Name: Rohit Kulkarni

USC ID: 5402749044

1. Supervised, Semi-Supervised, and Unsupervised Learning

1.b.i) Supervised Learning

Test Errors and Best C

```
Test errors : [0.878, 0.739, 0.835, 0.878, 0.765, 0.957, 0.852, 0.826, 0.957, 0.93, 0.93, 0.852, 0.922, 0.852, 0.939, 0.765, 0.896, 0.896, 0.72
2, 0.939, 0.765, 0.896, 0.896, 0.722, 0.939, 0.765, 0.896, 0.896, 0.722]
Best C : [{'C': 1000.0}, {'C': 10.0}, {'C': 100.0}, {'C': 10.0}, {'C': 10.0}, {'C': 100.0}, {'C': 100.0}, {'C': 10.0}, {'C': 1.0}, {'C': 10.0},
{'C': 10.0}, {'C': 100.0}, {'C': 1.0}, {'C': 1.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0},
{'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}]
```

Average Accuracy: 0.856072

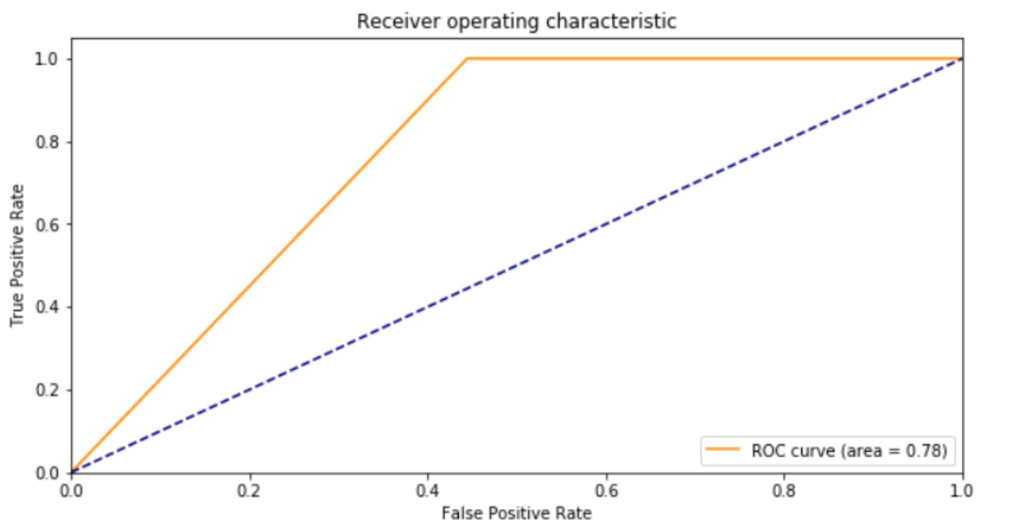
Average Precision: 0.856072

Average Recall: 0.856072

Average F score: 0.856072

Average AUC values: 0.880859

ROC Curve:



Confusion Matrix: [40, 32],

[0, 43]

1.b.ii) Semi-Supervised Learning/ Self-training

Test Errors and Best C

```
Test errors : [0.94, 0.944, 0.951, 0.937, 0.919, 0.947, 0.835, 0.958, 0.744, 0.972, 0.881, 0.923, 0.916, 0.874, 0.853, 0.898, 0.895, 0.961, 0.91
9, 0.972, 0.958, 0.965, 0.961, 0.853, 0.874, 0.874, 0.919, 0.919, 0.821, 0.965, 0.954, 0.916]
Best C : [{'C': 10000.0}, {'C': 1.0}, {'C': 1000000.0}, {'C': 1.0}, {'C': 10000.0}, {'C': 10.0}, {'C': 10.0}, {'C': 1.0}, {'C': 100.0}, {'C': 1
0.0}, {'C': 100000.0}, {'C': 100000.0}, {'C': 1.0}, {'C': 100000000.0}, {'C': 10.0}, {'C': 1.0}, {'C': 100000.0}, {'C': 1.0}, {'C': 1000000.0},
{'C': 10.0}, {'C': 10.0}, {'C': 1.0}, {'C': 1.0}, {'C': 10000000.0}, {'C': 10.0}, {'C': 10000.0}, {'C': 10.0}, {'C': 10.0}, {'C': 100.0}, {'C':
10000.0}, {'C': 1.0}, {'C': 100000.0}]
```

