Y = species(inds);
25 SVMModel = fitcsvm(X, Y, 'Standardize', true, 'KernelFunction', 'RBF'
    , ...
    'KernelScale', 'auto');
26
27
28 CVSVMModel = crossval(SVMModel, 'Kfold', 10, 'Leaveout', 'off');
29 kfold_loss = kfoldLoss(CVSVMModel)
30
31 SVMModel = fitcsvm(X, Y, 'Standardize', true, 'KernelFunction', 'RBF'
    , ...
    'KernelScale', 'auto');
32
33
34 CVSVMModel_xval = crossval(SVMModel, 'Leaveout', 'on');
35 leave_one_out_loss = kfoldLoss(CVSVMModel_xval)

```

5.3 Q3

```

1 %% Same features
2
3 load fisheriris
4 X = meas(:,1:2);
5 Y = species;
6
7 figure('Position',[250 250 600 400]);
8 gscatter(X(:,1),X(:,2),Y);
9 h = gca;
10 lims = [h.XLim h.YLim]; % Extract the x and y axis limits
11 title('\bf Scatter Diagram of Iris Measurements 1^{st} and 2^{nd}
    {}');
12 xlabel('Petal Length (cm)');
13 ylabel('Petal Width (cm)');
14 legend('Location','Northwest');
15
16 SVMModels = cell(3,1);
17 classes = unique(Y);
18 rng(1); % For reproducibility
19
20 for j = 1:numel(classes)
21     indx = strcmp(Y,classes(j)); % Create binary classes for each
        classifier
22     SVMModels{j} = fitsvm(X,indx,'ClassNames',[false true],'
        Standardize',true,...
        'KernelFunction','rbf','BoxConstraint',1);
23
24 end
25
26 d = 0.02;
27 [x1Grid,x2Grid] = meshgrid(min(X(:,1)):d:max(X(:,1)),...
        min(X(:,2)):d:max(X(:,2)) );
28
29 xGrid = [x1Grid(:),x2Grid(:)];
30 N = size(xGrid,1);
31 Scores = zeros(N,numel(classes));
32
33 for j = 1:numel(classes)
34     [~,score] = predict(SVMModels{j},xGrid);
35     Scores(:,j) = score(:,2); % Second column contains positive-
        class scores
36
37 end
38 [~,maxScore] = max(Scores,[],2);
39

```



```

40 figure('Position',[250 250 600 400]);
41 h(1:3) = gscatter(xGrid(:,1),xGrid(:,2),maxScore,...
42     [1 0 0; 0 1 0; 0 0 1]);
43 hold on
44 h(4:6) = gscatter(X(:,1),X(:,2),Y);
45
46 title('\bf Iris Classification Regions for features 1 2');
47 xlabel('Petal Length (cm)');
48 ylabel('Petal Width (cm)');
49
50 hold on
51
52 colors = [0 1 1; 1 0 1; 0 0 0];
53 sizes = [15,60,150];
54 for j = 1:3
55 SVM_SV = X(SVMModels{j}.IsSupportVector,:);
56 h(j+6) = scatter(SVM_SV(:,1),SVM_SV(:,2),'CData',repmat(colors(j
57    ,:),size(SVM_SV,1),1),'SizeData',sizes(j)*ones(size(SVM_SV,1)
58     ,1));
59 hold on
60 sum(SVMModels{j}.IsSupportVector)
61 end
62 legend(h,{ 'setosa region','versicolor region','virginica region'
63     ,...
64     'observed setosa','observed versicolor','observed virginica'
65     ,...
66     'SVs1','SVs2','SVs3'},...
67     'Location','NorthwestOutside')
68
69 axis tight
70 hold on
71
72 %% Other features
73
74 load fisheriris
75 X = meas(:,3:4);
76 Y = species;
77
78 figure('Position',[250 250 600 400]);
79 gscatter(X(:,1),X(:,2),Y);
80 h = gca;
81 lims = [h.XLim h.YLim]; % Extract the x and y axis limits
82 title('\bf Scatter Diagram of Iris Measurements 3^{rd} and 4^{th}
83     }}');

```

