Arda Cankat BATI
21302578
Sec-2

CS-464 Machine Learning HW#1

Question 1-1

Machine 1		Machine 2		
65 S	+	230 S	=	295
135 F	+	900 F	=	1035

$$P(M_1|S) = \frac{P(M_1 \cap S)}{P(S)} = \frac{\frac{\#\text{Successes in } M_1}{\#\text{All games}}}{\frac{\#\text{Successes}}{\#\text{All games}}} = \frac{\frac{65}{1330}}{\frac{295}{1330}} = \frac{65}{295}$$

Q. 1-2

$P(S \mid M_1, Me) = \frac{40}{100}$

$P(S \mid M_1, Friend) = \frac{25}{100}$

M_1, I am more likely to win

$P(S \mid M_2, Me) = \frac{212}{1040}$

$P(S \mid M_2, Friend) = \frac{18}{90}$

$\frac{212}{1040} > \frac{18}{90}$, M_2 I am more likely to win

Q 1-3

Total wins $= 295$

Total Losses $= 1035$

$P(S \mid Me) = \frac{252}{1040} \simeq 0.221$

$P(S \mid Friend) = \frac{43}{190} \simeq 0.226$

Friend is more likely to win

Q. 2-1

$C = \{b \neq 6 \cap r \neq 1\}$, $b = \{1,2,3,4,5\}$, $r = \{2,3,4,5,6\}$ for this case

$$P(b=1, r=3 \mid C) = \underbrace{P(b=1 \mid C)\, P(r=3 \mid C)}_{\text{Conditional Independence}} = \frac{1}{\binom{5}{1}} \cdot \frac{1}{\binom{5}{1}} = \frac{1}{25}$$

Q. 2-2

$D = \{b+r = \text{even}\}$, $P(b=3, c=5 \mid D) = \dfrac{1 \text{ case}}{\# D\text{'s possibilities}} = \dfrac{1}{18}$

$b+r = \text{even}$ has $\binom{6}{1}\binom{3}{1} = 18$ possibilities

Q 2-3

- Info C creates conditional independence between the events

$$P(b=1, r=3 \mid C) = P(b=1 \mid C)\, P(r=3 \mid C)$$

- Info D leads to conditional dependence between the events

$$P(b=3, c=5 \mid D) \neq P(b=3 \mid D) \cdot P(c=5 \mid D)$$

$$\frac{1}{18} \neq \frac{1}{6} \cdot \frac{1}{6}$$

Question 3

$$P(\lambda \mid x = \{x_1, \dots, x_n\}) \propto \underbrace{P(x = \{\dots\} \mid \lambda)}_{\text{Likelihood}} \underbrace{P(\lambda)}_{\text{prior}}$$

$$\lambda^{new} = \underset{\lambda}{\operatorname{argmax}} \left(P(x = \{x_1, \dots, x_n\} \mid \lambda)\, P(\lambda) \right)$$

$$\lambda^{new} = \underset{\lambda}{\operatorname{argmax}} \left(\prod_{i=1}^{n} \frac{\lambda^{x_i} e^{\lambda}}{x_i!} \right) \cdot \frac{1}{\sqrt{2\pi\beta^2}}\, e^{\frac{-\lambda^2}{2\beta^2}}$$

!!! (Question 3 continues in the final page!)

Question 4-1

The denominator $\sum_k P(Y = y_k) \prod_{J=1}^{V} P(X_J \mid Y = y_k)^{t_{WJ,i}} = P(D_i)$

$P(D_i)$ is the probability of ith mail. It is a positive constant so it doesn't effect our $\underset{y_k}{\operatorname{argmax}}(\dots)$ expression. $\left(\underset{y_k}{\operatorname{argmax}}(F(x)) = \underset{y_k}{\operatorname{argmax}}\left(\frac{F(x)}{A}\right) \right)$

A = constant

4.2 The percentages of spam and valid mails are both 50%.
Therefore, the dataset is very balanced.

4.3

The values were reported in the command window as:

Number of errors in prediction: 130

Accuracy of prediction: 0.5 → 50%

- The dataset contains many zeros. Their corresponding MLE estimates cause overfitting of the classifier. $Log(0) = -\infty$ terms completely dominate the sum. Therefore, all the situations are ties between ham/spam. We always set the decision to non-spam in case of ties. So the classifier always predicts non-spam mail

4.4

Prediction errors = 7, Accuracy = 0.9731.

