Pattern Classification and Recognition
ECE 681

Spring 2019
Homework #1: Performance Evaluation (ROC Curves)

Due: 5:00 PM, Tuesday, January 29, 2019
Grace Period Concludes: 11:30 PM, Friday, February 1, 2019

This homework assignment is worth **200 points.**
Each problem is worth some multiple of 10 points, and will be scored on the below letter scale.
The letter grades B through D may be modified by + (+3%) and A through D may be modified by a - (-3%).

A+ = 100%: Exceeds expectations, and no issues identified
A = 95%: Meets expectations, and (perhaps) minor/subtle issues
B = 85%: Issues that need to be addressed
C = 75%: Significant issues that must be addressed
D = 65%: Major issues, but with noticeable perceived effort
F = 50%: Major issues, and insufficient perceived effort
Z = 30%: Minimal perceived effort
N = 0%: Missing, or no (or virtually no) perceived effort

Your homework is not considered submitted until both components (**self-contained pdf file** and your code) have been submitted. Please do not include a print-out of your code in the pdf file.

The majority of homework assignments this semester will involve coding. For example, in this homework you will be writing code to generate and plot ROCs. I strongly suggest making use of data structures so the input/output argument lists for your functions cannot become unwieldy. For example, instead of
`[pd,pfa,auc,...] = generateROC(decisionStatistics,truth,thresholds,...)`,
I suggest something like[1] `[roc] = generateROC(classifierOutput,rocOptions)`, where `roc` is a structure with fields `pd`, `pfa`, `auc`, and is extensible to also contain any other information about the ROC, such as the maximum probability of correct decision, `classifierOutput` is a structure with fields `decisionStatistics`, `truth`, and optionally may also contain other information regarding the classifier and its output even if that other information is not needed to generate the ROC, and `rocOptions` is a structure with fields that contain the information necessary to select the thresholds (method for selecting the thresholds and parameter(s) the threshold selection method may need).

I also suggest thinking about how to modularize your code. For example, instead of calculating performance metrics (e.g, AUC, maximum probability of correct decision) within your function that generates the ROC, consider having separate functions (that can be called by the function that generates your ROC) to calculate these performance metrics. If the functions that calculate the performance metrics are separate from the function that generates the ROC, then you will have flexibility to find these metrics for separately from generating an ROC.

[1]Note I am not specifying the syntax for the code you write or use. I am suggesting you think about how you can structure and organize your code so it can be extended to additional use cases "easily".

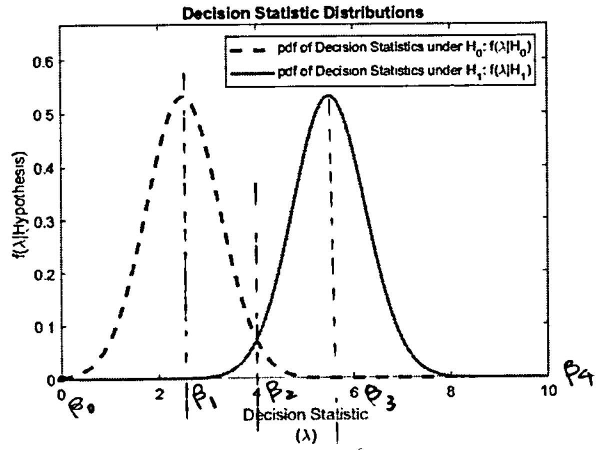
Decision Statistics $\leftrightarrow$ ROCs

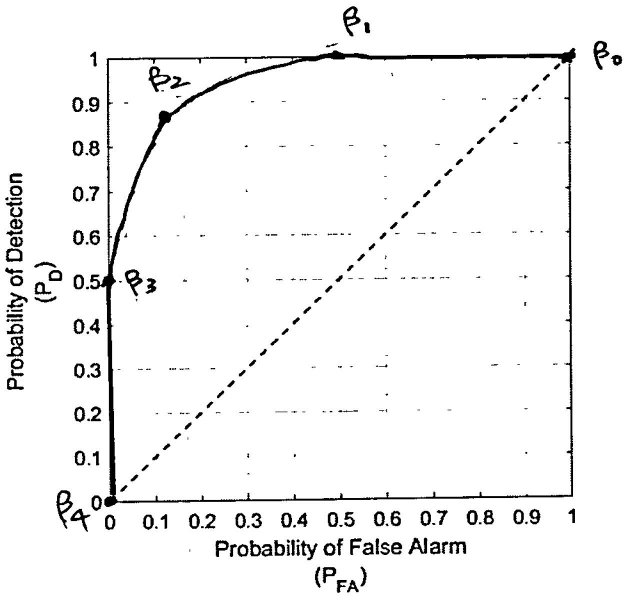
Having intuition for what the shape of the ROC reveals about the distributions of the underlying decision statistics can be quite helpful, as insight regarding the distributions of decision statistics can inform further algorithmic improvements.

(20) 1. For each set of decision statistic distributions given below, sketch (qualitatively, but as accurately as you can) the corresponding ROC. We are not concerned about the precision of the ROC, but rather its general shape and relative position within the ROC axes.

(a)

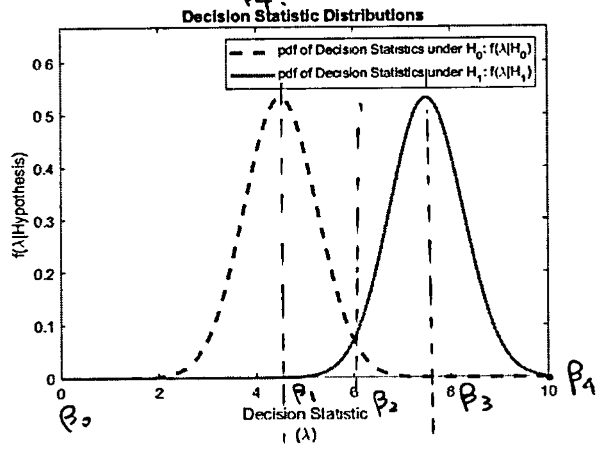
	P_D	P_{FA}
β_0:	1	1
β_1:	1	1/2
β_2:	1/2	0
β_3:	7/8	1/8
β_4:	0	0

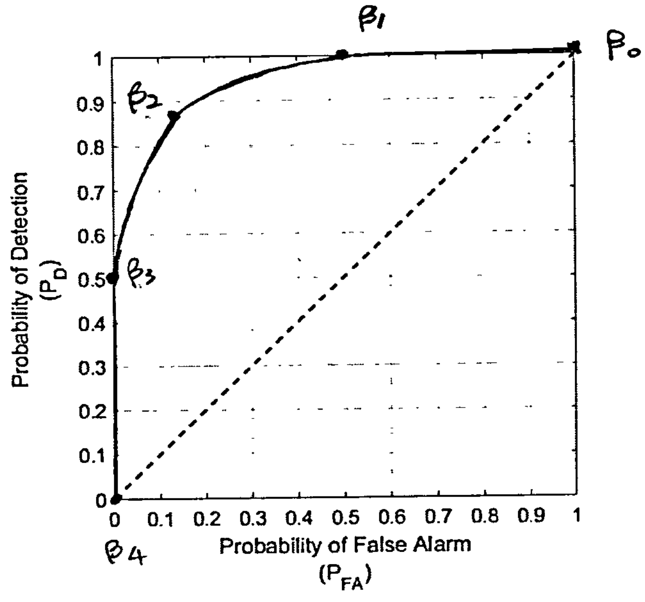




(b)

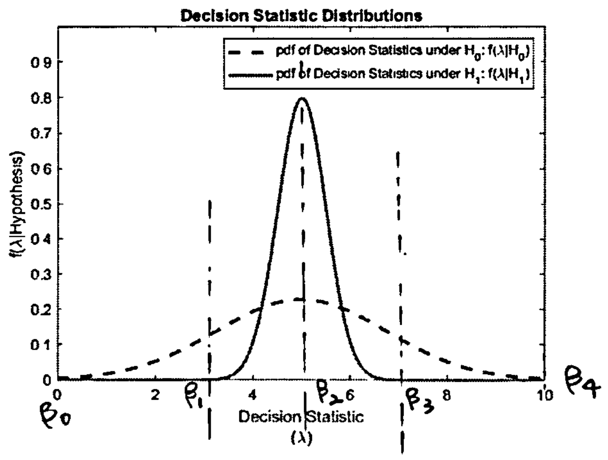
	P_D	P_{FA}
β_0:	1	1
β_1:	1	1/2
β_2:	7/8	1/8
β_3:	1/2	0
β_4:	0	0

