(1) How did you determine the radius of the halo to extract each object?

Well with cars there is a high possibility that there are other cars around a red or a white car. In order to make sure that the algorithm doesn't mistakenly overlap two cars as the same (even if they are the same color) then we realized we need to keep the radius small so that we can keep our false positives as low as possible. For pools their shapes can vary, but there is not a high chance of there being overlapping pools in one area. An example would be a neighborhood with every other house having a pool. We know we can increase the size of the halo radius to encapsulate the whole shape of the pool while being able to identify it.

(2) Is color the only feature you can use to be able to discriminate classes? What other features can be included?

No color isn't the only one. After brainstorming with our group we decided that shape can be a feature to extract as well with location. Shape of cars are generally the same, usually a rectangular shape. But with pools and ponds, the shapes can vary based on the design. So with location, we thought we can distinguish pools easier since usually there are one pool per household. With cars, there are multiple that appear near each other on highways, parking lots etc.

(3) Make a confusion matrix for every clustering algorithm in the validation set. Describe each confusion matrix.

```
EM for GMM Model
-------------------------------------------
Confusion Matrices in the Training Set
-------------------------------------------
RGB
[[  366    30    77    13]
 [   24     8   272     6]
 [  101    12  3618   194]
 [ 5662 21746   939 49199]]
-------------------------------------------
HSV
[[   18   421     1    64]
 [    6    57     0   223]
 [   66   117   132  3608]
 [23600  9354 41842  2758]]
-------------------------------------------
YIQ
[[   10    32     4   437]
 [    2   247     0    50]
 [  195  3586     8   126]
 [46701   171 20840  9858]]
-------------------------------------------
HLS
[[    1   120    27   339]
 [    0    52     9   236]
 [  112    47    62  3718]
 [42693  8722 23547  2582]]
```

```
---------------------------------------------
EM for GMM Model
---------------------------------------------
Confusion Matrices in the Validation Set
---------------------------------------------
RGB
[[    1     1    13   120]
 [    0     0    56    13]
 [    6    54   888    36]
 [ 5210 11726    47  2396]]
---------------------------------------------
HSV
[[   10     0   115     2]
 [   34     0    25     0]
 [  843     8    60    49]
 [    5  3171  5253 10992]]
---------------------------------------------
YIQ
[[  108     2     0     9]
 [   13     0     0    57]
 [   21    37     2   918]
 [ 2397 11842  5116    45]]
---------------------------------------------
HLS
[[    1     3    74    29]
 [    0     4    62    13]
 [   31    13   923     8]
 [10475  6084   698  2149]]
---------------------------------------------
```

Above is the confusion matrices for the EM algorithm. From what we learned in class, each row-column pair (ie 1:1 for row 1 and column 1) signifies that a data point was rightfully identified. Each row signifies the clusters and as we can see for RGB, it most accurately learned the 3rd cluster. For the fourth cluster we can see it was hard for the algorithm to identify what data point belonged where, thus giving us a spread distribution among the columns. For HSV we can see that it didn't accurately identify the data points to their respective clusters, with our most accurate identifications coming in the 4th row. For YIQ in the first row first column we can see it was very accurate in identifying those points to their cluster. But, as we move down we see it becomes a bit sparse in the clustering process.

K-Means

Confusion Matrices in the Training Set

RGB

```
[[   13    19    61   377]
 [   22   101    98    61]
 [  200   347  1802  1585]
 [30337 36968  5219  5057]]
```

HSV

```
[[  425    55    15     1]
 [   56   219     4     3]
 [ 2998    39   135   736]
 [ 6062  8461 46092 16966]]
---------------------------------------------
YIQ
[[   21   395    16    52]
 [   89    62    54    81]
 [  267  1614   308  1762]
 [39232  5210 26335  6769]]
---------------------------------------------
HLS
[[  420    62     5    12]
 [   43   252     0     5]
 [ 3141    70   665    55]
 [ 5498  7900 33109 31030]]
---------------------------------------------
---------------------------------------------
Confusion Matrices in the Testing Set
---------------------------------------------
RGB Validation
[[   4    9   17  112]
 [  15   23   28   16]
 [  53   70  457  390]
 [7634 9231 1253 1255]]
---------------------------------------------
HSV Validation
[[   28     1    61    26]
 [   19    10    43    10]
 [  576     0   351    69]
 [ 1665    17 13818  3873]]
---------------------------------------------
YIQ Validation
[[   40    41    42     5]
 [   31    12    29     6]
 [  211    49   578   115]
 [ 4425  1403 12773   807]]
---------------------------------------------
HLS Validation
[[   45     0    19    49]
 [   11     3    19    31]
 [  592     0   132   249]
 [ 2560    14  3088 13755]]
```