Additive smoothing prevented $Log(0) = -\infty$ cases and yielded a much more accurate result.

4.5

Top 10 Scores	0.4339	0.2719	0.2455	0.2235	0.2137	0.1907	0.1576	0.1550	0.1355	0.1338
Top 10's Indices	4	8	55	15	7	20	45	131	132	98

4.6

Features are removed one-by-one from the least informative one.
The resulting accuracy is given as a plot.

Question 5.1 The estimated parameters are shown in the command window. Estimated averages and variances are given seperately in two tables.

Question 5.2

The confusion matrices were created in two seperate cases. In the first case train and test data were used in order. In the second case train and test data were swapped.

The tables are given as follows

Case 1 : Table 1: Shows confusion matrix for all classes
Table 2: " " " for class 1
Table 3: " " " for class 2
Table 4: " " " for class 3
Case 2: Tables 5 to 8, same as above

Question 3 Continued:

$$\lambda^{new} = \underset{\lambda}{\operatorname{argmax}} \left(\prod_{i=1}^{n} \frac{\lambda^{x_i}}{x_i!} \right) \cdot \frac{e^{\lambda n}}{\sqrt{2\pi\beta^2}} e^{\frac{-\lambda^2}{2\beta^2}}$$

log

$$\lambda^{new} = \underset{\lambda}{\operatorname{argmax}} \left(-n\lambda - \frac{\lambda^2}{2\beta^2} - \ln(\sqrt{2\pi\beta^2}) + \ln(\lambda) \sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \ln(x_i!) \right)$$

d/dλ

$$\text{derivative} = -n - \frac{2\lambda}{2\beta^2} + \frac{\left(\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \ln(x_i!)\right)}{\lambda} = 0$$