```

80 xlabel('Petal Length (cm)');
81 ylabel('Petal Width (cm)');
82 legend('Location','Northwest');
83
84 SVMModels = cell(3,1);
85 classes = unique(Y);
86 rng(1); % For reproducibility
87
88 for j = 1:numel(classes)
89     indx = strcmp(Y,classes(j)); % Create binary classes for each
        classifier
90     SVMModels{j} = fitcsvm(X,indx,'ClassNames',[false true],'
        Standardize',true,...
91         'KernelFunction','rbf','BoxConstraint',1);
92     sum(SVMModels{j}.IsSupportVector)
93 end
94
95 d = 0.02;
96 [x1Grid,x2Grid] = meshgrid(min(X(:,1)):d:max(X(:,1)),...
97     min(X(:,2)):d:max(X(:,2)));
98 xGrid = [x1Grid(:),x2Grid(:)];
99 N = size(xGrid,1);
100 Scores = zeros(N,numel(classes));
101
102 for j = 1:numel(classes)
103     [~,score] = predict(SVMModels{j},xGrid);
104     Scores(:,j) = score(:,2); % Second column contains positive-
        class scores
105 end
106
107 [~,maxScore] = max(Scores,[],2);
108
109 figure('Position',[250 250 600 400]);
110
111 h(1:3) = gscatter(xGrid(:,1),xGrid(:,2),maxScore,...
112     [1 0 0; 0 1 0; 0 0 1]);
113 hold on
114 h(4:6) = gscatter(X(:,1),X(:,2),Y);
115
116 title('\bf Iris Classification Regions for features 3 4');
117 xlabel('Petal Length (cm)');
118 ylabel('Petal Width (cm)');
119
120 colors = [0 1 1; 1 0 1; 0 0 0];
121 sizes = [15,60,150];

```

```
122 for j = 1:3
123 SVM_SV = X(SVMModels{j}.IsSupportVector,:);
124 h(j+6) = scatter(SVM_SV(:,1),SVM_SV(:,2),'CData',repmat(colors(j),:),size(SVM_SV,1),1),'SizeData',sizes(j)*ones(size(SVM_SV,1),1));
125 hold on
126 end
127
128 legend(h,{ 'setosa region','versicolor region','virginica region'
129           'observed setosa','observed versicolor','observed virginica'
130           'SVs1','SVs2','SVs3'},...
131         'Location','NorthwestOutside')
132
133 axis tight
134
135 %% Save all
136 chdir('..')
137 addpath('export_fig')
138 chdir('HW3')
139 figHandles = findall(0,'Type','figure');
140
141 for i = 1:numel(figHandles)
142     export_fig(['Q3_',num2str(i)], '-png', figHandles(i), '-append')
143 end
144
145 close all
```

5.4 Q4

```

1
2 rng default % For reproducibility
3 grnpop = mvnrnd([1,0],eye(2),10);
4 redpop = mvnrnd([0,1],eye(2),10);
5
6 plot(grnpop(:,1),grnpop(:,2),'go')
7 hold on
8 plot(redpop(:,1),redpop(:,2),'ro')
9 hold off
10
11 redpts = zeros(100,2);grnpts = redpts;
12 for i = 1:100
13     grnpts(i,:) = mvnrnd(grnpop(randi(10),:),eye(2)*0.02);
14     redpts(i,:) = mvnrnd(redpop(randi(10),:),eye(2)*0.02);
15 end
16
17 figure
18 plot(grnpts(:,1),grnpts(:,2),'go')
19 hold on
20 plot(redpts(:,1),redpts(:,2),'ro')
21 hold off
22
23 cdata = [grnpts;redpts];
24 grp = ones(200,1);
25 % Green label 1, red label -1
26 grp(101:200) = -1;
27
28 c = cvpartition(200,'Kfold',10);
29
30
31 opts = struct('Optimizer','bayesopt','ShowPlots',true,'
    CVPartition',c,...
32     'AcquisitionFunctionName','expected-improvement-plus');
33 svmmod = fitcsvm(cdata,grp,'KernelFunction','rbf',...
34     'OptimizeHyperparameters','auto','
    HyperparameterOptimizationOptions',opts)
35
36 lossnew = kfoldLoss(fitcsvm(cdata,grp,'CVPartition',c,'
    KernelFunction','rbf',...
37     'BoxConstraint',svmmod.HyperparameterOptimizationResults.
    XAtMinObjective.BoxConstraint,...
38     'KernelScale',svmmod.HyperparameterOptimizationResults.
    XAtMinObjective.KernelScale))

```

```
39
40 d = 0.02;
41 [x1Grid,x2Grid] = meshgrid(min(cdata(:,1)):d:max(cdata(:,1)),...
42     min(cdata(:,2)):d:max(cdata(:,2)));
43 xGrid = [x1Grid(:),x2Grid(:)];
44 [~,scores] = predict(svmmod,xGrid);
45 figure;
46 h = nan(3,1); % Preallocation
47 h(1:2) = gscatter(cdata(:,1),cdata(:,2),grp,'rg','+*');
48 hold on
49 h(3) = plot(cdata(svmmod.IsSupportVector,1),...
50     cdata(svmmod.IsSupportVector,2),'ko');
51 contour(x1Grid,x2Grid,reshape(scores(:,2),size(x1Grid)),[0 0],'k'
52     );
53 legend(h',{'-1','+1','Support Vectors'},'Location','Southeast');
54 axis equal
55 hold off
```