Above is the confusion matrices for the K-means algorithm. As we can see from previous analysis, the first cluster doesn't identify the points well at all. The only K-means identifies well for RGB is the 3rd cluster. For HSV we can see a similar pattern, yet the 4th cluster being the one that identifies it more so than the others. As we look into the testing set we can see it still isn't too accurate but it is more accurate than the training set confusion matrices.

FCM

Confusion Matrices in the Training Set

RGB
[[405   62   20   12]
 [72   122   68   23]
 [1768 1627   366   152]
 [5383 9154 36811 26222]]
HSV
[[1   16   38   430]
 [2   4   185   94]
 [1333   64   84 2444]
 [26688 31158 13454 6272]]

YIQ
[[18   51   12   420]
 [55   123   48   65]
 [361 1600   263 1696]
 [35854 13683 22526 5492]]

HLS
[[55   420   3   13]
 [234   53   1   6]
 [435 2012 1427   64]
 [10468 6103 30373 30600]]

Confusion Matrices in the Testing Set

RGB Validation
[[97 9 2 5]
 [13 44 15 7]
 [431 417 104 39]
 [1328 2261 9129 6666]]

HSV Validation
[[62 28 4 33]
 [29 27 13 10]
 [260 240 0 479]
 [12893 5234 26 1229]]

YIQ Validation
[[8 32 50 21]
 [2 27 32 12]
 [9 343 578 54]
 [761 4040 13199 1399]]

HLS Validation
[[3 31 33 54]
 [12 11 18 29]
 [0 453 298 215]
 [19 1839 5576 11976]]

For FCM we can see in the training set that it identifies the first cluster accurately for RGB, but as we go down it seems to become inaccurate again. But for the testing set we can see it still is sporadically identifying them inaccurately, possibly due to the nature of assigning partial membership.

PCM

Confusion Matrices in the Training Set

RGB
[[31   17   0   438]
 [103   46   0   134]
 [683   312   66 2872]
 [30856 25040 13794 7875]]
HSV
[[92   15   9   358]
 [241   7   9   31]
 [2594   0   294 1035]
 [16117 13473 43507 4485]]

YIQ
[[5   465   0   6]
 [25   250   0   2]
 [19 3475   70   301]
 [11126 41966 14091 10466]]

HLS
[[12   423   3   48]
 [141   131   7   13]
 [2552 1187   79   92]
 [3972 6734 24682 42191]]

Confusion Matrices in the Testing Set

RGB Validation
[[9   4   0 113]
 [25   13   0   43]
 [191   61   22 697]
 [7664 6272 3516 1937]]

HSV Validation
[[62   23   53   0]
 [29   5   42   0]
 [470   270   223   18]

```
 [ 3149  6093 10120    10]]
-------------------------------------------
YIQ Validation
[[   32   104     0     0]
 [   14    73     0     0]
 [  354   685     0     0]
 [ 7283 12022     0     0]]
-------------------------------------------
HLS Validation
[[   26    34    66     0]
 [   31     8    33     0]
 [  329   293   371     1]
 [ 4474  1395 13389   117]]
-------------------------------------------
```

For PCM we can observe a similar spread of inaccuracy among the confusion matrices compared to the others. The identifications seem quite sparse for FCM and PCM compared to K-means and EM algorithms.

(4) Plot their respective ROC curves. **ROC curve graphs are below**

(5) What is the accuracy for each clustering algorithm in both training and validation sets?