Average Accuracy: 0.947368

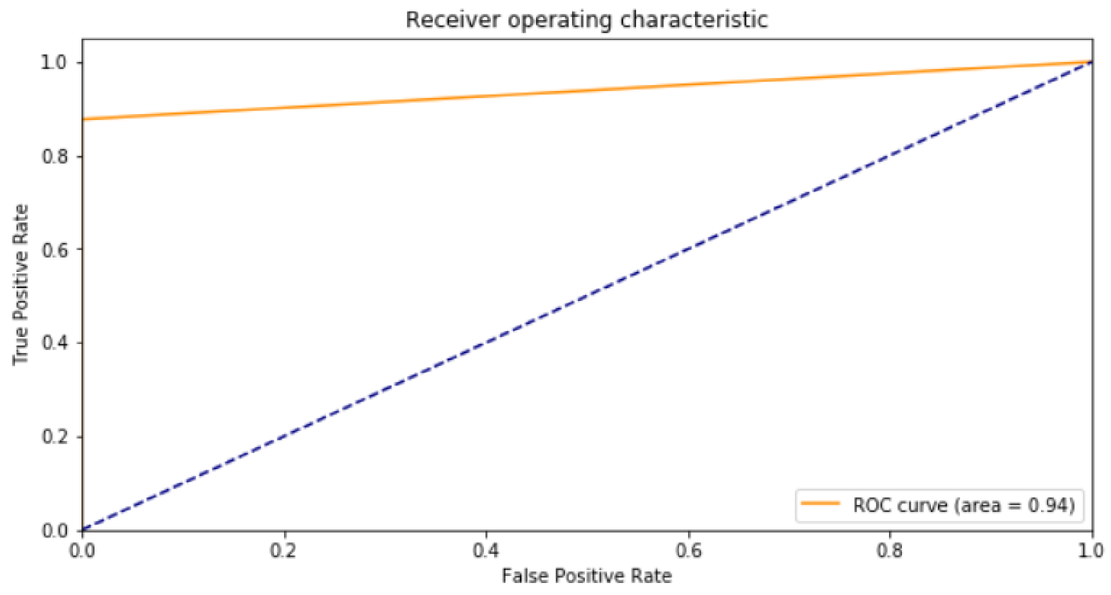
Average Precision: 0.947368

Average Recall: 0.947368

Average F score: 0.947368

Average AUC values: 0.940487

ROC Curve:



Confusion Matrix: [179,   0],

[13, 93]]

1.b.iii) Unsupervised Learning

To avoid algorithm to be trapped in the local minimum one way is:

K-means is sensitive to initial placement of cluster centers. So, with badly placed cluster centers, the algorithm can converge into a local minimum that isn't particularly useful, when the same analysis with better initial centers will generate a much better solution.

One solution is: We could try many random starting points

Alternative 1: We can use the largest minimum distance algorithm to determine K initial cluster focal points, and then we combine it with the traditional K-Means algorithm, at last, accomplish the classification of pattern congregation. The improved K-Means algorithm is obviously better than traditional one in aspects such as: the precision of cluster, the speed of cluster, stability and so on.

Alternative 2: We could try non-local split-and-merge moves: Simultaneously merge two nearby clusters and split a big cluster into two.

For train data

Average Accuracy: 0.849717

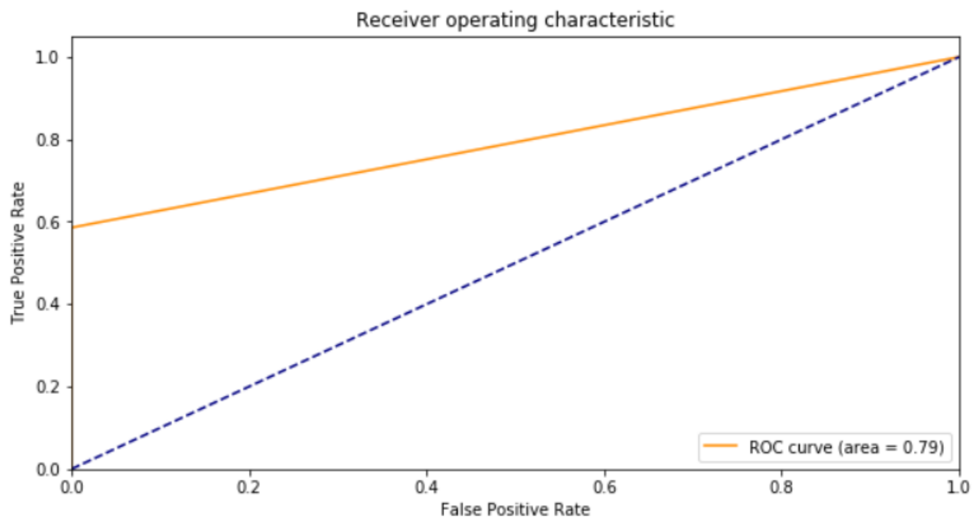
Average Precision: 0.849717

Average Recall: 0.849717

Average F score: 0.849717

Average AUC values: 0.799

ROC Curve:



Confusion Matrix: [285, 0],

[70, 99]