$$-n\beta^2 - \lambda + \frac{(\dots)\beta^2}{\lambda} = 0$$

$$\lambda = -n\beta^2 + \frac{(\dots)\beta^2}{\lambda}$$

$$\lambda_{max} = \beta^2 \left(\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} \ln(x_i!) - n \right)$$

```matlab
function main(Q4trainFeatures, Q4testFeatures, Q4trainLabels,
 Q4testLabels, Q5trainfeatures, Q5testFeatrues )

% ------------------------------------------------

% *****  QUESTION 4-3 ***********************

trainData = dlmread(Q4trainFeatures);
testData = dlmread(Q4testFeatures);
trainLabels = dlmread(Q4trainLabels,' ');
testLabels = dlmread(Q4testLabels,' ');


hamWordsSum = sum(trainData((trainLabels == 0),:),1);
spamWordsSum = sum(trainData((trainLabels > 0)),1);
hamWordsTotal =  sum(sum(trainData((trainLabels == 0),:)));
spamWordsTotal =  sum(sum(trainData((trainLabels > 0))));
Likelihood_ham = hamWordsSum/hamWordsTotal;
Likelihood_spam = spamWordsSum/spamWordsTotal;
PriorHam = sum((trainLabels == 0))./size(trainLabels,1);
PriorSpam = sum((trainLabels > 0))./size(trainLabels,1);

logR = log2(Likelihood_ham);
logR2 = log2(Likelihood_spam);

for i = 1:260
    P3_resultHam(i,1:2500) = testData(1,:).*logR;
    P3_resultSpam(i,1:2500) = testData(i,:).*logR2;
end

P3_resultHam(isnan(P3_resultHam)) = 0;
P3_resultSpam(isnan(P3_resultSpam)) = 0;

P3_resultHam = sum(P3_resultHam,2) + log2(PriorHam);
P3_resultSpam = sum(P3_resultSpam,2) + log2(PriorSpam);

predictedClass(P3_resultHam == P3_resultSpam) = 0;
predictedClass(P3_resultHam > P3_resultSpam) = 0;
predictedClass(P3_resultHam < P3_resultSpam) = 1;

realClass(1:130) = 0;
realClass(131:260) = 1;

errorT = realClass - predictedClass;
error = abs(realClass - predictedClass);

noErrors1 = sum(error);
disp('******** The code ends - Result displays begin here! ********');
fprintf('\n');
disp('******** QUESTION 4-3 ********');
fprintf('\n');
```

```
fprintf('Number of errors in the prediction of test data is:  %.0f
 \n', noErrors1);

accuracy1 = 1 - noErrors1/size(predictedClass,2);
fprintf('Accuracy in the prediction of test data is:  %.2f \n',
 accuracy1);
fprintf('In percentage, accuracy =  %.2f', accuracy1*100);
disp(' %');
fprintf('\n');


% ---------------------------------------------------

% *****  QUESTION 4-4 ***********************

alpha = 1;
hamWordsSum = sum(trainData((trainLabels == 0),:),1);
spamWordsSum = sum(trainData(trainLabels > 0,:),1);
totalWordHam =  sum(sum(trainData((trainLabels == 0),:)));
totalWordSpam=  sum(sum(trainData(trainLabels > 0,:)));
Likelihood_ham = (hamWordsSum + alpha)/(totalWordHam +
 alpha*size(trainData,1));
Likelihood_spam = (spamWordsSum + alpha)/(totalWordSpam +
 alpha*size(trainData,1));

logR = log(Likelihood_ham);
logR2 = log(Likelihood_spam);

for i = 1:260
    resultHam(i,1:2500) = testData(i,:).*logR;
    resultSpam(i,1:2500) = testData(i,:).*logR2;
end

resultHam(isnan(resultHam)) = 0;
resultSpam(isnan(resultSpam)) = 0;

resultHam = sum(resultHam,2) + log2(PriorHam);
resultSpam = sum(resultSpam,2) + log2(PriorSpam);

predictedClass(resultHam == resultSpam) = 0;
predictedClass(resultHam > resultSpam) = 0;
predictedClass(resultHam < resultSpam) = 1;

realClass(1:130) = 0;
realClass(131:260) = 1;

errorT = realClass - predictedClass;
error = abs(realClass - predictedClass);
noErrors2 = sum(error);
accuracy2 = 1 - noErrors2/size(predictedClass,2);

fprintf('\n');
disp('******** QUESTION 4-4 ********');
fprintf('\n');
```

```