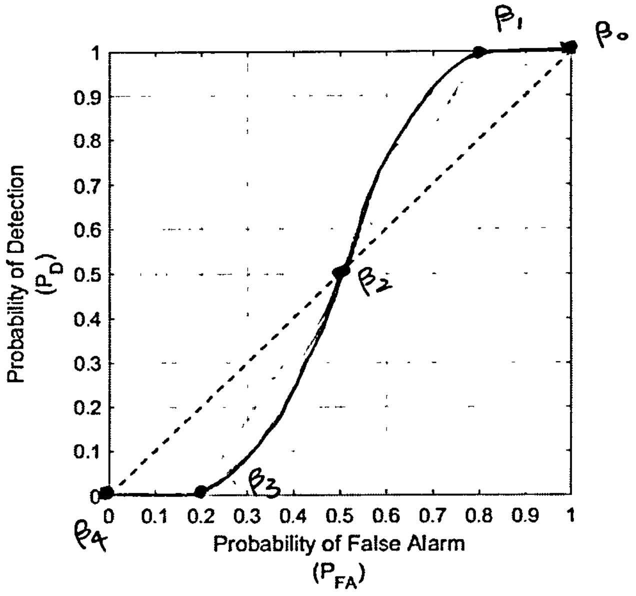




(c)

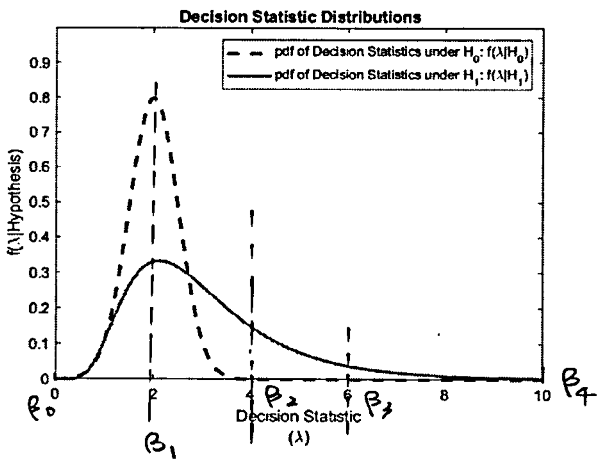
	P_D	P_{FA}
β_0 :	1	1
β_1 :	1	4/5
β_2 :	1/2	1/2
β_3 :	0	1/5
β_4 :	0	0

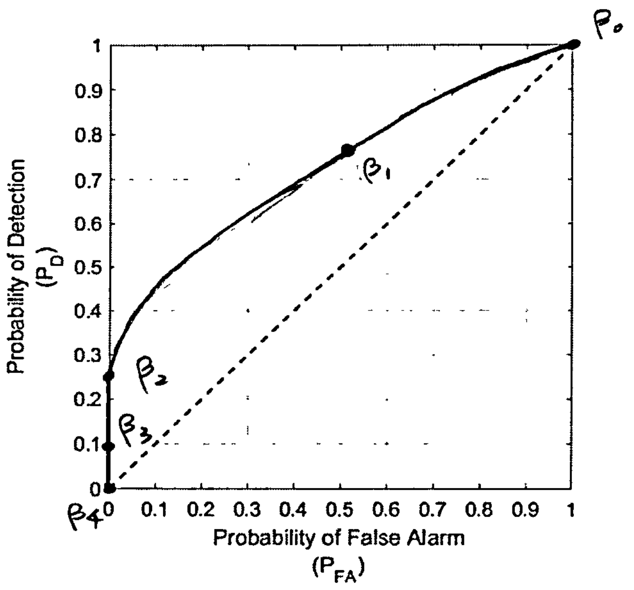




(d)

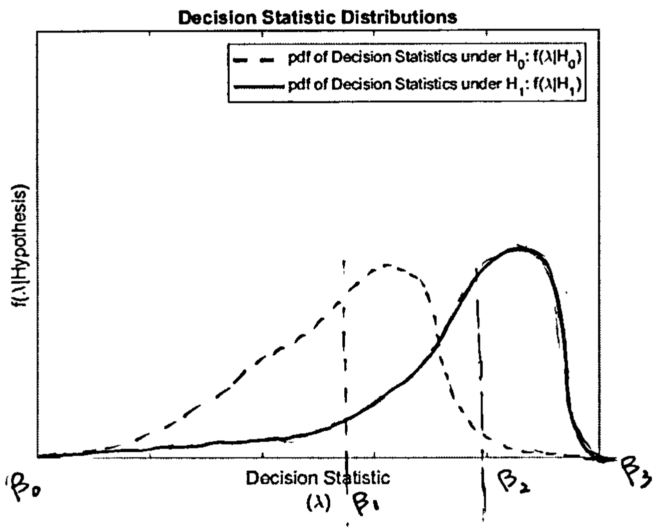
	P_D	P_{FA}
β_0 :	1	1
β_1 :	3/4	1/2
β_2 :	1/4	0
β_3 :	1/10	0
β_4 :	0	0

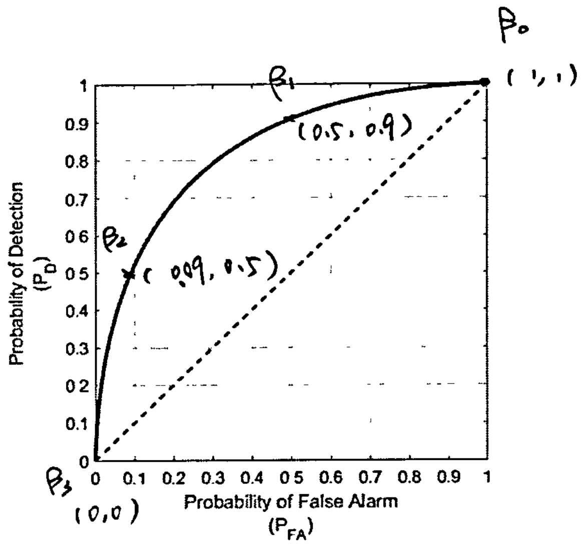




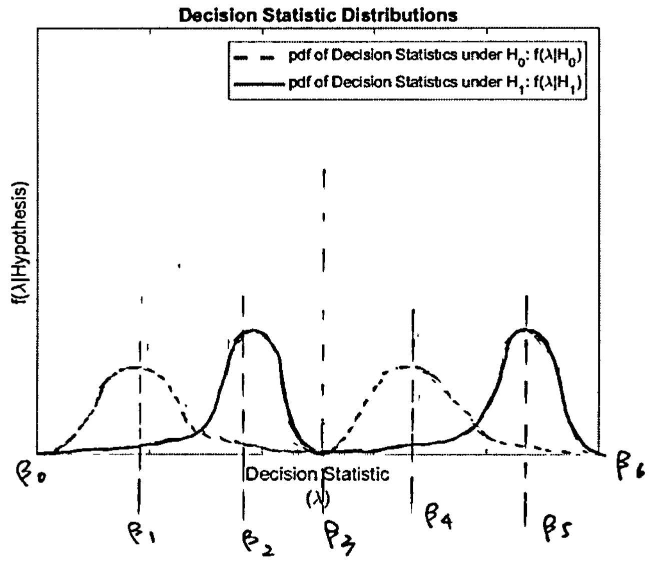
(20) 2. For each ROC given below, sketch (qualitatively, but as accurately as you can) a set of decision statistic distributions that could have plausibly produced the given ROC. We are not concerned about the precision of the pdfs, but rather their general shapes and locations relative to each other. (P_{FA} , P_D)