For test data

Average Accuracy: 0.850683

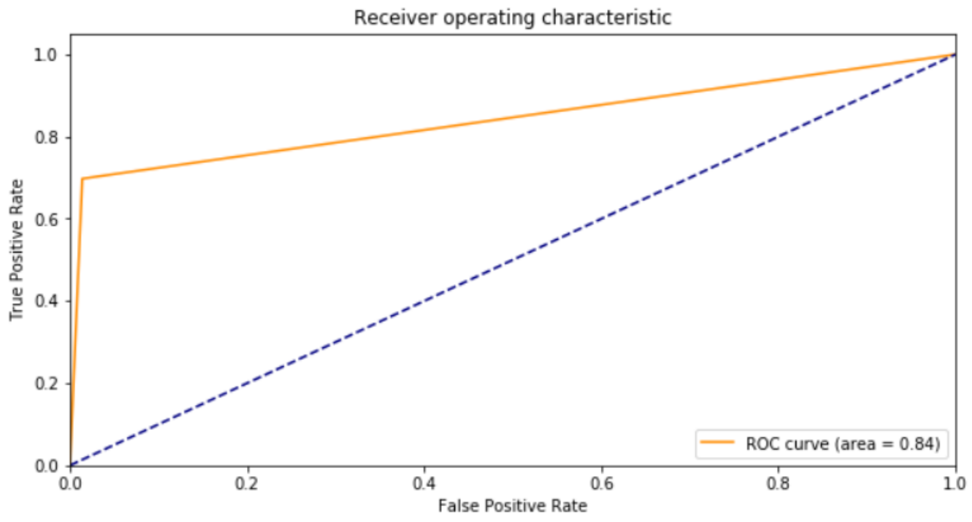
Average Precision: 0.850683

Average Recall: 0.850683

Average F score: 0.850683

Average AUC values: 0.801403

ROC Curve:



Confusion Matrix: [71, 1],

[13, 30]

1.b.iv) Spectral Clustering

In multivariate statistics and the clustering of data, spectral clustering techniques make use of the spectrum of the similarity matrix of the data to perform dimensionality reduction before clustering in fewer dimensions.

For train data

Average Accuracy: 0.856681

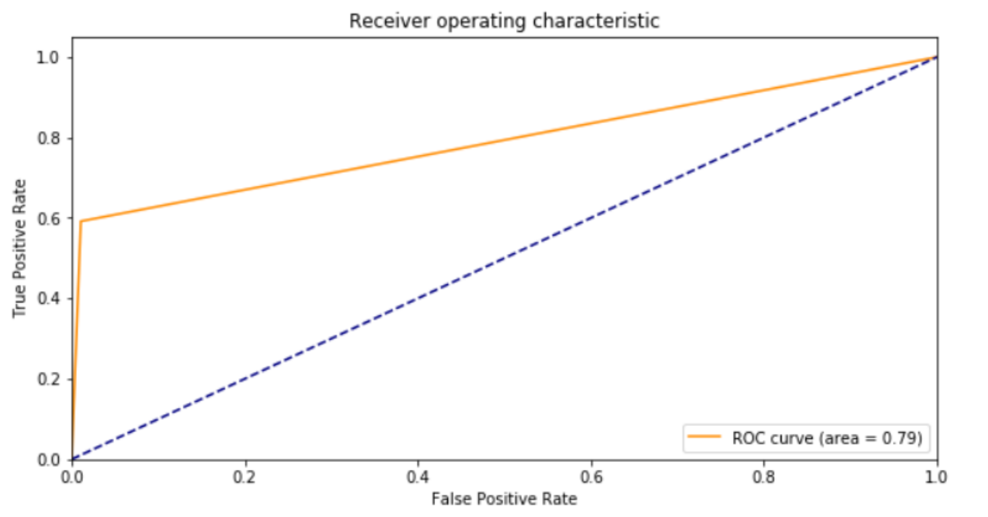
Average Precision: 0.856681

Average Recall: 0.856681

Average F score: 0.856681

Average AUC values: 0.809743

ROC Curve:



Confusion Matrix: [282,   3],

[69, 100]