fprintf('Number of errors in the prediction of test data is:  %.2f
 \n', noErrors2);
fprintf('Accuracy in the prediction of test data is:  %.4f \n',
 accuracy2);
fprintf('\n');


% ------------------------------------------------

% *****  QUESTION 4-5 **********************

ham = trainData((trainLabels == 0),:);
N10 = sum(ham > 0); %N10   %contains word t and class = 0
N00 = 350 - N10; %N00   %doesn't contain word t and class = 0

spam = trainData((trainLabels > 0),:);
N11 = sum(spam > 0); %N11  %contains word t and class = 1
N01 = 350 - N11; %N01   %doesn't contain word t and class = 1

N = 700;
M(1,:) = (N11/N).*log2((N*N11)./((N11+N10).*(N11+N01)));
M(2,:) = (N01/N).*log2((N*N01)./((N01+N00).*(N11+N01)));
M(3,:) = (N10/N).*log2((N*N10)./((N11+N10).*(N10+N00)));
M(4,:) = (N00/N).*log2((N*N00)./((N00+N01).*(N00+N10)));
M = sum(M);

M(isnan(M))= 0;
[sortedM,sortingIndices] = sort(M,'descend');

top10Values = sortedM(1:10);
top10Indices = sortingIndices(1:10);

fprintf('\n');
disp('******** QUESTION 4-5 ********');
fprintf('\n');

fprintf('Top 10 values:');
disp(top10Values);

fprintf('Top 10s indices:');
disp(top10Indices);

% ------------------------------------------------

% *****  QUESTION 4-6 **********************

PriorHam = sum((trainLabels == 0))./size(trainLabels,1);
PriorSpam = sum((trainLabels > 0))./size(trainLabels,1);

spam = trainData(351:700,1:2500);
spam = spam > 0;
N_11 = sum(spam);
N_01 = 350 - N_11;
```

```
ham = trainData(1:350,1:2500);
ham = ham > 0;
N_10  = sum(ham);
N_00 = 350 - N_10 ;

N = 700;
Mutual_Info= (N_11)/N.*log2((N*N_11)./((N_11+N_10).*(N_11+N_01))) ...
+ (N_01/N).*log2((N*N_01)./((N_01+N_00).*(N_11+N_01))) ...
+ (N_10/N).*log2((N*N_10)./((N_10+N_11).*(N_10+N_00))) ...
+ (N_00/N).*log2((N*N_00)./((N_00+N_01).*(N_00+N_10))) ;

Mutual_Info(isnan(Mutual_Info))=0;

[sorted_Mutual_Info, sorted_indices]=sort(Mutual_Info,'descend');

trainData(701,:)=sorted_Mutual_Info;
trainData=sortrows(trainData' , 701);
trainData(:,701) = [];
testData(261,:)=sorted_Mutual_Info;
testData=sortrows(testData',261);
testData(:,261) = [];

realClass(1:130) = 0;
realClass(131:260) = 1;
trainData = trainData';
testData = testData';
alpha = 1;

for i=1:2499

    trainData(:,1)=[];
    testData(:,1)=[];

    hamWordsSum = sum(trainData(1:350,1:end));
    spamWordsSum = sum(trainData(351:700,1:end));
    hamWordsTotal = sum(sum(trainData(1:350,1:end)));
    spamWordsTotal =  sum(sum(trainData(351:700,1:end)));

    Likelihood_ham = (hamWordsSum + alpha)/(hamWordsTotal +
 alpha*size(trainData,1));
    Likelihood_spam = (spamWordsSum + alpha)/(spamWordsTotal +
 alpha*size(trainData,1));

    resultSpam = log(PriorSpam) +
 sum(testData(:,:).*log(Likelihood_spam),2);
    resultHam = log(PriorHam) +
 sum(testData(:,:).*log(Likelihood_ham),2);
    resultHam(isnan(resultHam)) = 0;
    resultSpam(isnan(resultSpam)) = 0;

    resultHam = sum(resultHam,2);
    resultSpam = sum(resultSpam,2);

    predictedClass(resultHam > resultSpam) = 0;
```

```
    predictedClass(resultHam == resultSpam) = 0;
    predictedClass(resultHam < resultSpam) = 1;

    error = abs(realClass - predictedClass);
    noErrors = length(error(error==1));
    accuracy(i) = (260-noErrors)/260;

end

plot(1:2499,accuracy);

fprintf('\n');
disp('******** QUESTION 4-6 ********');
fprintf('\n');
disp('Plot is given.');
title('QUESTION 4-6 Removed Features vs Accuracy ')
xlabel('Number of removed features')