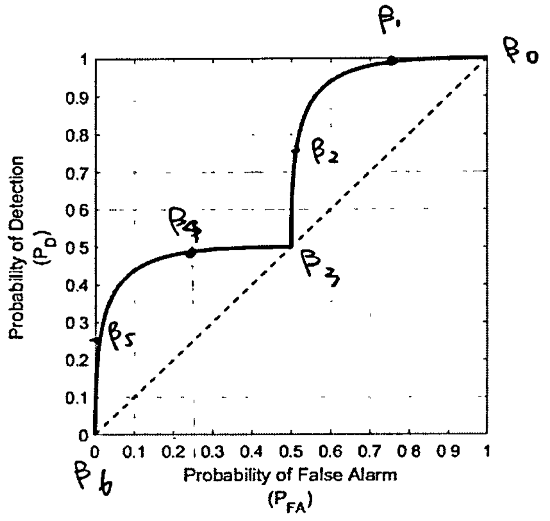
(a)



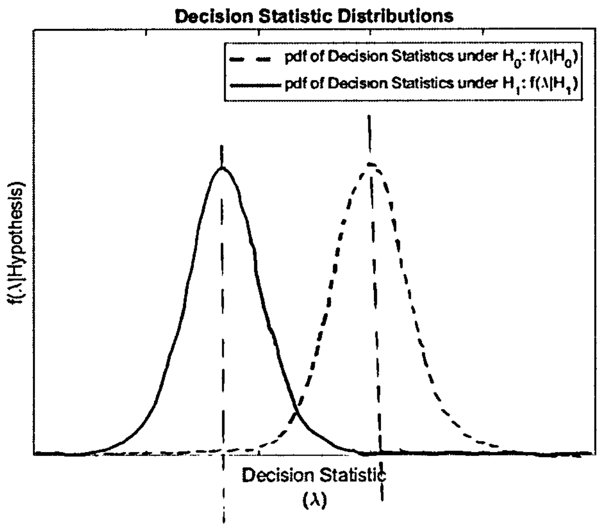


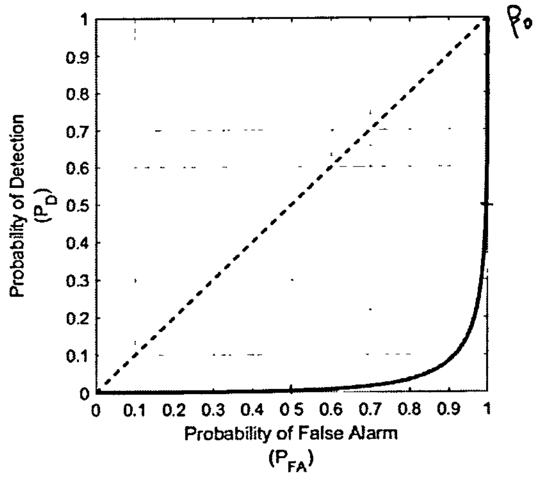
(b)



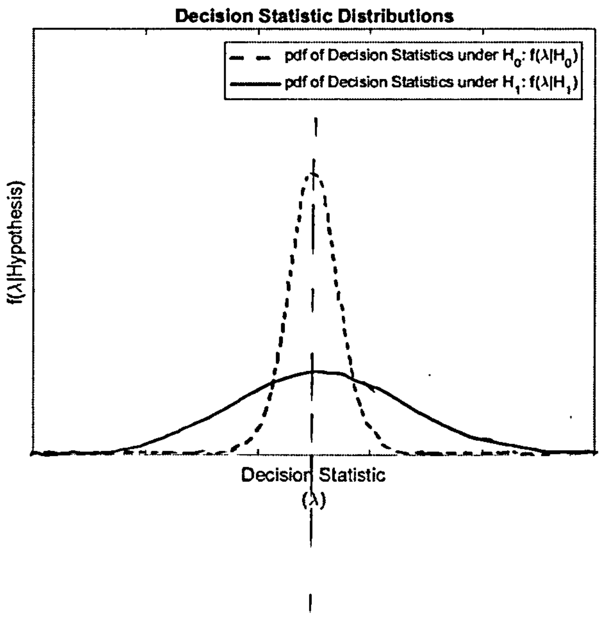


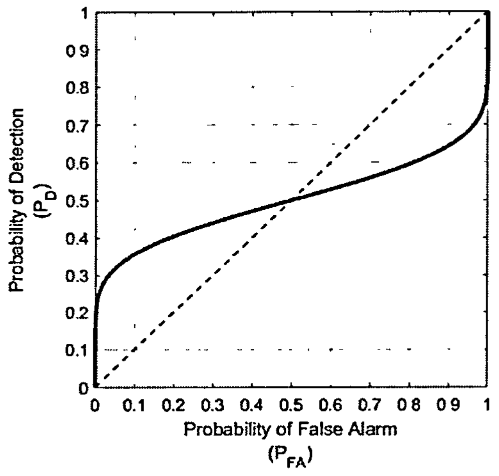
(c)


Decision Statistic Distributions
pdf of Decision Statistics under H_0: $f(\lambda|H_0)$
pdf of Decision Statistics under H_1: $f(\lambda|H_1)$
$f(\lambda|\text{Hypothesis})$
Decision Statistic
(λ)


P₀
Probability of Detection
(P_D)
Probability of False Alarm
(P_{FA})
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

(d)

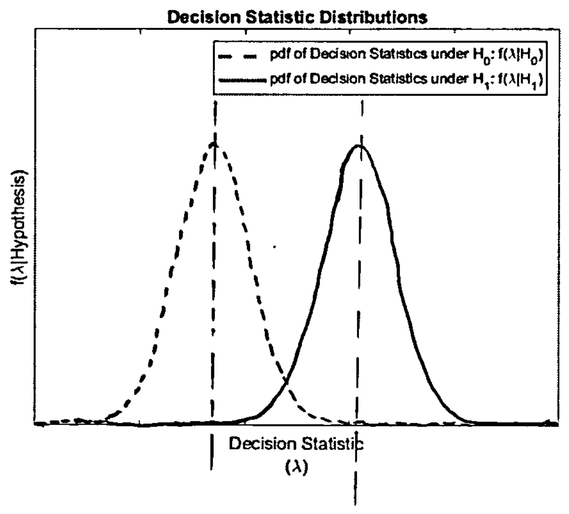

Decision Statistic Distributions
pdf of Decision Statistics under H_0: $f(\lambda|H_0)$
pdf of Decision Statistics under H_1: $f(\lambda|H_1)$
$f(\lambda|\text{Hypothesis})$
Decision Statistic
(λ)

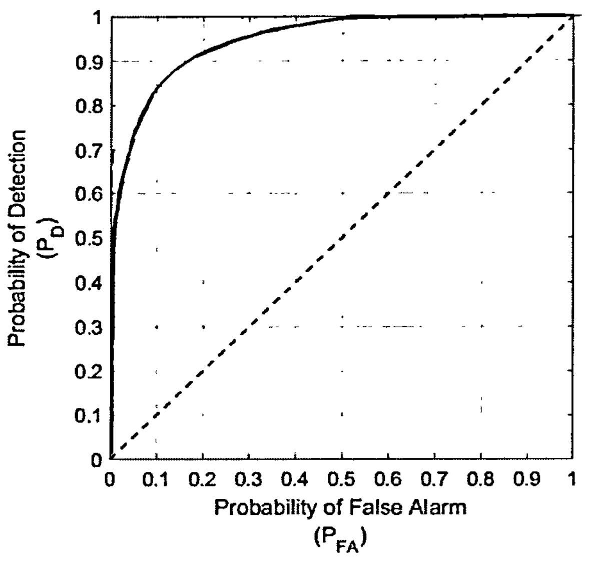

Probability of Detection
(P_D)
Probability of False Alarm
(P_{FA})
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1

(10) 3. (a) What mathematical transformation could we apply to the decision statistics for question 2c so that the ROC lies entirely above the chance diagonal?

We could negate our raw data, that it is $-\lambda$.
then the pdf of Decision Statistics will switch.