	RGB	HSV	YIQ	HLS
FCM Train	**34%**	**3.9%**	**3.6%**	**19.6%**
FCM Valid	**17%**	**3.2%**	**3.4%**	**30%**
PCM Train	**9.7%**	**5.9%**	**13.1%**	**51.5%**
PCM Valid	**9.65%**	**1.4%**	**.510%**	**2.53%**
K-means Train	**4.2%**	**11.3%**	**4.3%**	**39%**
K-means Valid	**4.2%**	**21%**	**3.5%**	**68%**
GMM Train	**60.2%**	**3.60%**	**12.3%**	**3.27%**
GMM Valid	**15.9%**	**54.0%**	**.277%**	**7.4%**

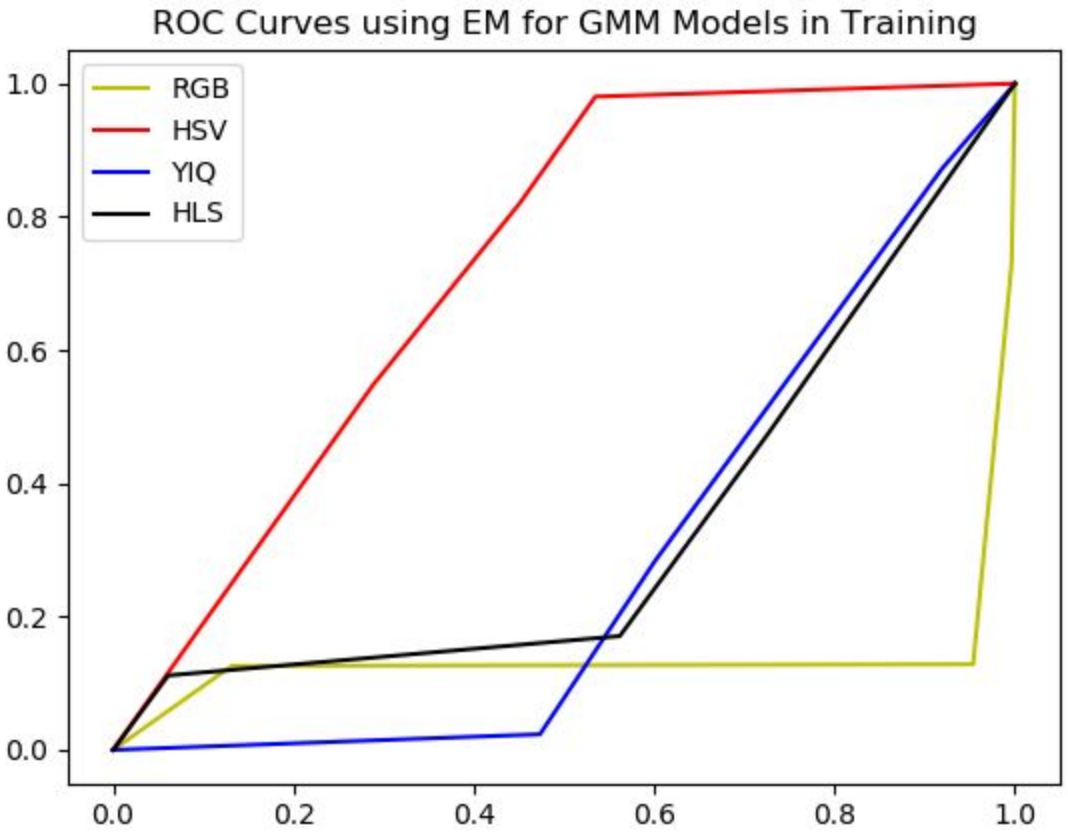
(6) Provide a detailed discussion about the steps you will need to take to solve the project. I suggest you to make a table of steps (feature extraction, detection, classification, and/or other) and include details about the algorithms you expect to implement for each step. Add a column for advantages, disadvantages and expected complications.
(This question does not have ONE solution. But it is important for you to start thinking about the steps you will need to take to solve the project, if you haven't started already.)