ylabel('Accuracy')

% ----------------------------------------------

% *****  QUESTION 5-1 ***********************

fprintf('\n');
disp('******** QUESTION 5-1 ********');
fprintf('\n');

X = dlmread(Q5trainfeatures,',');
[class0, average, variance] = runDatasets(X, []);

Priors = [1/3 1/3 1/3]; %All 3 classes are equally likely

disp('Estimated averages of train data for different class labels.')
T1 = array2table(average,'VariableNames',
{'Class_1','Class_2','Class_3'},'RowNames',
{'Feature1_Ave','Feature2_ave','Feature3_ave','Feature4_ave','Feature5_ave'});
fprintf('\n');
disp(T1);
disp('*****************************************');
fprintf('\n');


disp('Estimated variances of train data for different class labels.');
T2 = array2table(variance,'VariableNames',
{'Class_1','Class_2','Class_3'},'RowNames',
{'Feature1_Var','Feature2_Var','Feature3_Var','Feature4_Var','Feature5_Var'});
fprintf('\n');
disp(T2);
disp('*****************************************');
fprintf('\n');


% ----------------------------------------------
```

```
% *****  QUESTION 5-2 ***********************

fprintf('\n');
disp('******** QUESTION 5-2 ********');
fprintf('\n');

% DATASETS IN GIVEN ORDER

X = dlmread(Q5trainfeatures,',');
testData = dlmread(Q5testFeatrues,',');

[class1] = runDatasets(X, testData);
createTables(class1, '(Datasets in given order)');

% DATASETS SWAPPED

class2 = runDatasets(testData, X);
createTables(class2,'(Datasets swapped) ');


% ***************  FUNCTIONS  *****************

function [class, average, variance] = runDatasets(train, test)

X = train;
testData  = test;

j = 1:1500;
i = 1:5;
k = 1:3;

index(:,1) = X(j,6) == 1;
index(:,2) = X(j,6) == 2;
index(:,3) = X(j,6) == 3;

S(1,:) = sum(X(j,i).*index(:,1));
S(2,:) = sum(X(j,i).*index(:,2));
S(3,:) = sum(X(j,i).*index(:,3));

average = (1/500).*S'; %i by k, average of ith feature for kth class
ave = (1/500).*S;

V(:,1) = sum(((X(j,i)-ave(1,i)).^2).*index(:,1));
V(:,2) = sum(((X(j,i)-ave(2,i)).^2).*index(:,2));
V(:,3) = sum(((X(j,i)-ave(3,i)).^2).*index(:,3));

variance = V/500;

if (size(test) == 0)
    class = 0;
    return;
end

Priors = [1/3 1/3 1/3]; %All 3 classes are equally likely
```

```
for i = 1:5
    for k = 1:3
        NormalDist1(i,k,1:1500) = (1./
(sqrt(2*pi*variance(i,k)))).*exp(-((testData(1:1500,i) -
 average(i,k)).^2)./(2*variance(i,k)));
    end
end

for k = 1:3
    logSum1(:,k,:) = log(1/3) + sum(log(NormalDist1(:,k,:)));
end

logSum1(logSum1 == -inf)= 0;
[Y,class] = max(logSum1,[],2);

end



function createTables(class, string)

    %Confusion matrice,each entry represents number of times a certain
 real class - prediction combination occured
    Total_Confusion_Matrix = zeros(3,3);
    Total_Confusion_Matrix(1,1) = sum(class(1:500) == 1);
    Total_Confusion_Matrix(1,2) = sum(class(1:500) == 2);
    Total_Confusion_Matrix(1,3) = sum(class(1:500) == 3);
    Total_Confusion_Matrix(2,1) = sum(class(501:1000) == 1);
    Total_Confusion_Matrix(2,2) = sum(class(501:1000) == 2);
    Total_Confusion_Matrix(2,3) = sum(class(501:1000) == 3);
    Total_Confusion_Matrix(3,1) = sum(class(1001:1500) == 1);
    Total_Confusion_Matrix(3,2) = sum(class(1001:1500) == 2);
    Total_Confusion_Matrix(3,3) = sum(class(1001:1500) == 3);

    % Columns are predicted classes, rows are real classes
    T1 = array2table(Total_Confusion_Matrix,'VariableNames',
{'Predicted_Class_1','Predicted_Class_2','Predicted_Class_3'},'RowNames',