(b) Sketch (qualitatively, but as accurately as you can) the new set of decision statistic distributions after applying the mathematical transformation, and the new ROC. We are not concerned about the precision of the sketches, but rather their general shapes and locations.





(10) 4. (a) What mathematical transformation could we apply to the decision statistics for question 2d so that the ROC lies entirely above the chance diagonal?

let $u(\lambda)$ be the mean of λ, we could get absolute value of $|\lambda - u(\lambda)|$ for our raw data.

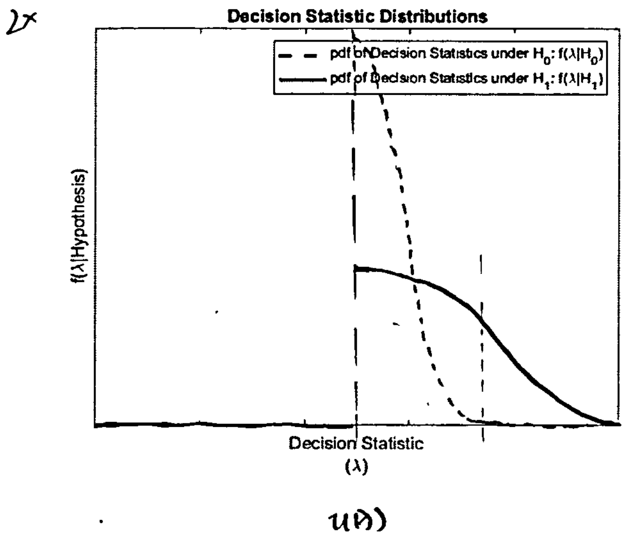
$$\begin{cases} \text{if } \lambda > u(\lambda), \text{ we keep the raw data.} \\ \text{if } \lambda < u(\lambda), \text{ we apply } \frac{x+\lambda}{2} = u(\lambda) \Rightarrow x = 2u(\lambda) - \lambda \text{ to raw data.} \end{cases}$$

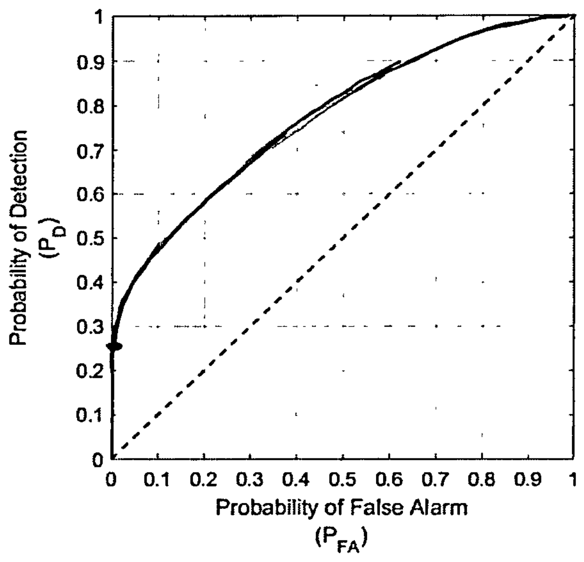
and λ will be replaced by x;

Then we would get the following pdf of Decision statistics.

The right part distribution will double, and the left part will be become 0.

(b) Sketch (qualitatively, but as accurately as you can) the new set of decision statistic distributions after applying the mathematical transformation, and the new ROC. We are not concerned about the precision of the sketches, but rather their general shapes and locations.





Generating ROCs

It is tremendously beneficial to have the ability to specify how you want an ROC to be generated, as there is no computational approach to generating ROCs that is universally better than all others under all conditions.[2]

Make sure you are able to generate an ROC by specifying the specific thresholds you want to apply and that you have flexibility to specify how those thresholds are selected – linearly spaced from $\min(\lambda)$ to $\max(\lambda)$, logarithmically spaced from $\min(\lambda)$ to $\max(\lambda)$, every n^{th} λ in the list of sorted decision statistics, every n^{th} H_0 λ in the list of sorted H_0 decision statistics, thresholds necessary to achieve a set of desired P_{FA} values, thresholds necessary to achieve a set of desired P_D values, etc. Even better is code that is extensible, so you can incorporate additional functionality, such as new approaches to specifying how the thresholds are selected or the ability to return other performance measures, as you encounter new use cases. You may choose to write your own function from scratch, or you may choose to leverage ROC generating functions available through standard Matlab[3] or Python packages, in which case you likely will find it helpful to write your own wrapper for these functions.

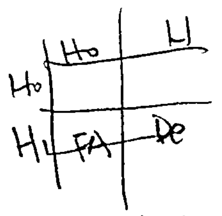
Regardless of whether you choose to write your own function or leverage functions that may be available through standard Matlab or Python packages, it is critical that you understand how the function(s) you are using work so you can effectively apply those functions to suit your needs and correctly interpret the results they provide.

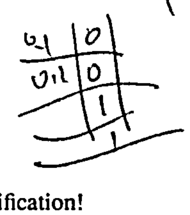
The following questions concern 4 sets of decision statistics that are provided as `csv` files. The csv files are organized such that each row contains the true class (either 0 or 1) followed by the associated decision statistic. For each set of decision statistics, generate the ROC using:

1. every decision statistic as a threshold (β is $[-\infty \; \{\text{sorted list of } \lambda\text{'s}\} \; +\infty]$))[4]
2. ✓ thresholds selected so they linearly sample the range of decision statistics (β is 99 linearly spaced samples from $\min(\lambda)$ to $\max(\lambda)$, plus $-\infty$ and $+\infty$)
3. ✓ thresholds selected so they sample every n^{th} decision statistic, where n is chosen so there will be 99 decision statistics selected as thresholds (or $n = 1$ if there too few decision statistics to down select such that 99 decision statistics are retained), plus $-\infty$ and $+\infty$
4. every H_0 decision statistic as a threshold, plus $-\infty$ and $+\infty$
5. thresholds selected so that P_{FA} is linearly sampled from 0 to 1 at an interval of 0.01 (101 samples of P_{FA})

In the questions that follow you will compare and comment on these 5 ROCs. (P_{FA}, P_D)

P threshold = x



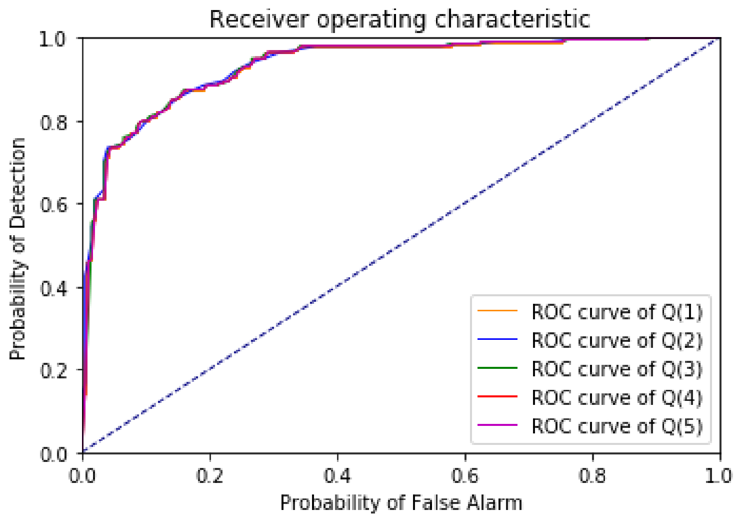


[2]It seems the "No Free Lunch" Theorem generalizes to computational goals beyond classification!

[3]If you are using Matlab, I recommend `perfcurve` (from the Statistics and Machine Learning toolbox) over `roc` (from the Neural Network toolbox) for generating an ROC curve because `roc` assumes the decision statistics fall in the range [0,1] while `perfcurve` makes no such assumptions.

[4]It is good practice to include both $-\infty$ and $+\infty$ as thresholds to ensure that the ROC spans from $(P_D,P_{FA}) = (0,0)$ to $(P_D,P_{FA}) = (1,1)$.

(20) 5. Compare the 5 ROCs for the decision statistics provided in the file moderateData.csv by plotting them on the same set of axes.