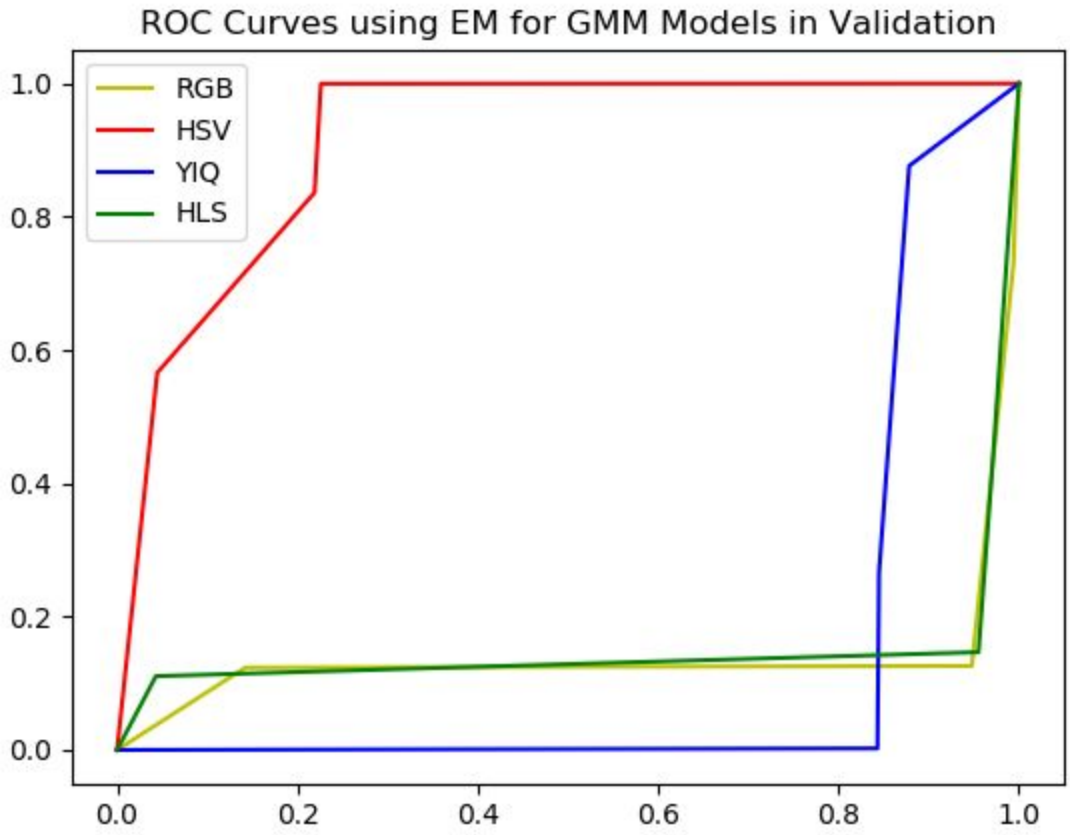
1. Our first step to complete this project is to label our training set so that we can use a supervised learning algorithm to best guess and identify our objects.
2. After that we can decide what features we can extract from each identifiable objects, we are leaning towards using color, shape and location. If we can keep our feature space small, we are expected to have a less complex NN because we wouldn't have as many hidden layers. Our team is planning on accumulating a large amount of data by finding similar satellite images and annotating them manually.
3. For detection, it would be best to split up the training data into 'regions' so we can implement a form of R-CNN from the Tensorflow API since we have already explored libraries and read some of the documentation. This would allow us