{'Actual_Class_1','Actual_Class_2','Actual_Class_3'});
    disp(strcat(string,' confusion table including all classes.'));
    fprintf('\n');
    disp(T1);
    disp('******************************************');
    fprintf('\n');
    %figure();
    %displayTable(T1)

    Table_Class1(1,1) = sum(class(1:500) == 1);
    Table_Class1(1,2) = sum(class(1:500)~= 1);
    Table_Class1(2,1) = sum(class(501:1500) == 1);
    Table_Class1(2,2) = sum(class(501:1500) ~= 1);
    T2 = array2table(Table_Class1,'VariableNames',
{'Predicted_True', 'Predicted_False'},'RowNames',
{'Actual_True', 'Actual_False'});
```

```
    disp(strcat(string,'Class 1 ', ' data confusion table.'));
    fprintf('\n');
    disp(T2);
    disp('*****************************************');
    fprintf('\n');
    %figure();
    %displayTable(T2)


    Table_Class2(1,1) = sum(class(501:1000) == 2);
    Table_Class2(1,2) = sum(class(501:1000)~= 2);
    Table_Class2(2,1) = sum(class(1:500) == 2) + sum(class(1001:1500)
 == 2);
    Table_Class2(2,2) = sum(class(1:500) ~= 2) + sum(class(1001:1500)
 ~= 2);
    T3 = array2table(Table_Class2,'VariableNames',
{'Predicted_True', 'Predicted_False'},'RowNames',
{'Actual_True', 'Actual_False'});
    disp(strcat(string,'Class 2 ', ' data confusion table.'));
    fprintf('\n');
    disp(T3);
    disp('*****************************************');
    fprintf('\n');
    %figure();
    %displayTable(T3)


    Table_Class3(1,1) = sum(class(1001:1500) == 3);
    Table_Class3(1,2) = sum(class(1001:1500)~= 3);
    Table_Class3(2,1) = sum(class(1:1000) == 3);
    Table_Class3(2,2) = sum(class(1:1000) ~= 3);
    T4 = array2table(Table_Class3,'VariableNames',
{'Predicted_True', 'Predicted_False'},'RowNames',
{'Actual_True', 'Actual_False'});
    disp(strcat(string, 'Class 3 ',' data confusion table.'));
    fprintf('\n');
    disp(T4);
    disp('*****************************************');
    fprintf('\n');
    %figure();
    %displayTable(T4)

end


end




%
```

```
******** The code ends - Result displays begin here! ********

******** QUESTION 4-3 ********

Number of errors in the prediction of test data is:  130
Accuracy in the prediction of test data is:  0.50
In percentage, accuracy =  50.00 %


******** QUESTION 4-4 ********

Number of errors in the prediction of test data is:  7.00
Accuracy in the prediction of test data is:  0.9731


******** QUESTION 4-5 ********

Top 10 values:  Columns 1 through 7

    0.4339    0.2719    0.2455    0.2235    0.2137    0.1907    0.1576

  Columns 8 through 10

    0.1550    0.1355    0.1338

Top 10s indices:     4     8    55    15     7    20    45   131   132
    98


******** QUESTION 4-6 ********

Plot is given.

******** QUESTION 5-1 ********

Estimated averages of train data for different class labels.

                    Class_1    Class_2    Class_3
                    _______    _______    _______

    Feature1_Ave    15.804     22.649     9.4639
    Feature2_ave    18.196     29.062     21.327
    Feature3_ave    16.378     27.088     19.211
    Feature4_ave    14.699     21.607     8.4899
    Feature5_ave    13.574     20.395     7.2602

*****************************************

Estimated variances of train data for different class labels.

                    Class_1    Class_2    Class_3
                    _______    _______    _______

    Feature1_Var    40.112     26.232     19.715
```

Feature2_Var	40.016	15.52	21.289
Feature3_Var	46.248	25.484	29.599
Feature4_Var	43.962	29.479	23.634
Feature5_Var	41.377	28.104	20.735

****

******** QUESTION 5-2 ********

(Datasets in given order) confusion table including all classes.

	Predicted_Class_1	Predicted_Class_2	Predicted_Class_3
Actual_Class_1	219	148	133
Actual_Class_2	9	482	9
Actual_Class_3	31	26	443

****

(Datasets in given order)Class 1 data confusion table.

	Predicted_True	Predicted_False
Actual_True	219	281
Actual_False	40	960

****

(Datasets in given order)Class 2 data confusion table.

	Predicted_True	Predicted_False
Actual_True	482	18
Actual_False	174	826

****

(Datasets in given order)Class 3 data confusion table.

	Predicted_True	Predicted_False
Actual_True	443	57
Actual_False	142	858

****

(Datasets swapped) confusion table including all classes.

	Predicted_Class_1	*Predicted_Class_2*	*Predicted_Class_3*
Actual_Class_1	*219*	*148*	*133*
Actual_Class_2	*9*	*482*	*9*
Actual_Class_3	*31*	*26*	*443*

****

(Datasets swapped)Class 1 data confusion table.

	Predicted_True	*Predicted_False*
Actual_True	*219*	*281*
Actual_False	*40*	*960*

****

(Datasets swapped)Class 2 data confusion table.

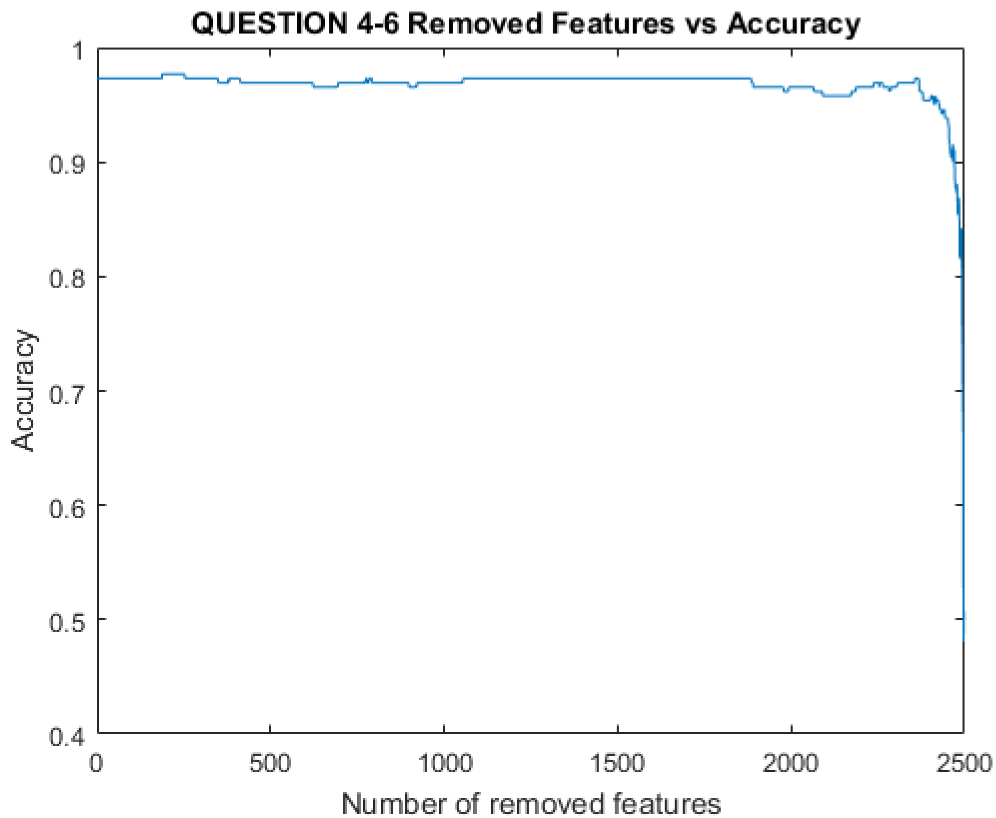
	Predicted_True	*Predicted_False*
Actual_True	*482*	*18*
Actual_False	*174*	*826*

****

(Datasets swapped)Class 3 data confusion table.

	Predicted_True	*Predicted_False*
Actual_True	*443*	*57*
Actual_False	*142*	*858*

****



Published with MATLAB® R2016b