(a) Which of the 5 approaches to selecting thresholds would you consider to be appropriate for this set of decision statistics? Why?

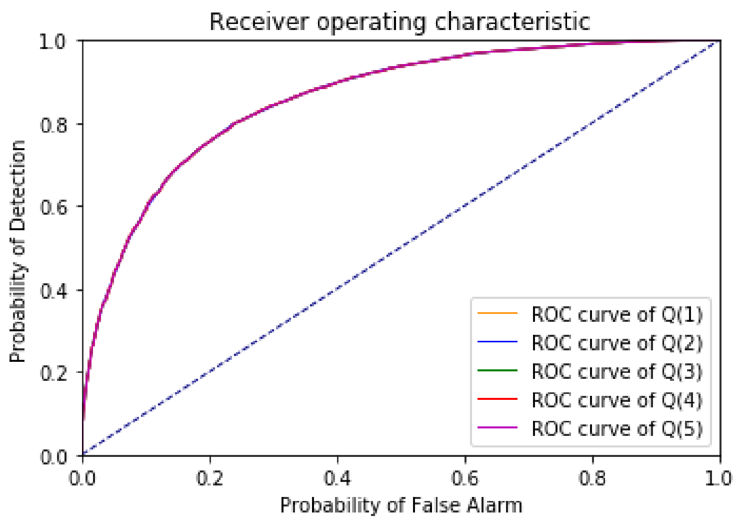
From the graph, we could see that five ROC curves have the same trend and large part of the curves are overlapped. Generally, we could say for moderateData.csv, these five approaches perform the same. Because accuracy is measured by the area under the ROC curve. All five curves have similar AUC. It seems we can't tell which one is better for this data.

When we looked into the data, we could see the reasons why five approaches perform almost the same. We have 400 test data, which won't cost too much time. 200 is labeled as 0, 200 is labeled as 1, which won't make bias. As we have moderate data, and H0 and H1 have the same probability, and the range is spanned almost evenly, we could use 1, 2, 3, and 5 approaches for this question.

(b) Which would you consider to be inappropriate for this set of decision statistics? Why?

Approach 4 is inappropriate for this dataset. We could also find that the range of H0 is -2 ~3, but the range of H1 is -1~4. Most of H1 has higher values. Thus approach 4 is inappropriate. Because if we use approach 4, it will lead to focus on the lower thresholds, because H0 is centralized in lower values.

(20) 6. Compare the 5 ROCs for the decision statistics provided in the file bigData.csv by plotting them on the same set of axes.



(a) Which of the 5 approaches to selecting thresholds would you consider to be appropriate for this set of decision statistics? Why?

From the graph, we could see that five ROC curves are nearly the same. Generally, we could say for bigData.csv, these five approaches got almost the same AUC. It seems we can't tell which one is better for this data. 5000 is labeled as 0, 5000 is labeled as 1, and test values are spanned evenly, which won't make bias. As we have a large data, we should avoid approaches 1. We could also see that H0 have small values, thus we should avoid test bias, thus we exclude approach 4.

Therefore, we could take approaches 2, 3, and 5 for this problem.

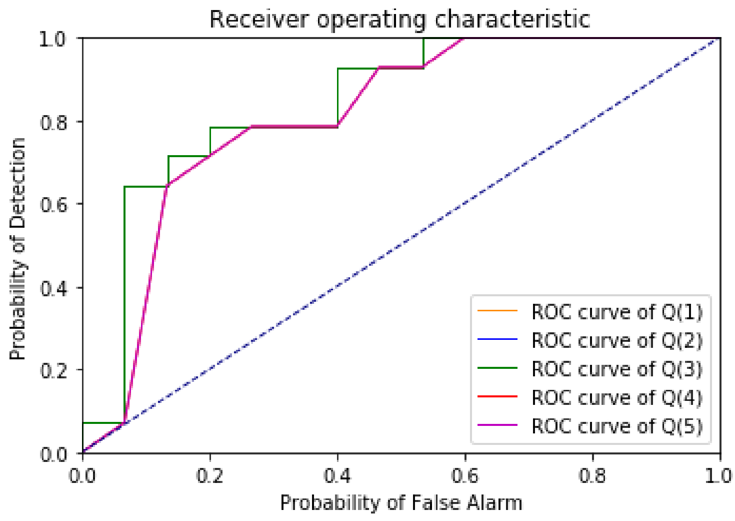
(b) Which would you consider to be inappropriate for this set of decision statistics? Why?

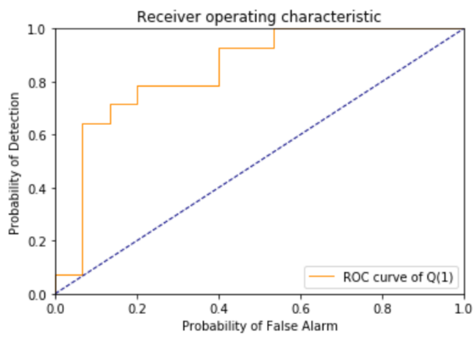
We noticed that the code for approach 1 took a long time to run. Because we have 10000 test data, which is time-wasting to perform 1. What is more, this approach will cause overfitting problems. An overfit model will have extremely low training error but a high testing error.

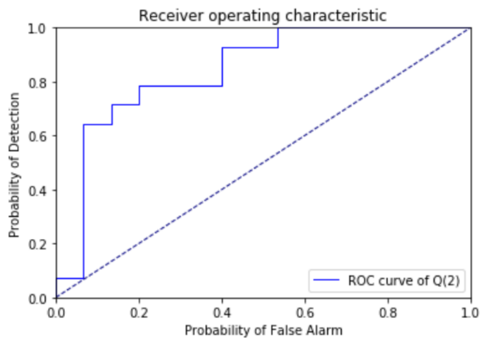
And we also notice that for H0 data, they have relatively small test values, so we should avoid only using H0 decision statistic as threshold, because it will make us only small value of threshold, which will make the training model unfit for the integral part of collected data. Therefore approaches 4 is inappropriate as well.

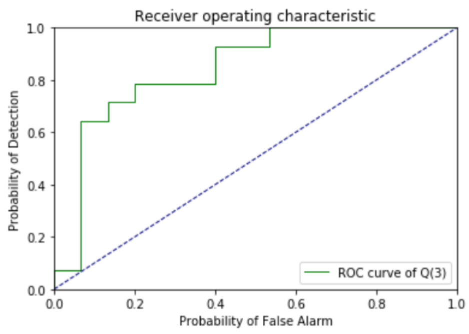
Thus approaches 1 and 4 should be avoided for this dataset.

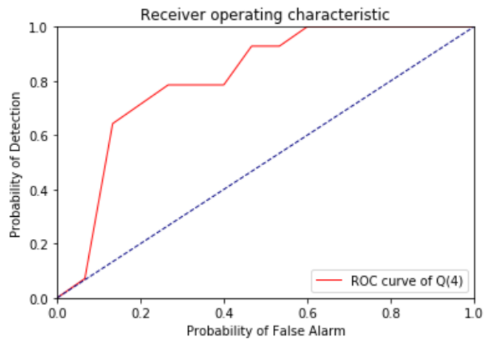
(20) 7. Compare the 5 ROCs for the decision statistics provided in the file smallData.csv by plotting them on the same set of axes.

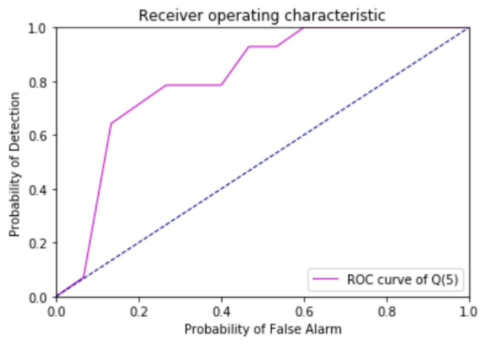












(a) Which of the 5 approaches to selecting thresholds would you consider to be appropriate for this set of decision statistics? Why?