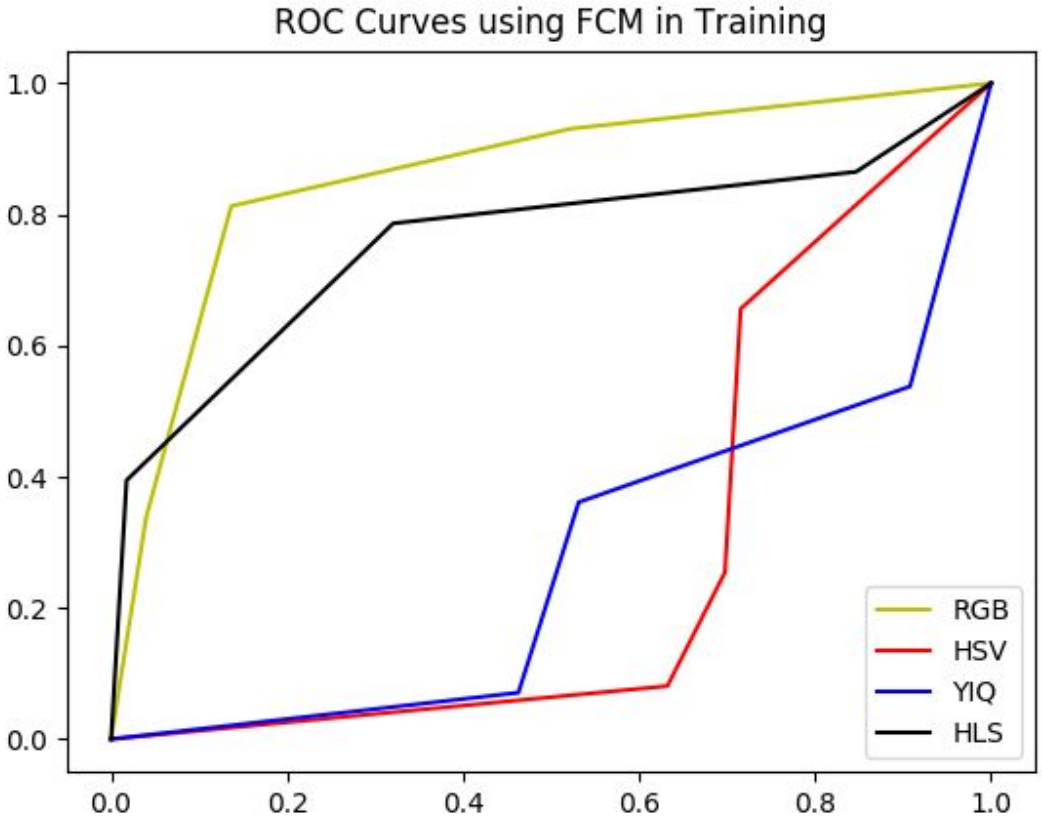
	ADVANTAGES	DISADVANTAGES	POSSIBLE COMPLICATIONS
CNN	-Locality: Since the pixels are close to each other (autocorrelation)	Complexity increases dramatically	Time given to train the NN

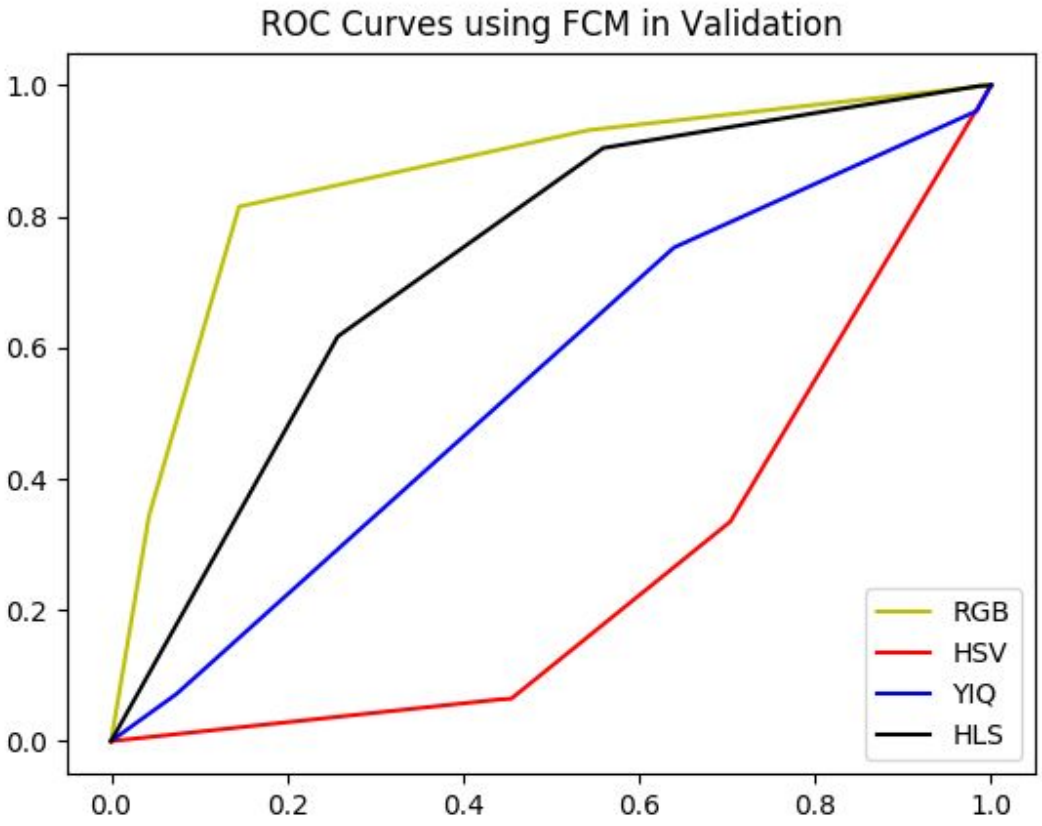
	-Accuracy		
KMEANS (Fail-safe)	-Allows for error, easy to match since the labels are in halos. -Easy implementation	Centers could be on the edge of the halos and the overlap would be small	Varying density of data for the clusters could contradict the size we initialize at