For Test Data

Average Accuracy: 0.825217

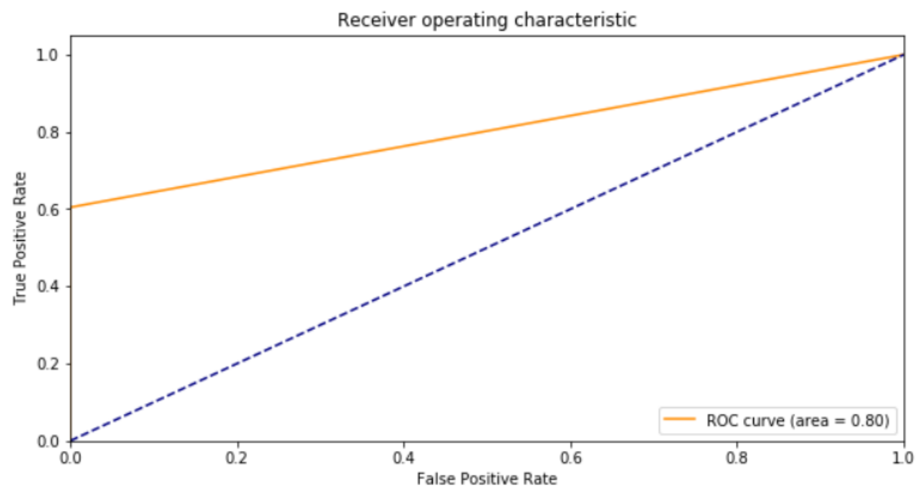
Average Precision: 0.825217

Average Recall: 0.825217

Average F score: 0.825217

Average AUC values: 0.76784

ROC Curve:



Confusion Matrix: [72, 0],

[17, 26]

From all the above accuracy scores we can Semi-Supervised Learning/ Self-training has best fitted the data compared to others.

2. Active Learning Using Support Vector Machines

2.b.i) Passive learning

Test errors and Best C

```
Test errors : [[0.909, 0.909, 0.977, 0.975, 0.975, 0.981, 0.981, 0.983, 0.983], [0.966, 0.966, 0.968, 0.968, 0.968, 0.981, 0.987, 0.985, 0.985],
[0.983, 0.983, 0.985, 0.989, 0.987, 0.987, 0.987, 0.987, 0.987], [0.987, 0.987, 0.987, 0.987, 0.992, 0.992, 0.987, 0.989, 0.989], [0.992, 0.992,
0.989, 0.987, 0.987, 0.987, 0.987, 0.987, 0.987]]
Best C : [{'C': 10.0}, {'C': 10.0}, {'C': 0.1}, {'C': 10.0}, {'C': 10.0}, {'C': 1.0}, {'C': 10.0}, {'C': 1.0}, {'C': 1.0}]
```

2.b.ii) Active learning

Test errors and Best C

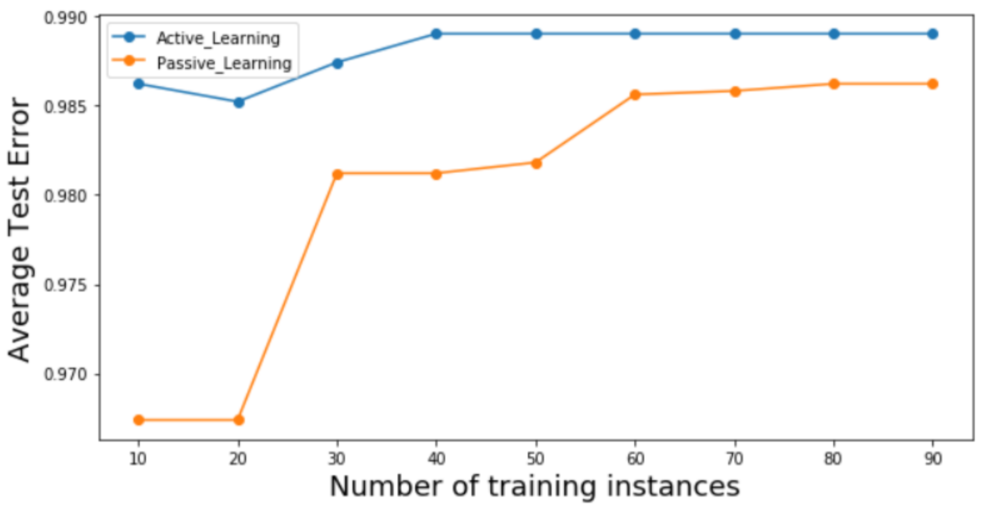
```
Test errors : [[0.975, 0.97, 0.981, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989], [0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989],
[0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989], [0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989], [0.989, 0.989,
0.989, 0.989, 0.989, 0.989, 0.989, 0.989, 0.989]]
Best C : [{'C': 0.1}, {'C': 0.1}, {'C': 0.1}, {'C': 0.1}, {'C': 0.1}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}, {'C': 10.0}]
```

2.b.iii)

Average errors from both Passive and Active learning

```
Average of 50 errors from passive learning 0    0.944333
1    0.990000
2    0.989556
3    0.987444
4    0.987000
dtype: float64
Average of 50 errors from active learning 0    0.935111
1    0.979000
2    0.979000
3    0.979000
4    0.979000
dtype: float64
```

Plotting average test error versus number of training instances for both active and passive


Active_Learning
Passive_Learning
Average Test Error
Number of training instances