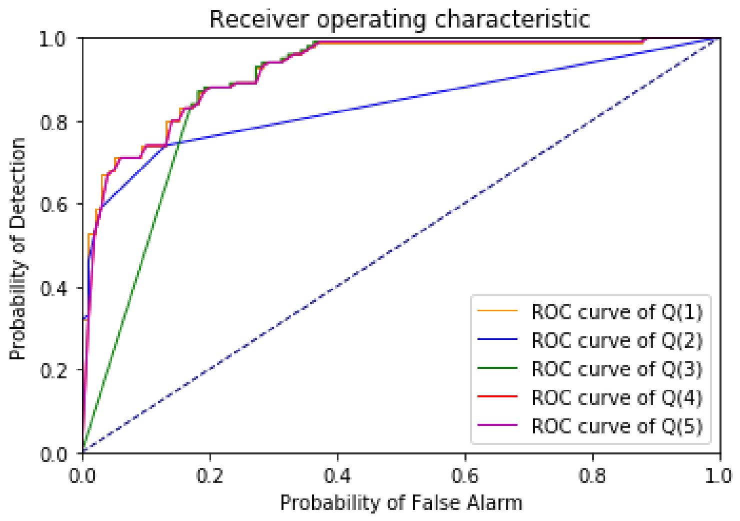
Because some curves are overlapped, I've posted them separately. We could see that approaches 1, 2, 3 have the same curves, while approaches 4, and 5 have the same curves. Because we have small data, so looping over raw data won't cause overfitting problem. And we need to increase the area under the ROC curve, which could enhance the accuracy. As we have only 30 data for this problem, approaches perform the same mechanism, therefore they have no differences for this dataset. Meanwhile, comparing with other two approaches, they have higher AUC, therefore, approaches 1, 2, 3 are appropriate for this question.

(b) Which would you consider to be inappropriate for this set of decision statistics? Why?

From the subplot graphs, we could see that these two approaches perform relatively poor than other approaches, because their AUC is relatively lower than others. According to the algorithm, they didn't test enough data, which caused underfitting problem. Approaches 4 and 5 will make a poor testing model.

Especially for small data, approaches 4 and 5 is inappropriate. A model that is underfit will have high training and high testing errors.

(20) 8. Compare the 5 ROCs for the decision statistics provided in the file logNormalData.csv by plotting them on the same set of axes.



(a) Which of the 5 approaches to selecting thresholds would you consider to be appropriate for this set of decision statistics? Why?

From the above graph, we could see that approaches 1,4 and 5 have almost the same curves. Because we have only 200 test data, approaches 1 won't cause overfitting problem. And we also notice that H0 and H1 have the same number. At the same time, the range of the dataset is very large. For approaches 4, most of test data has relatively same range with H0 data, therefore approaches 4 is also fine. While for approaches 5, we select Pfa evenly across x axis, which is perfect for moderate dataset.

Therefore, approaches 1,4 5 is appropriate for this dataset.

(b) Which would you consider to be inappropriate for this set of decision statistics? Why?

Approaches 2 and 3 is inappropriate for this dataset. Because for approach 2, we the min and max value have big variance, while most of test values are centralized in relatively small value. For approaches 3, it is also the same problem. Because most of test data has relative small value, strictly looping over nth decision statistic would take the same weight for large and small decision statistics, which is inappropriate in this question.

Therefore approaches 2 and 3 is inappropriate.

9. For each of the five approaches for selecting thresholds so generate an ROC considered here, explain why it is, or not, universally applicable, meaning it will provide a good representation of the ROC without unnecessary computations.

Approach 1: For small dataset, this approach works perfectly. Because it loops over every test data to compute PA and PDA, and could help avoid underfitting problems for small data. However, for large dataset, this will cause overfitting problem which causes unnecessary computations.

Approach 2: This approach works fine for dataset which have evenly spanned test values. And picking decision statistics evenly for 99 times, won't cause unnecessary computations for large dataset. However, for dataset whose min and max test values have great variance, will cause problem. Especially, when test value focuses on a relatively low/high range, but have a really large/small max test value. This approaches won't make a good representation of ROC.

Approach 3: This approach works for evenly spanned test value. However, for widely spanned test value, whose value differs a lot, and which are centralized in some certain value, this won't make a good test model.

Approach 4: This approach is not good for most of dataset. Because it only focuses on H0 values. Most of time, H0 and H1 have different test values and have different proportions. If we only select H0 decision statistic as threshold, we will lose the weight of H1 test influence. This causes serious problem especially when the dataset carries large proportion of H1 dataset, and when H1 and H0 have distinguished different value.

Approach 5: For most of dataset, th-is approach works well. Comparing with the above five approaches, this approach would generate a good representation of ROC. Because it evenly picks the PFA on the x axis, which would generate an evenly weighted thresholds. It won't cause underfitting or overfitting problems. It is the most applicable approach when we don't know the size or other information of our dataset.

ROCs ↔ Summary Performance Measures

It is helpful to be able to compare ROCs more quantitatively than visually evaluating which is closer to the upper left corner of the ROC graph. Two summary performance measures that are commonly used are AUC (the <u>a</u>rea <u>u</u>nder the ROC <u>c</u>urve) and the mmaximum probability of correct decision, $\max P_{cd}$, (or, equivalently, the minimum probability of error, $\min P_e$).

Make sure you have a way to find summary performance measures for an ROC. You should be able to find AUC, $\max P_{cd}$, and $\min P_e$.

(10) 10. How are $\max P_{cd}$ and $\min P_e$ related?

Their sum equals to 1.

(10) 11. Assume the priors on the two hypotheses are equal ($p(H_0) = p(H_1) = 0.5$), and sketch two ROCs where the ROC with the higher AUC has lower $\max P_{cd}$.

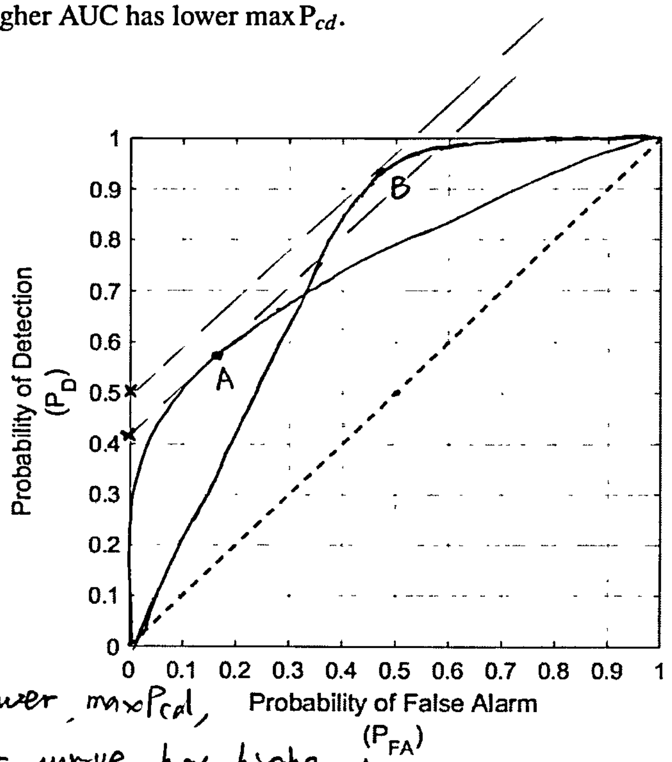
	0.5	0.5
	H_0	H_1
H_0	CR	Miss
H_1	FA	De

$P_{FA}: x$

$P_D: y$

$P_{cd} = \frac{1}{2}(1-x) + \frac{1}{2}y$

$y = 2P_{cd} + 1 + x$



Point A has lower $\max P_{cd}$, but its ROC curve has higher AUC.

(20) 12. The csv file `rocData.csv` contains (P_{FA}, P_D) pairs for an ROC curve, and is organized such that each row contains P_{FA} followed by P_D.