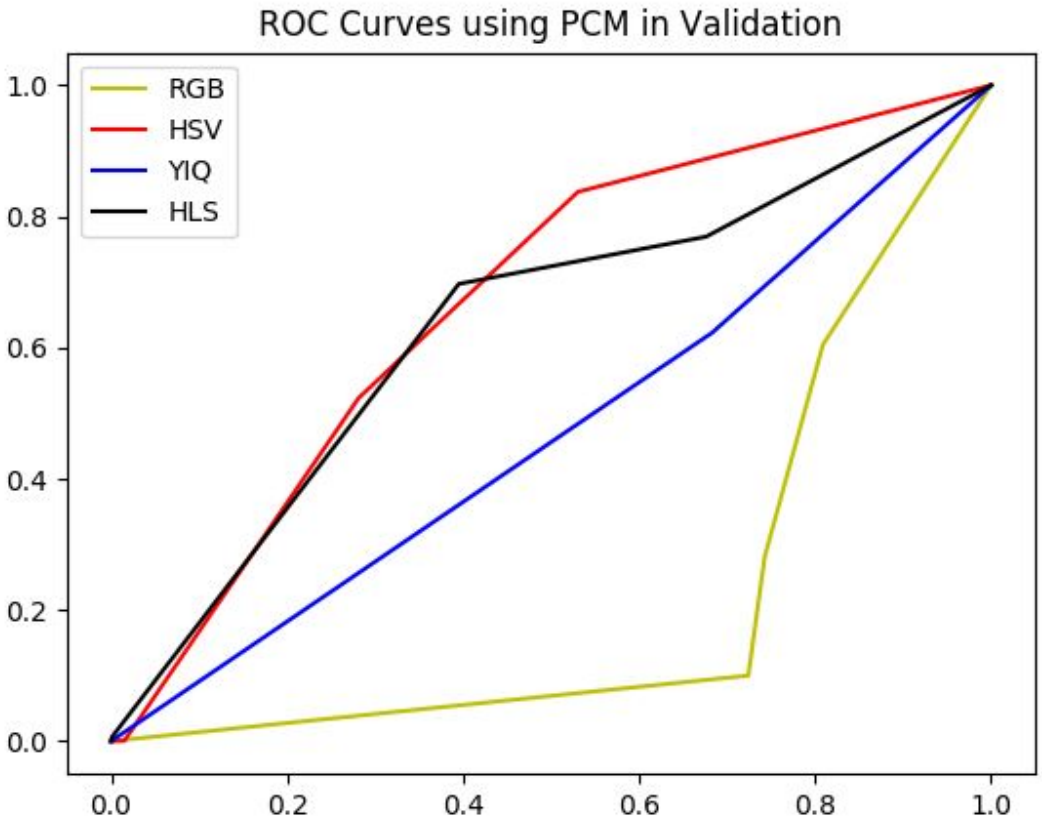
GRAPHICS:

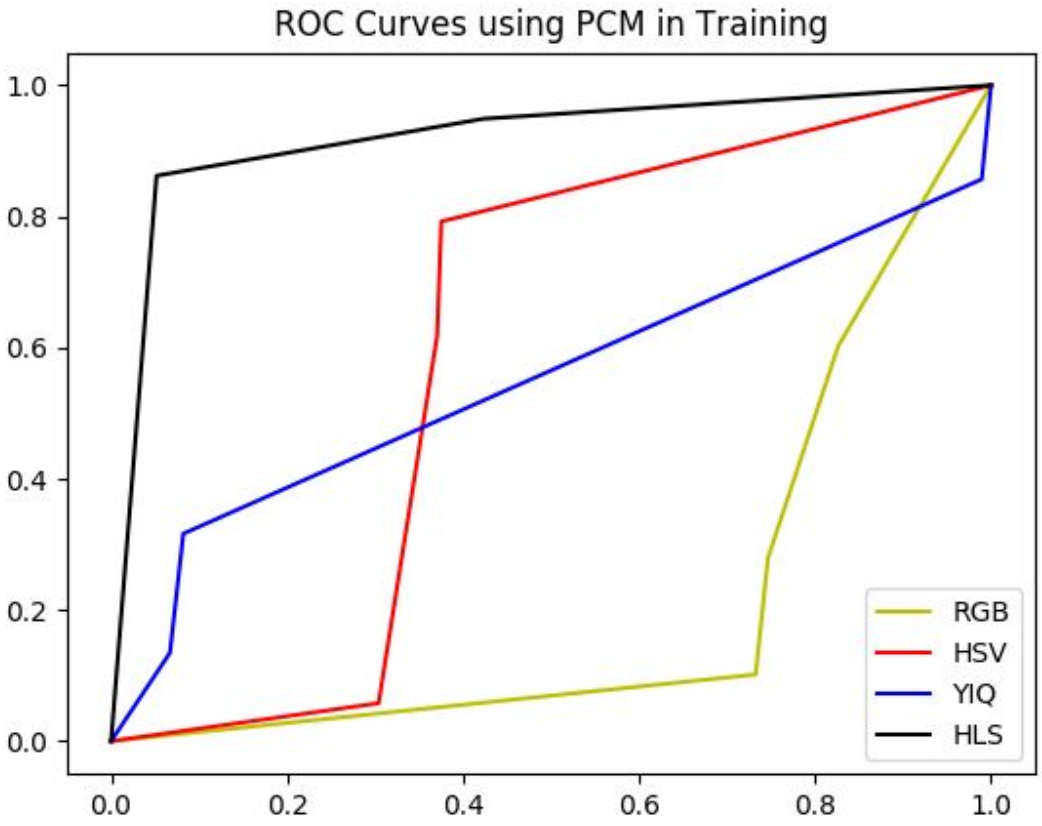


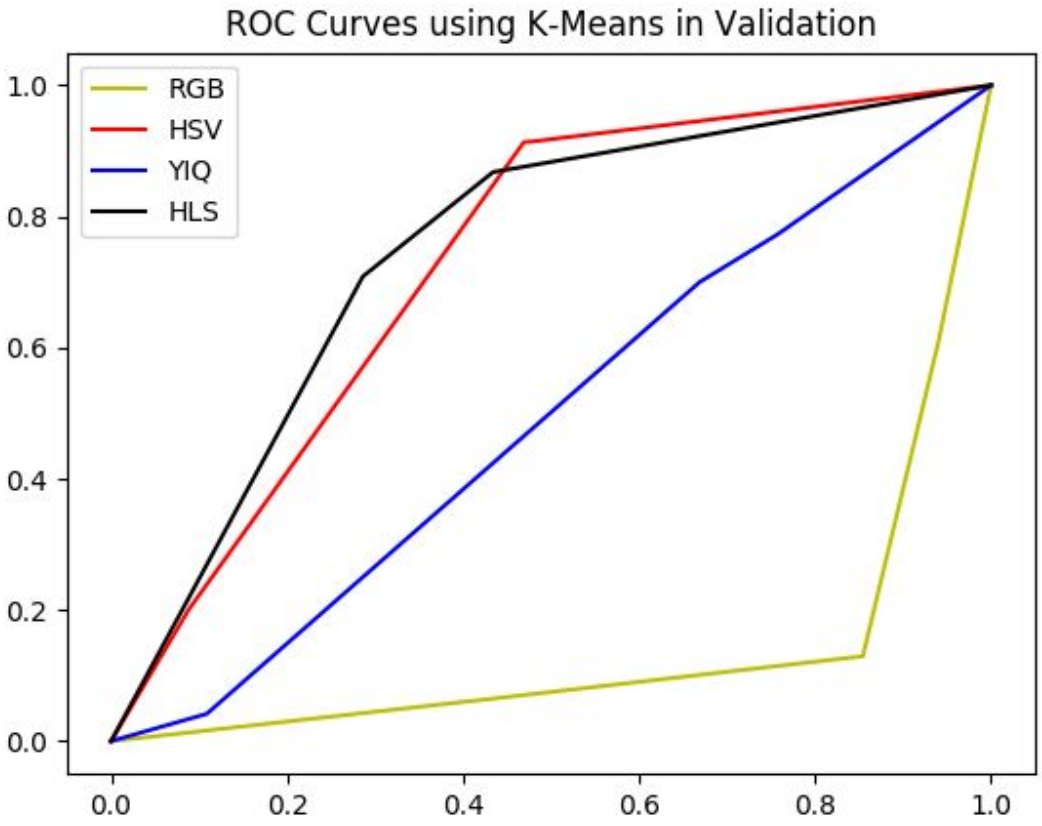

ROC Curves using EM for GMM Models in Validation
RGB
HSV
YIQ
HLS
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0

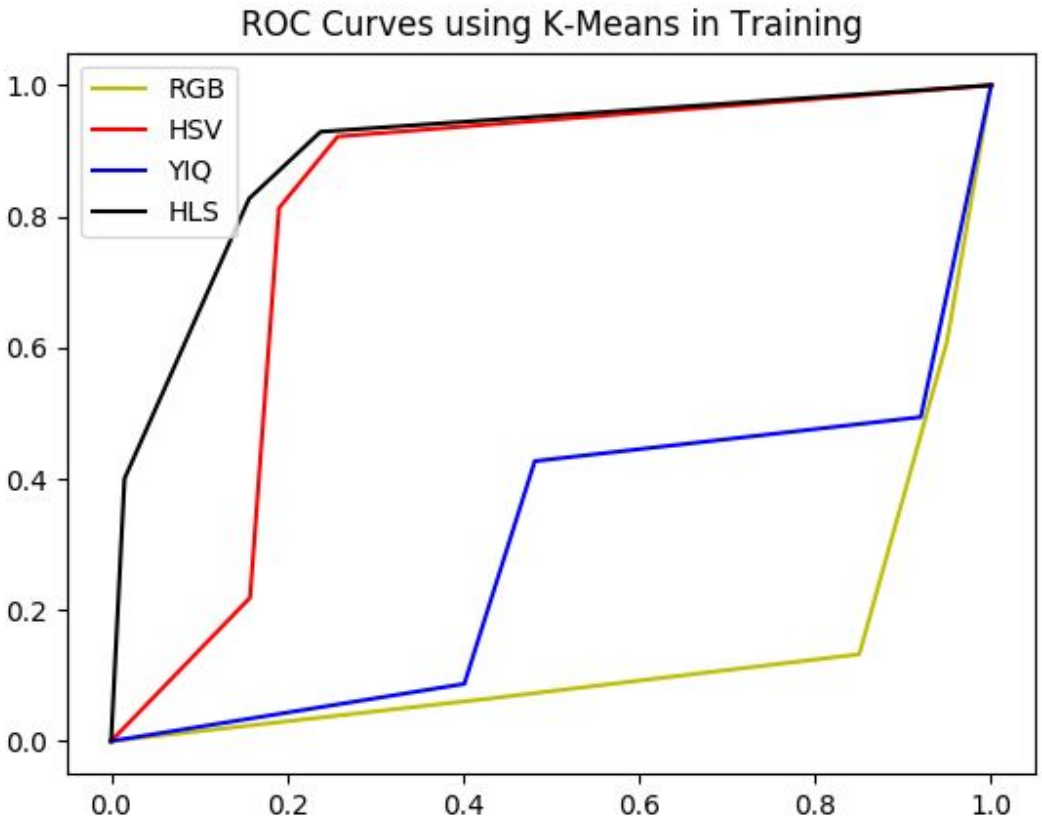

ROC Curves using FCM in Training
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
RGB
HSV
YIQ
HLS


ROC Curves using FCM in Validation
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
RGB
HSV
YIQ
HLS


ROC Curves using PCM in Validation
RGB
HSV
YIQ
HLS
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curves using PCM in Training
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
RGB
HSV
YIQ
HLS


ROC Curves using K-Means in Validation
RGB
HSV
YIQ
HLS
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


ROC Curves using K-Means in Training
RGB
HSV
YIQ
HLS
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0