(a) On a single set of axes, plot:

1. the ROC curve
2. the operating point corresponding to $\max P_{cd}$ when $p(H_0) = p(H_1)$
3. the operating point corresponding to $\max P_{cd}$ when $2p(H_0) = p(H_1)$
4. the operating point corresponding to $\max P_{cd}$ when $p(H_0) = 2p(H_1)$

X. Y.

(b) Provide the AUC for the ROC and the maximum P_{cd} for each of the three operating points. These performance metrics can be provided either in the legend of the graph, or included as separate text below the graph.

$p(H_0 + H_1) = 1$, let $P_{FA}: x$

$P_D: y$

	H_0	H_1
H_0	CR	Miss
H_1	FA	D

a) 2)

$$\begin{cases} p(H_0)+p(H_1)=1 \\ p(H_0)=p(H_1) \end{cases} \Rightarrow p(H_0)=p(H_1)=0.5$$

$$P_{cd} = \tfrac{1}{2}(1-x)+\tfrac{1}{2}y \Rightarrow y = 2P_{cd} - 1 + x$$

3)

$p(H_0) = 1/3$

$p(H_1) = 2/3$

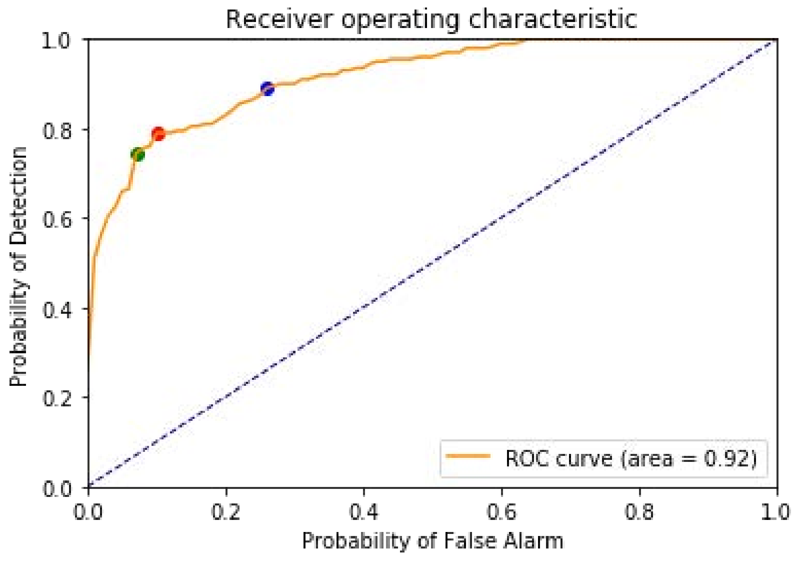
$$P_{cd} = \tfrac{1}{3}(1-x)+\tfrac{2}{3}y \Rightarrow y = \tfrac{1}{2}(x+3P_{cd}-1)$$

4)

$p(H_0) = 2/3$

$p(H_1) = 1/3$

$$P_{cd} = \tfrac{2}{3}(1-x)+\tfrac{1}{3}y \Rightarrow y = 2x+3P_{cd}-2$$



point 1: red
point: blue
point: green

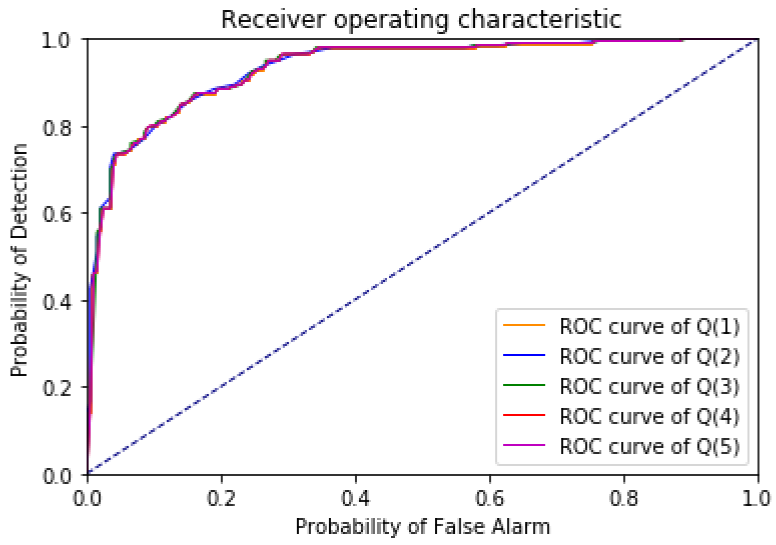
```
In [1]: import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        import itertools as it
        %matplotlib inline
```

```
In [66]: def ROC(path):
             df = pd.read_csv(path)
             df.head()

             truth = np.array(df)[:,0]
             test = np.array(df)[:,1]
             zeros = np.count_nonzero(truth==0)
             ones = len(truth)-zeros
             mydict = list(zip(test, truth))
             mydict_sorted = mydict[:]
             mydict_sorted.sort()

             # Q(1) solution
             PD1 = []
             PFA1 = []
             PD1.append(1)
             PFA1.append(1)
             for k,v in mydict_sorted:
                 D1 = 0
                 FA1 = 0
                 for test, truth in mydict_sorted:
                     if test >= k and truth == 1:
                         D1 += 1
                     elif test >= k and truth == 0:
                         FA1 += 1
                 PD1.append(D1/ones)
                 PFA1.append(FA1/zeros)
             PD1.append(0)
             PFA1.append(0)

             # Q(2) solution
             PD2 = []
             PFA2 = []
             PD2.append(1)
             PFA2.append(1)
             minlam = mydict_sorted[0][0]
             maxlam = mydict_sorted[-1][0]
             dist = (maxlam - minlam)/98
             for i in range(0,99):
                 D2 = 0
                 FA2 = 0
                 k = minlam + i * dist
                 for test, truth in mydict_sorted:
                     if test >= k and truth == 1:
                         D2 += 1
                     elif test >= k and truth == 0:
                         FA2 += 1
```

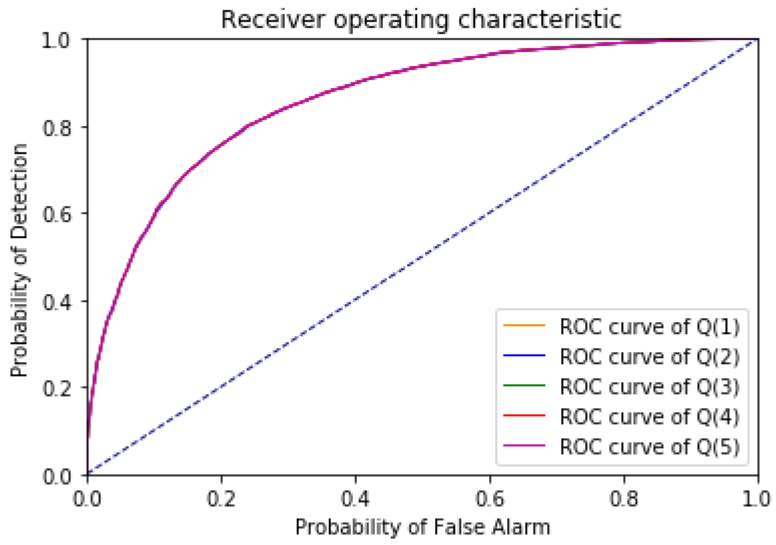
```
            PD2.append(D2/ones)
            PFA2.append(FA2/zeros)
        PD2.append(0)
        PFA2.append(0)

        # Q(3) solution
        PD3 = []
        PFA3 = []
        thld3 = []
        length = df.shape[0]
        for k,v in mydict_sorted:
            thld3.append(k)
        PD3.append(1)
        PFA3.append(1)
        m = 0
        n = length//100
        if n == 0:
            n = 1
        iternum = 99
        if df.shape[0]<99:
             iternum= df.shape[0]
        for i in range(0,iternum):
            D = 0
            FA = 0
            m =  n * i
            for test, truth in mydict_sorted:
                if test >= thld3[m] and truth == 1:
                    D += 1
                elif test >= thld3[m] and truth == 0:
                    FA += 1
            PD3.append(D/ones)
            PFA3.append(FA/zeros)
        PD3.append(0)
        PFA3.append(0)

        # Q(4) solution
        PD4 = []
        PFA4 = []
        thld4 = []
        for k,v in mydict:
            if v==0:
                thld4.append(k)
        thld4.sort()
        PD4.append(1)
        PFA4.append(1)
        for k in thld4:
            D = 0
            FA = 0
            for test, truth in mydict_sorted:
                if test >= k and truth == 1:
                    D += 1
                elif test >= k and truth == 0:
                    FA += 1
            PD4.append(D/ones)
            PFA4.append(FA/zeros)
        PD4.append(0)
        PFA4.append(0)
```

```
    PD5 = []
    PFA5 = []

    # Q(5) solution
    PD5.append(1)
    PFA5.append(1)
    num = df.shape[0]
    for k in range(0,101):
        D = 0
        FA = 0
        k = 0.01 * num * i
        for test, truth in mydict_sorted:
            if test >= k and truth == 1:
                D += 1
            elif test >= k and truth == 0:
                FA += 1
        PD5.append(D/ones)
        PFA5.append(FA/zeros)
    PD5.append(0)
    PFA5.append(0)

    #plot ROC curves
    plt.figure()
    plt.plot(PFA1, PD1, color='darkorange',
         linewidth = 1, label='ROC curve of Q(1)')
    plt.plot(PFA2, PD2, color='b',
         linewidth = 1, label='ROC curve of Q(2)')
    plt.plot(PFA3, PD3, color='green',
        linewidth = 1, label='ROC curve of Q(3)')
    plt.plot(PFA4, PD4, color='red',
        linewidth = 1, label='ROC curve of Q(4)')
    plt.plot(PFA4, PD4, color='m',
      linewidth = 1, label='ROC curve of Q(5)')
    plt.plot([0, 1], [0, 1], color='navy', linewidth = 1, linestyle='--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.0])
    plt.xlabel('Probability of False Alarm')
    plt.ylabel('Probability of Detection')
    plt.title('Receiver operating characteristic')
    plt.legend(loc="lower right")
    plt.show()
```

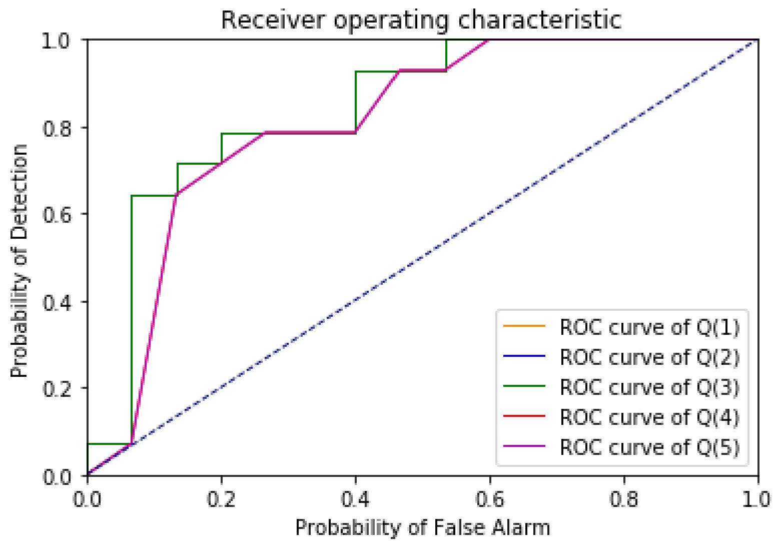
In [52]:
```
ROC('moderateData.csv')
```



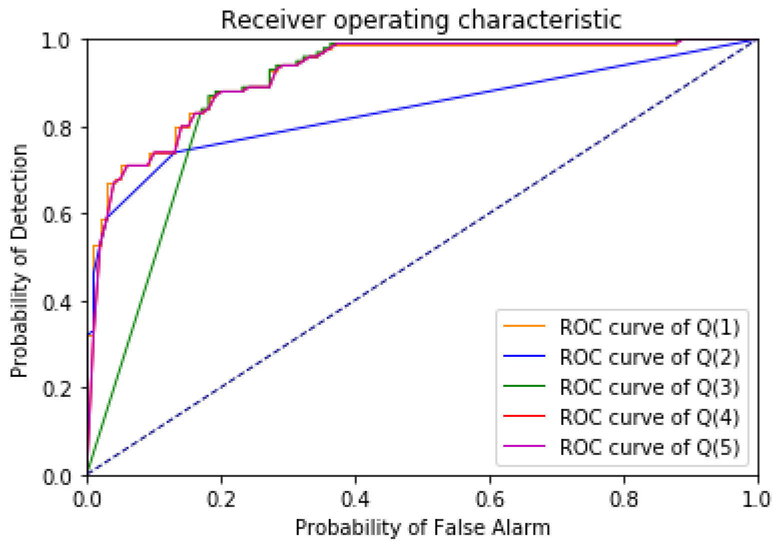
```
In [36]: ROC('bigData.csv')
```



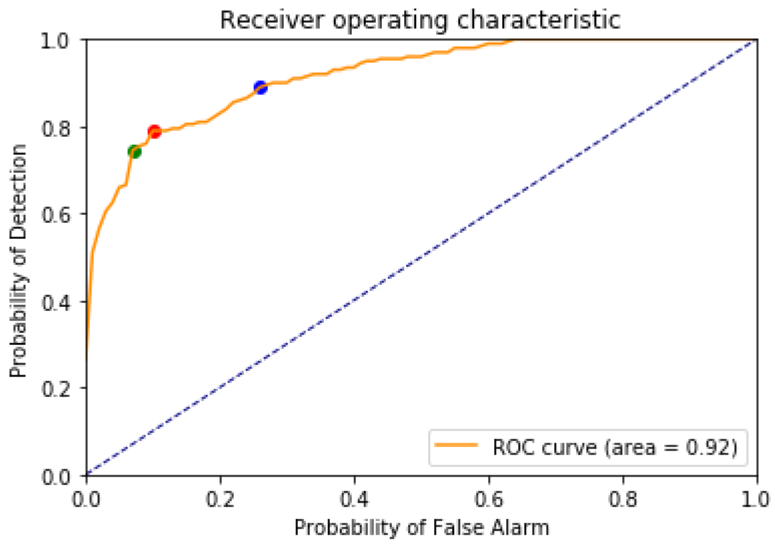
```
In [67]: ROC('smallData.csv')
```



In [38]:
```
ROC('logNormalData.csv')
```



In [48]:
```
df = pd.read_csv('rocData.csv')
PFA = np.array(df)[:,0]
PD = np.array(df)[:,1]

# Point 1
mydict = dict()
for i in range(0,len(PFA)):
    Pcd = 0.5 * (1 - PFA[i]) + 0.5 * PD[i]
    mydict[Pcd] = i
x = PFA[mydict[max(mydict)]]
y = PD[mydict[max(mydict)]]

# Point 2
mydict1 = dict()
for i in range(0,len(PFA)):
    Pcd =  (1 - PFA[i])/3 + 2 * PD[i]/3
    mydict1[Pcd] = i
x1 = PFA[mydict1[max(mydict1)]]
y1 = PD[mydict1[max(mydict1)]]

# Point 3
mydict2 = dict()
for i in range(0,len(PFA)):
    Pcd =  2 * (1 - PFA[i])/3 + PD[i]/3
    mydict2[Pcd] = i
x2 = PFA[mydict2[max(mydict2)]]
y2 = PD[mydict2[max(mydict2)]]

from sklearn.metrics import roc_curve, auc

# Plot
plt.figure()
plt.plot(PFA,PD,label='ROC curve (area = %0.2f)' % auc(PFA,PD), c = 'darko
range')
plt.plot([0, 1], [0, 1], color='navy', linewidth = 1, linestyle='--')
plt.scatter(x, y, c='red')
plt.scatter(x1, y1, c='blue')
plt.scatter(x2, y2, c='green')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.xlabel('Probability of False Alarm')
plt.ylabel('Probability of Detection')
plt.title('Receiver operating characteristic')
plt.legend(loc="lower right")
plt.show()
```



In [ ]: