Prashanth Manja

USC ID: 3073150764

ML- Assignment -5

1. Supervised, Semi-Supervised, and Unsupervised Learning

Attribute Information:

Given is the variable name, variable type, the measurement unit and a brief description. The "Blood Transfusion Service Center" is a classification problem. The order of this listing corresponds to the order of numerals along the rows of the database.

R (Recency - months since last donation),
F (Frequency - total number of donation),
M (Monetary - total blood donated in c.c.),
T (Time - months since first donation), and
a binary variable representing whether he/she donated blood in March 2007 (1 stand for donating blood; 0 stands for not donating blood).

Check for missing data:

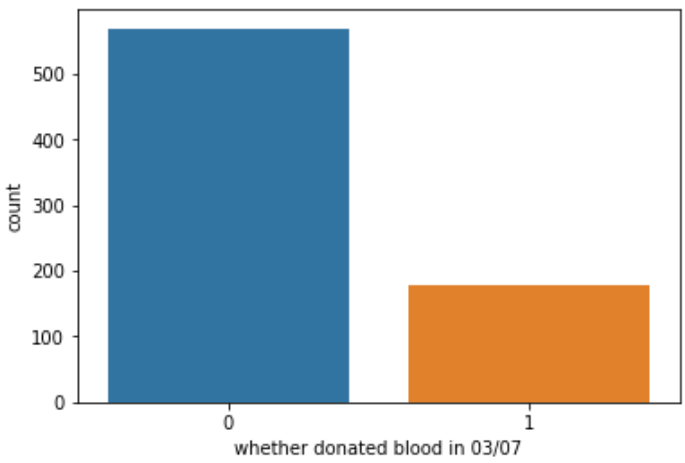
```
RangeIndex: 748 entries, 0 to 747
Data columns (total 5 columns):
Recency (months)                    748 non-null int64
Frequency                           748 non-null int64
Monetary (c.c. blood)               748 non-null int64
Time (months)                       748 non-null int64
whether donated blood in 03/07      748 non-null int64
dtypes: int64(5)
memory usage: 29.3 KB
```

So, the dataset has a total of 748 data points and no missing data points.

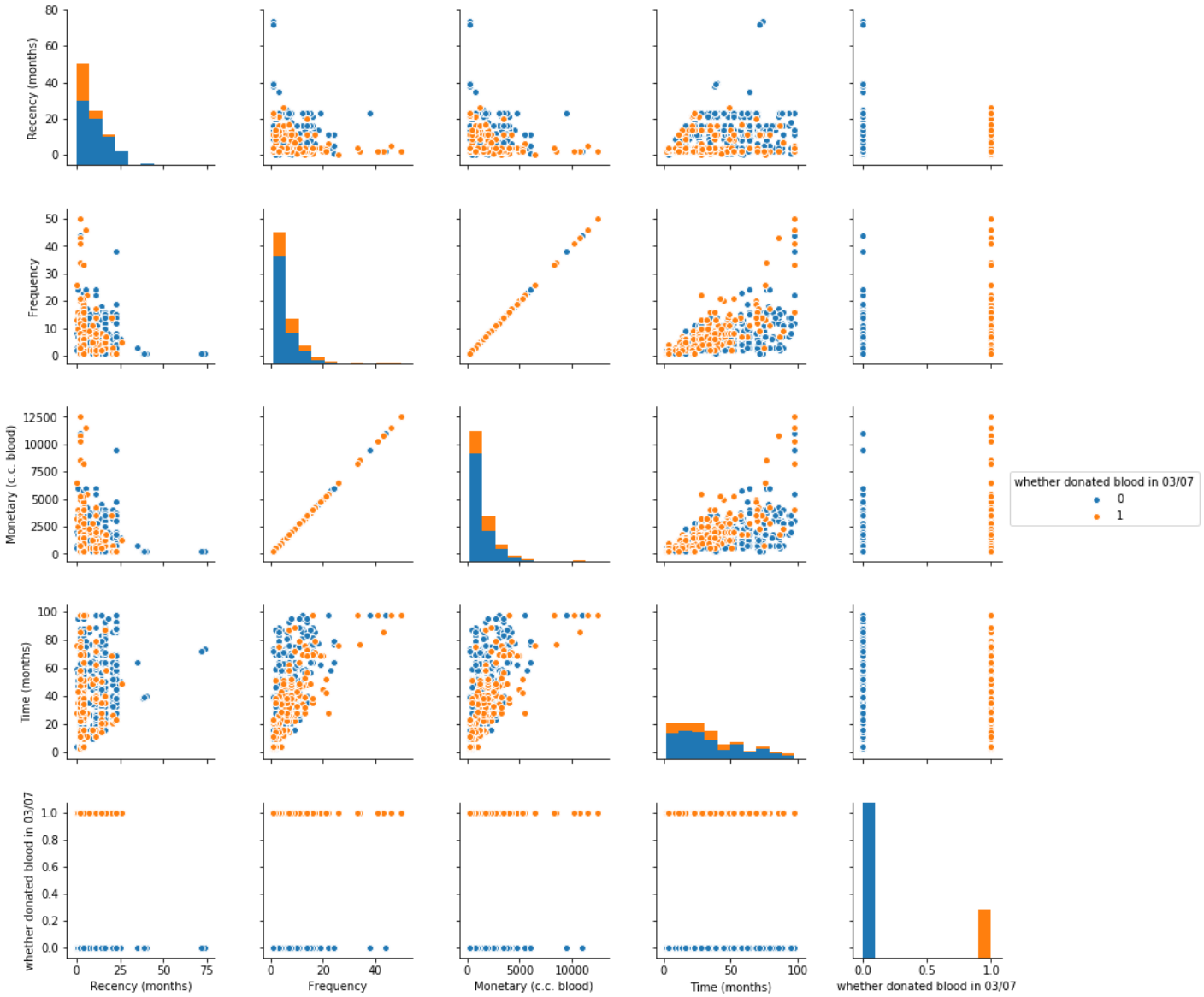
Number of features for each class in the training set:

```
0    570
1    178
```

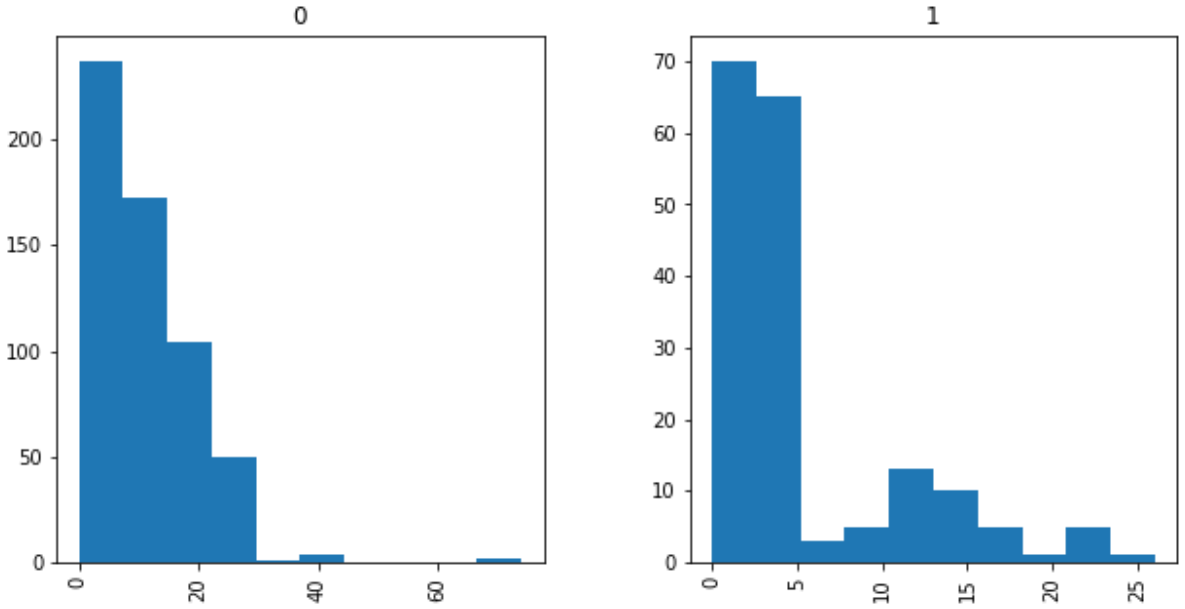
('Number of people who donated in march: '**178**)
('Number of people who didn't donated in march: **570**)
('Total number: **748**)

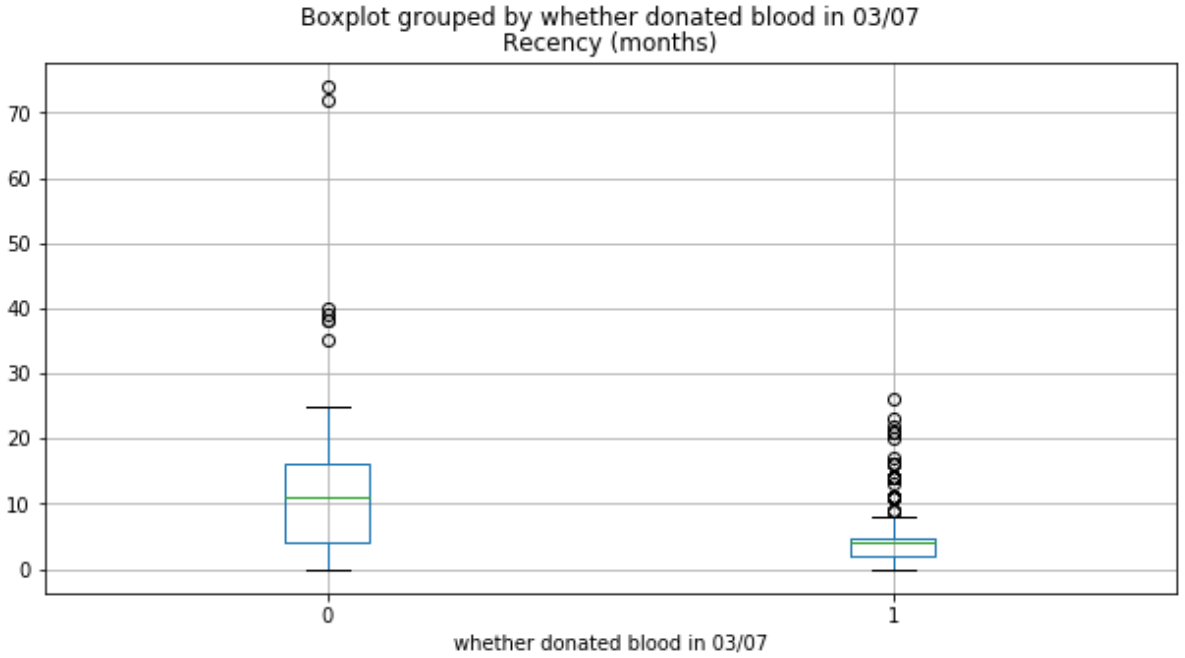


Pairplot :

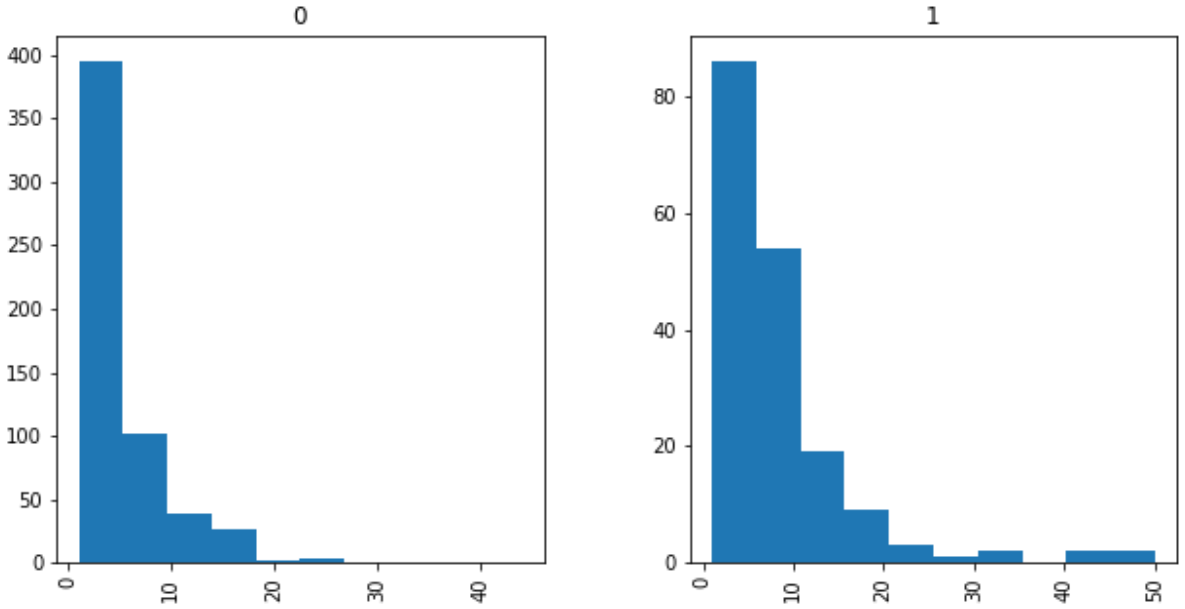

Recency (months)
Frequency
Monetary (c.c. blood)
Time (months)
whether donated blood in 03/07
whether donated blood in 03/07
0
1

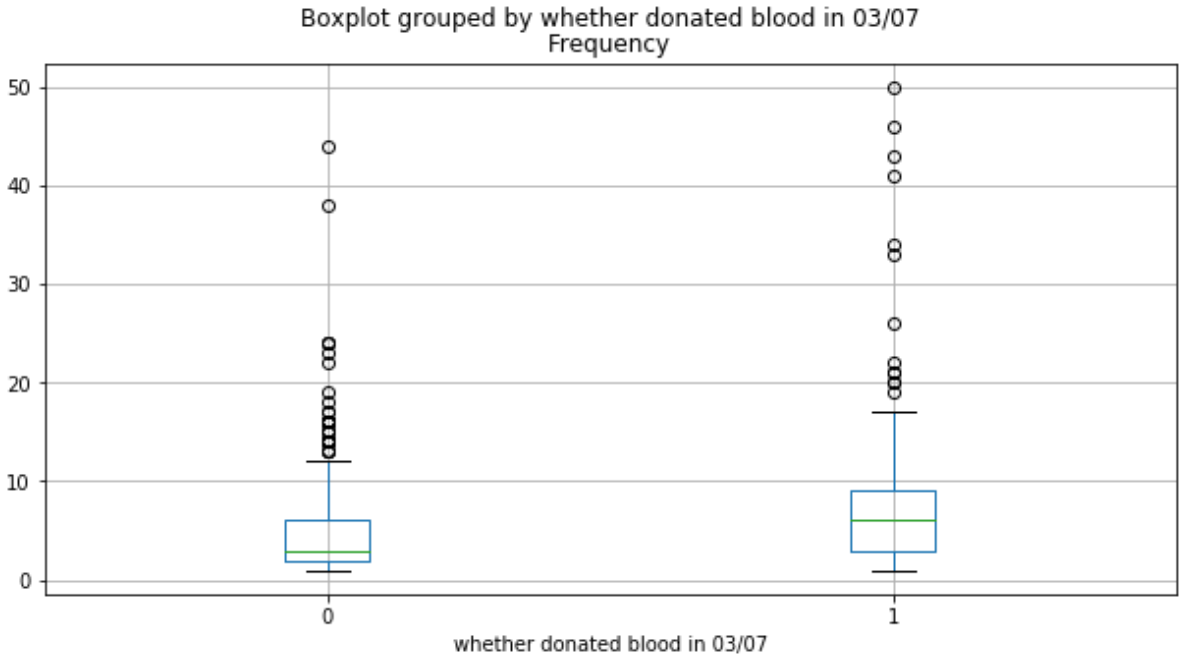
Recency vs Class(whether donated blood in 03/07):



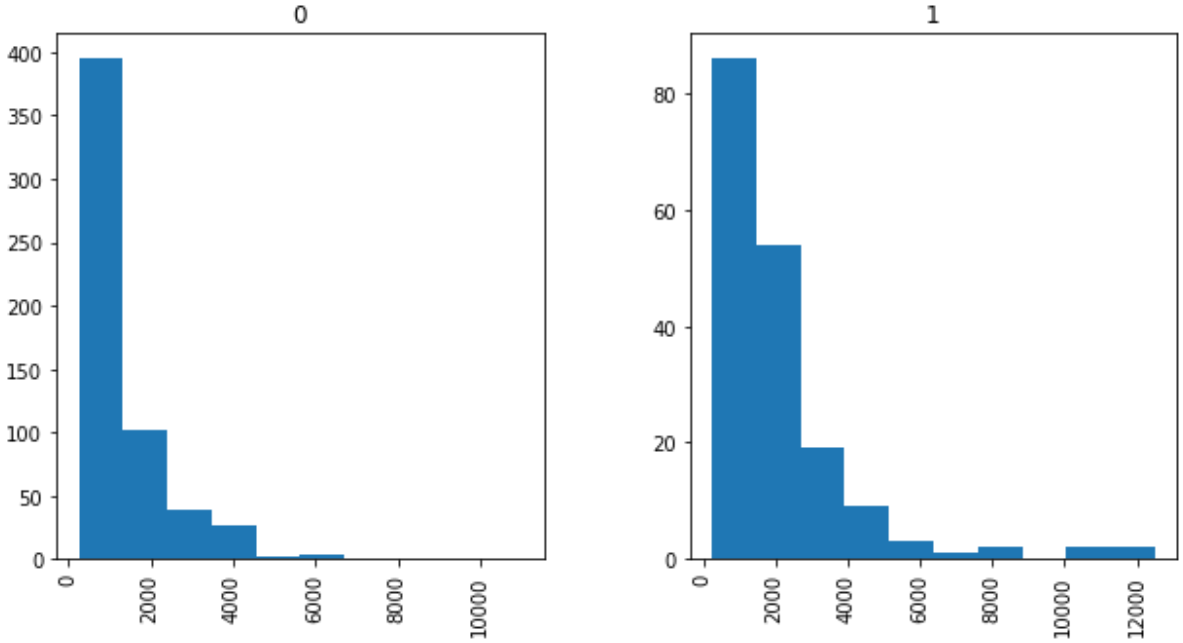


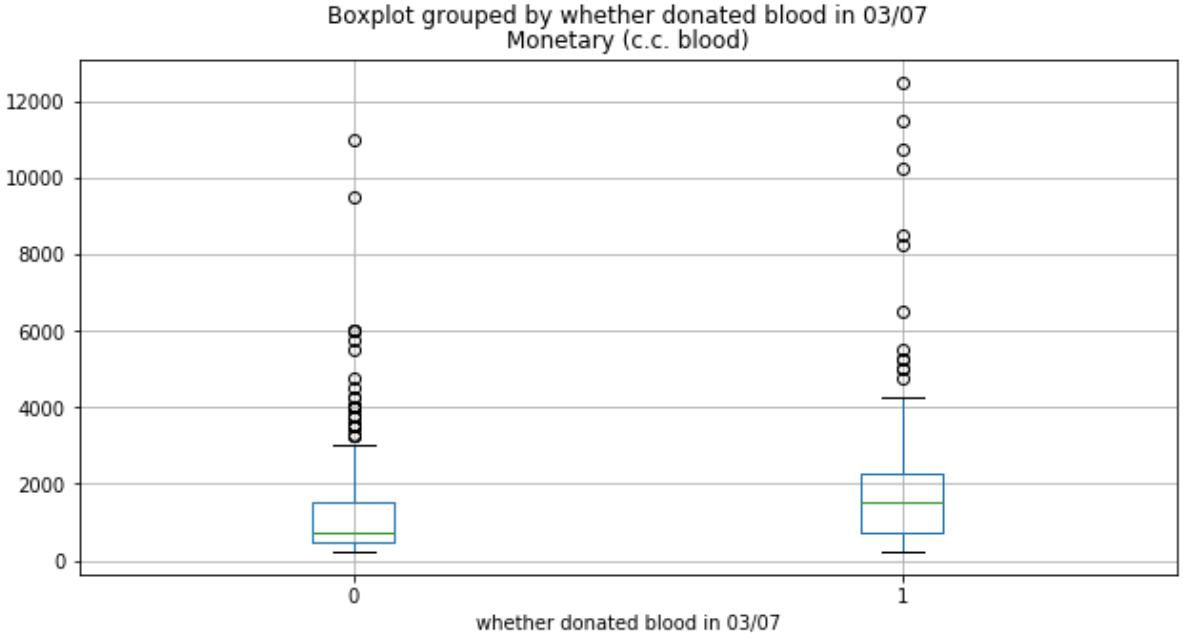
Frequency vs Class(whether donated blood in 03/07):



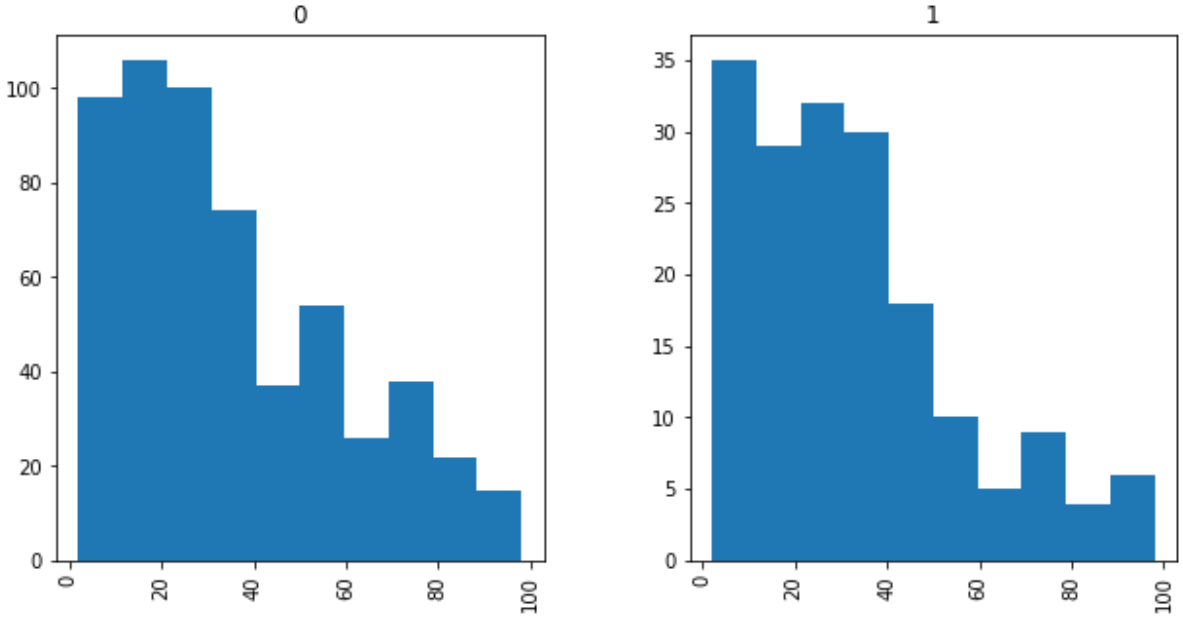


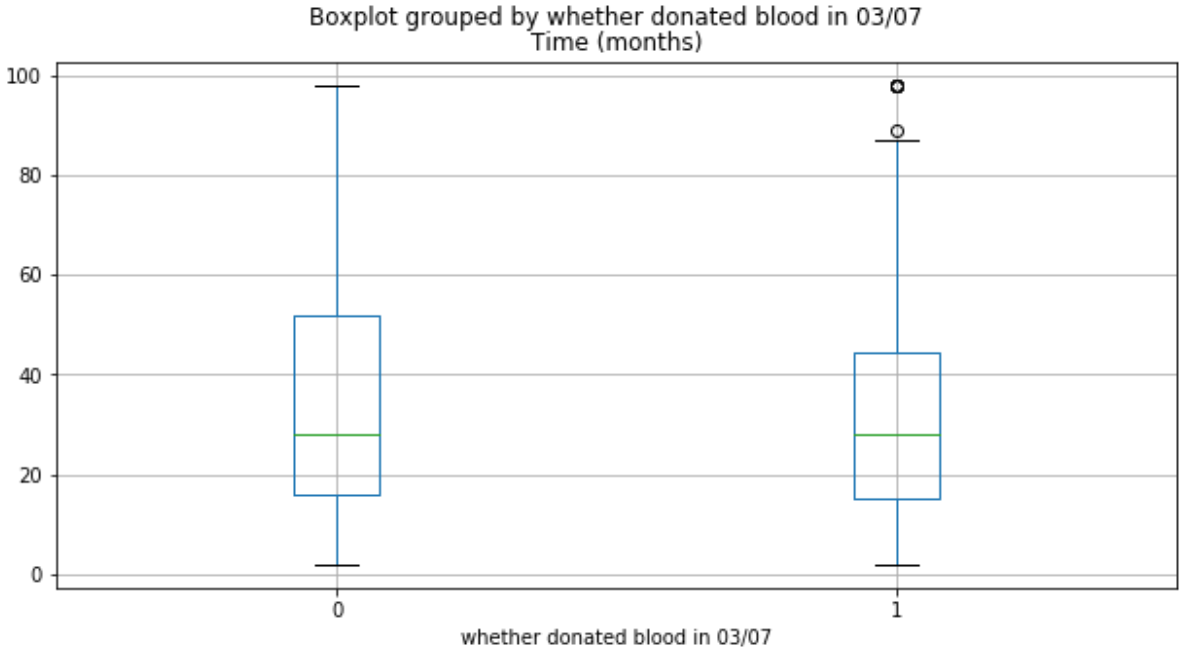
Monetary (c.c. blood) vs Class(whether donated blood in 03/07)





Time (months) vs Class(whether donated blood in 03/07)

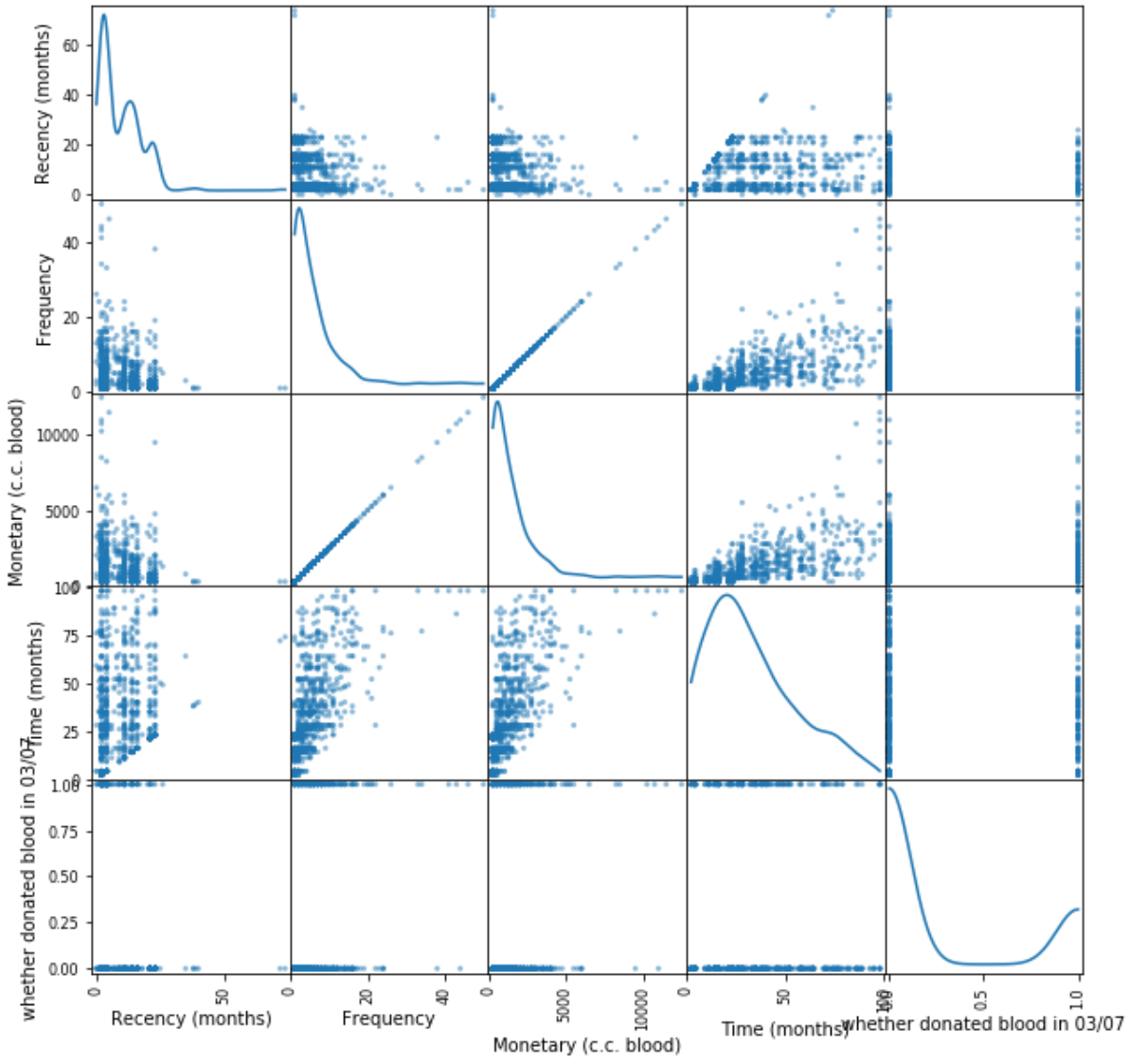




Pivot Table: (mean)

whether donated blood in 03/07	Frequency	Monetary (c.c. blood)	Recency (months)	Time (months)
0	4.801754	1200.438596	10.771930	34.770175
1	7.797753	1949.438202	5.455056	32.719101

Check the correlation between the features and the label:



Test data: (head)(first 20% of positive and negative class)

	Recency (months)	Frequency	Monetary (c.c. blood)	Time (months)	whether donated blood in 03/07
373	9	4	1000	65	0
454	21	1	250	21	0
455	21	1	250	21	0
456	21	1	250	21	0
457	21	1	250	21	0

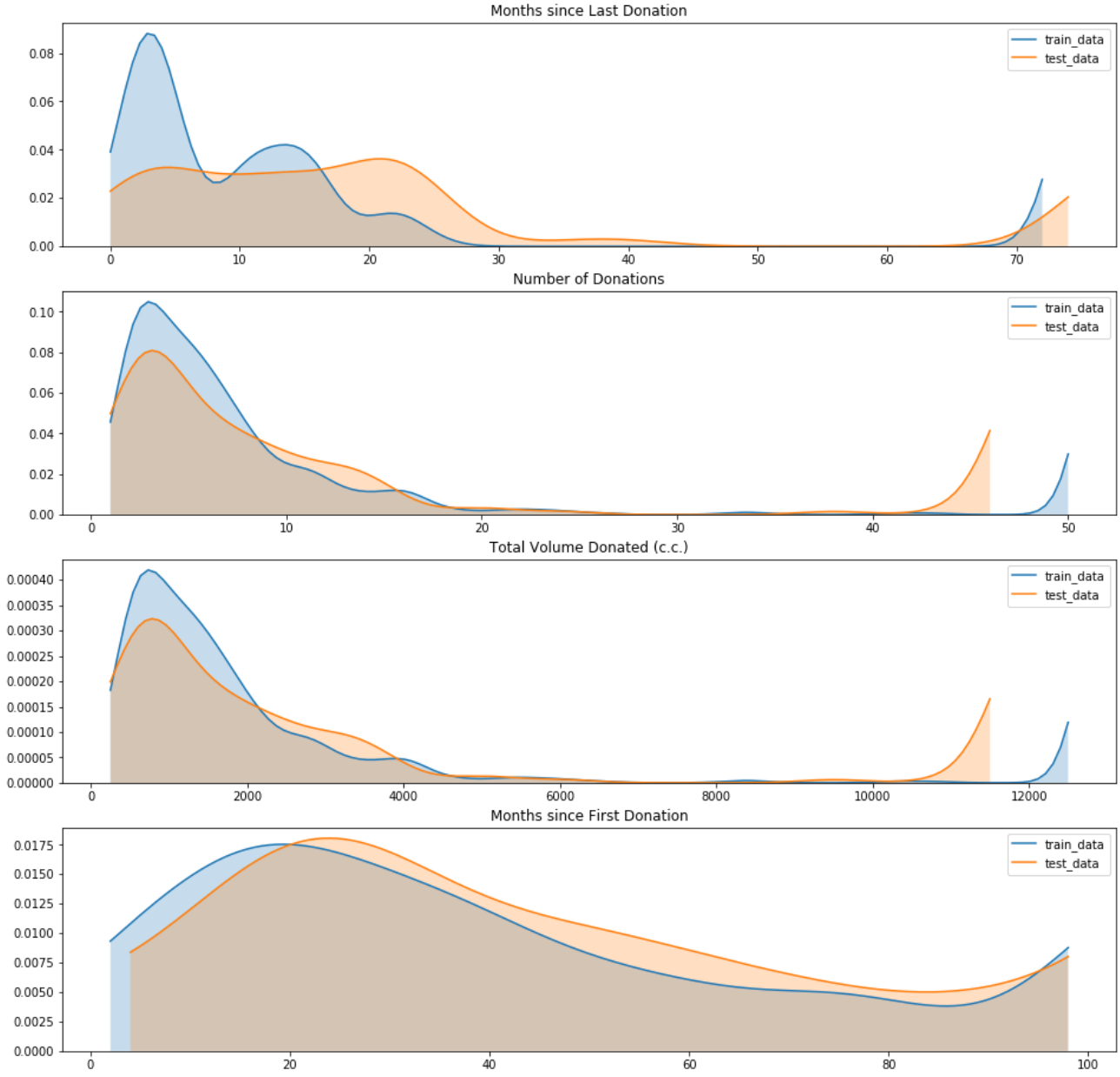
```
(150, 5)
```

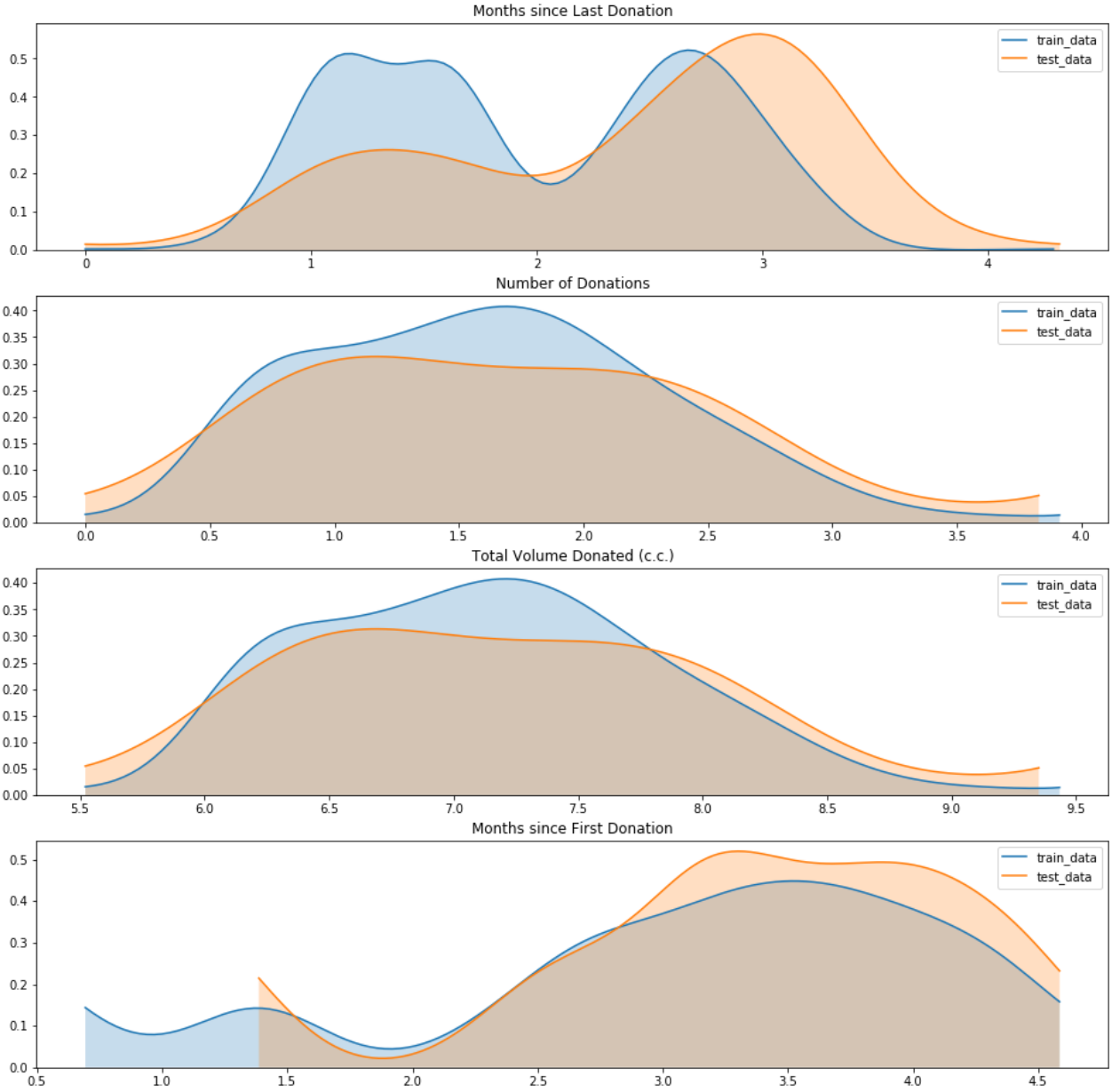
Train data: (head)

```
(598, 5)
```

	Recency (months)	Frequency	Monetary (c.c. blood)	Time (months)	whether donated blood in 03/07
0	2	50	12500	98	1
4	1	24	6000	77	0
5	4	4	1000	4	0
7	1	12	3000	35	0
10	4	23	5750	58	0

Train Vs Test:




Months since Last Donation
train_data
test_data
Number of Donations
train_data
test_data
Total Volume Donated (c.c.)
train_data
test_data
Months since First Donation
train_data
test_data

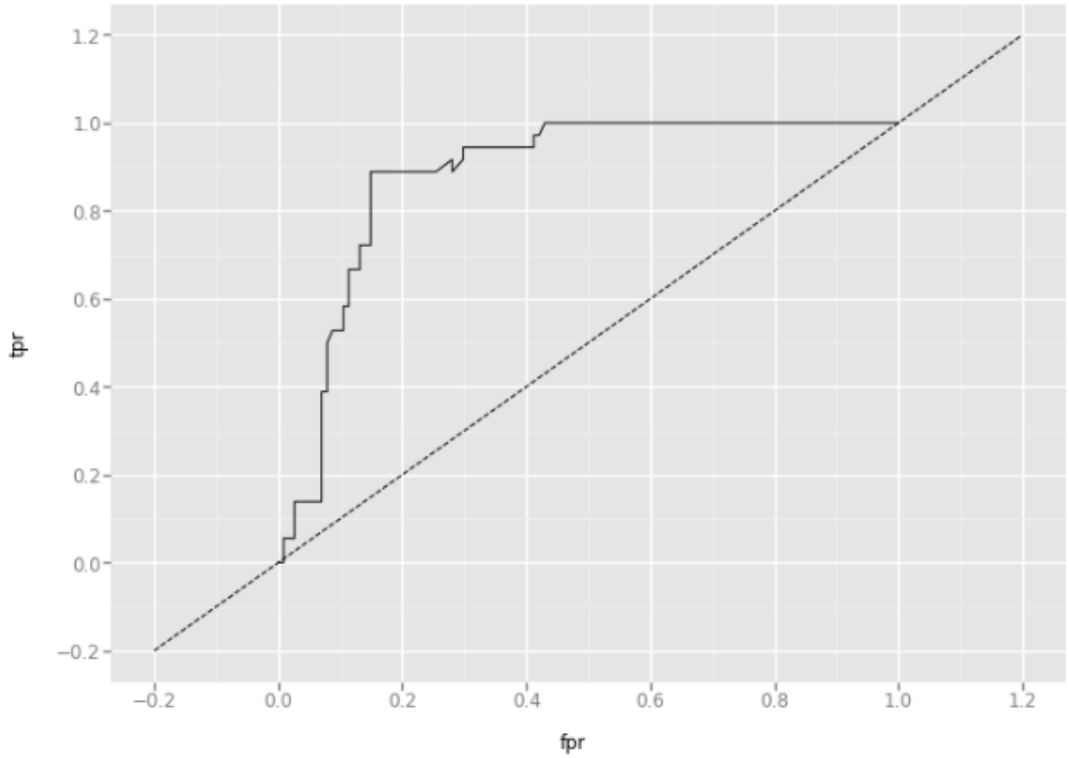
Supervised Learning:

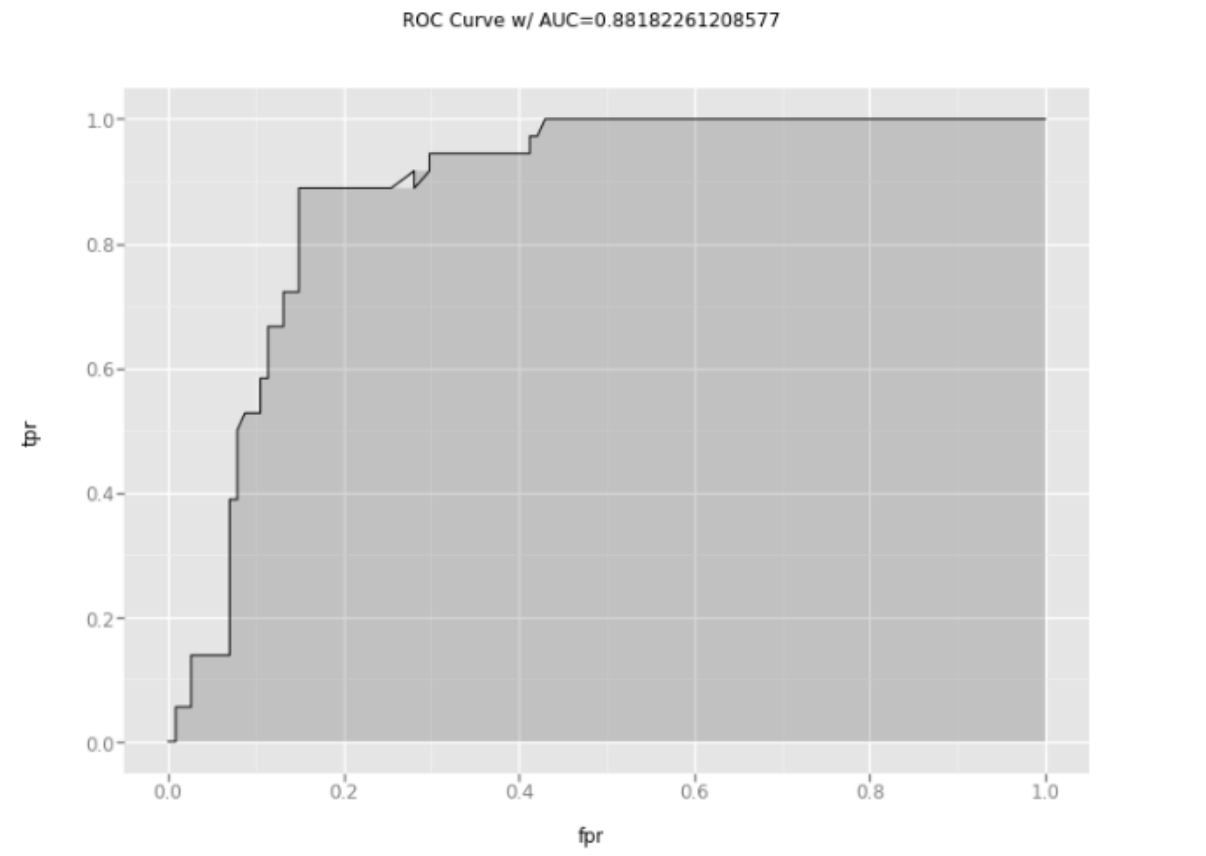
1. Accuracy_Score : 0.76
2. Confusion Matrix : array([[113, 1],
                             [35, 1]],

3. Precision_recall:

	precision	recall	f1-score	support
0	0.76	0.99	0.86	114
1	0.50	0.03	0.05	36
avg / total	0.70	0.76	0.67	150

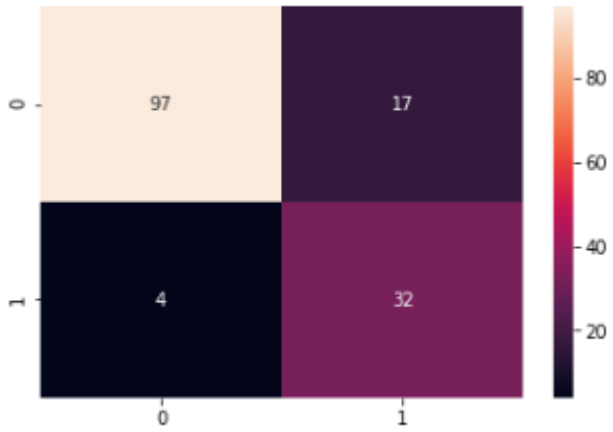
4. ROC-AUC: 0.88182261208577





Find the optimal threshold and tuning the P:

Optimal Threshold Value: 0.19618488029573874



So here we observe that by tuning the default P (0.5) value for predict function We observe an improved Confusion Matrix.

2. Semi-Supervised Learning/ Self-training:

1. Before adding the closest unlabelled data one by one:

Accuracy_Score : 0.7625418060200669

Confusion Matrix:

```
array([[228,   0],
       [ 71,   0]]
```

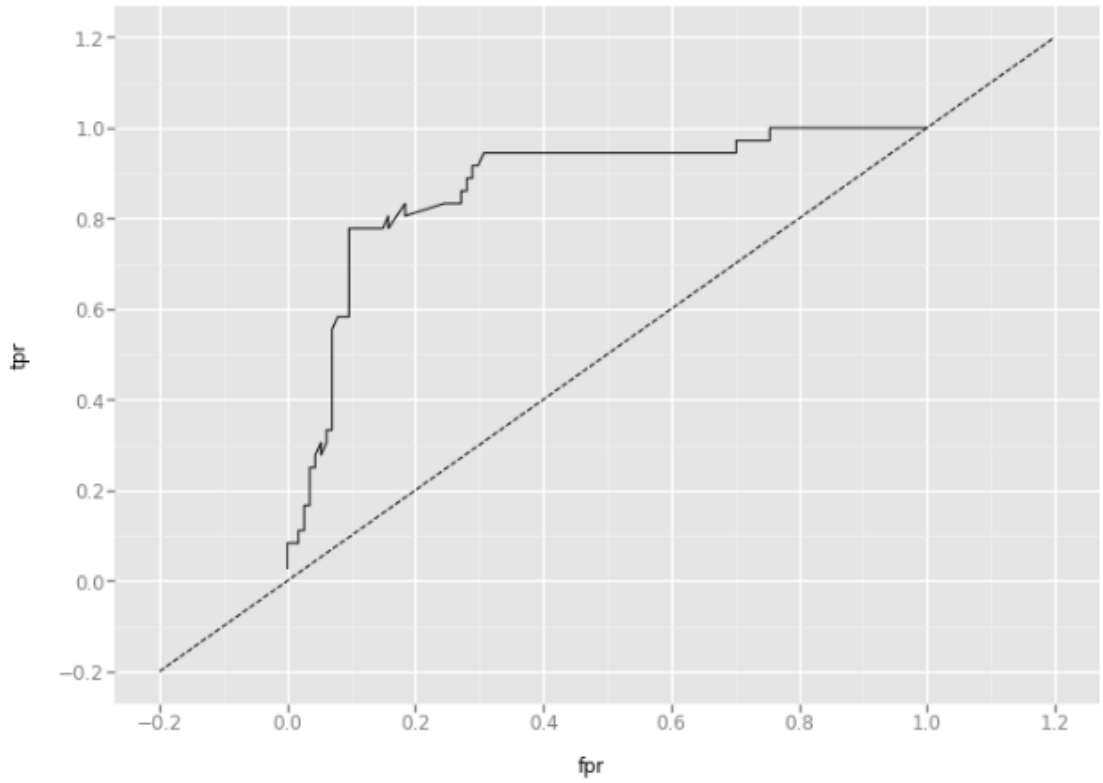
2. After adding the closest unlabelled data one by one:

Accuracy_Score :0.7666666666666667

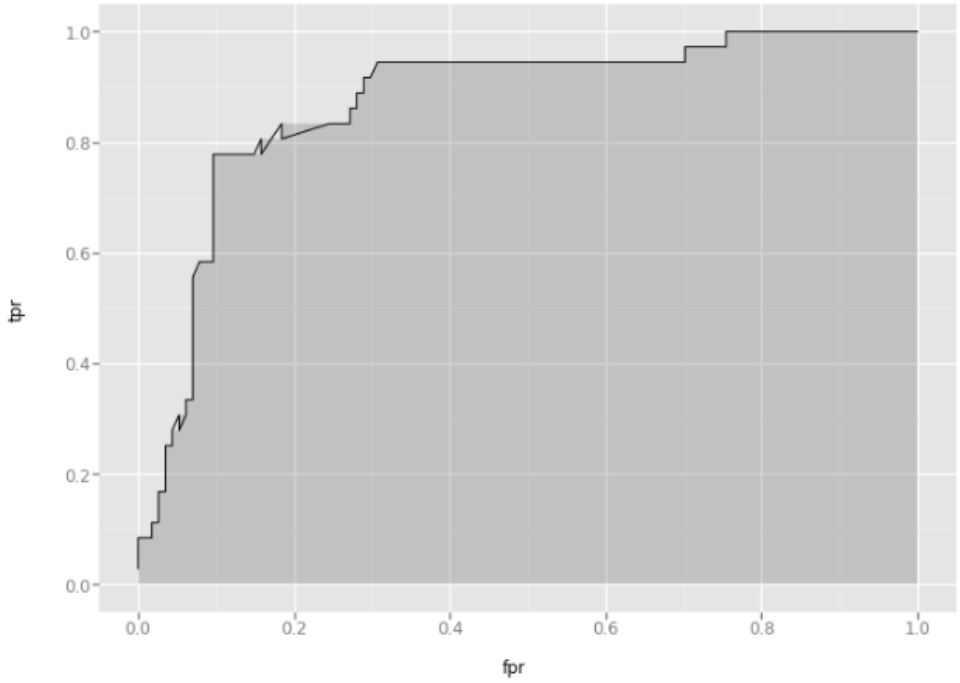
Confusion Matrix:

```
array([[114,   0],
       [ 35,   1]]
```

	precision	recall	f1-score	support
0	0.77	1.00	0.87	114
1	1.00	0.03	0.05	36
avg / total	0.82	0.77	0.67	150



ROC Curve w/ AUC=0.8725633528265108



3. Unsupervised Learning:

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

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

Coordinates of cluster centres:

```
[[8.64509804e+00 3.67254902e+00 9.18137255e+02 2.69627451e+01]
 [6.85227273e+00 1.55340909e+01 3.88352273e+03 6.51477273e+01]]
```

Closest data-point to centres C1 and C2 (indexes are) : `[495  20]`

The majority poll becomes the label predicted by k-means for the members of each cluster. Then comparing the labels provided by k-means with the true labels of the training data and report accuracy and the confusion matrix.

Accuracy score: 0.7625418060200669

Confusion Matrix: array([[456,   0],
[142,   0]],

Classifying test data:

Accuracy score: 0.76

Confusion Matrix: array([[114,   0],
[36,   0]]

One expects that supervised learning on the full data set works better than semi-supervised learning with half of the data set labelled. –This holds to be true as per the results obtained from above

One expects that unsupervised learning underperforms in such situations.

K-Means Clustering on Multi-Class and Multi-Label DataSet

Anuran Calls (MFCCs) Data Set

This dataset was used in several classifications tasks related to the challenge of anuran species recognition through their calls.

It is a multilabel dataset with three columns of labels. This dataset was created segmenting 60 audio records belonging to 4 different families, 8 genus, and 10 species.

Each audio corresponds to one specimen (an individual frog), the record ID is also included as an extra column.

The amount of instances per class are:

Families:
Bufonidae 68
Dendrobatidae 542
Hylidae 2165
Leptodactylidae 4420

Genus:
Adenomera 4150
Ameerega 542
Dendropsophus 310
Hypsiboas 1593
Leptodactylus 270
Osteocephalus 114
Rhinella 68
Scinax 148

Species:
AdenomeraAndre 672
AdenomeraHylaedactâ€¦ 3478
Ameeregatrivittata 542
HylaMinuta 310
HypsiboasCinerascens 472
HypsiboasCordobae 1121
LeptodactylusFuscus 270
OsteocephalusOophaâ€¦ 114
Rhinellagranulosa 68
ScinaxRuber 148

Each instance has three labels: Families, Genus, and Species. Each of the labels has multiple classes. We wish to solve a multi-class and multi-label problem.

Data Pre-processing:

Count of Each Class in Each of the Label:

1. Family -Label

```
Leptodactylidae    3075
Hylidae            1518
Dendrobatidae       394
Bufonidae            50
```

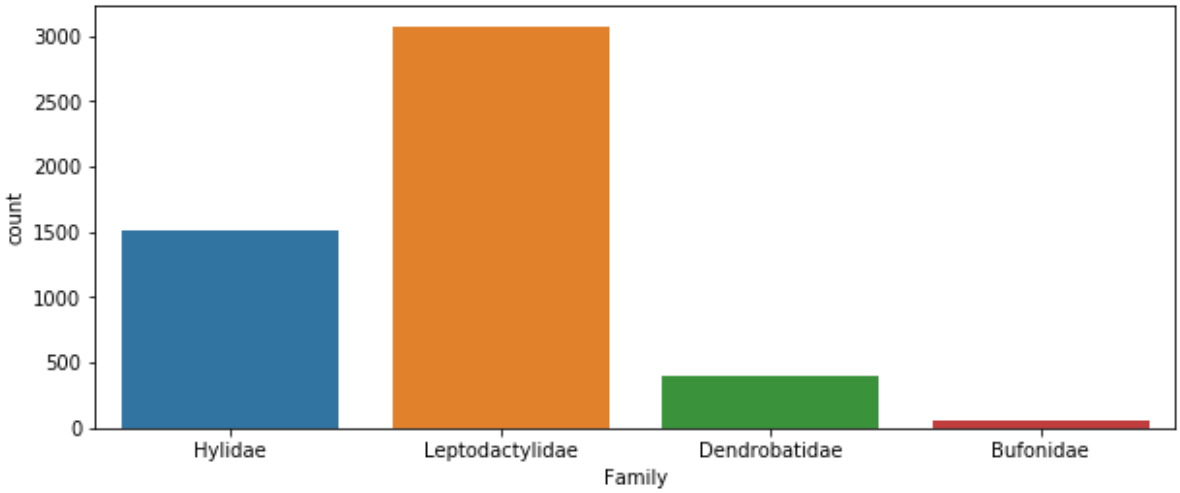
2. Genus-Label

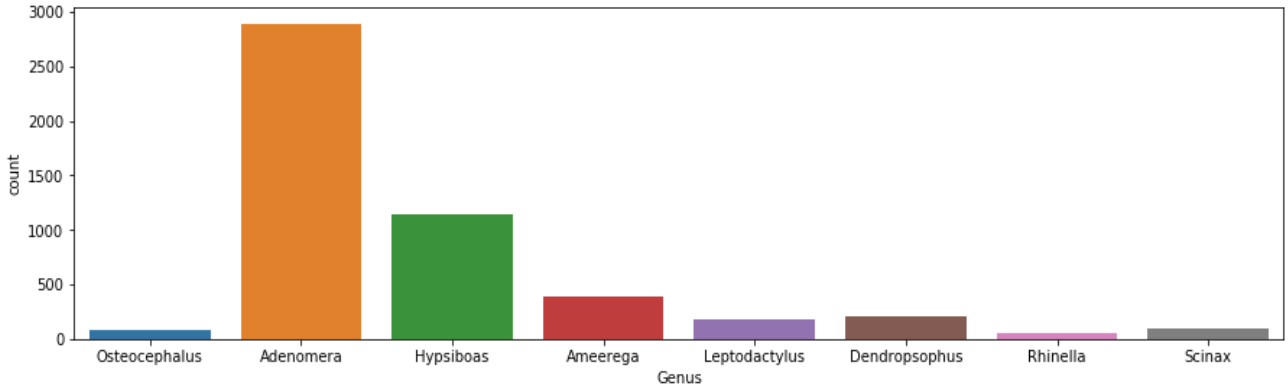
```
Adenomera         2891
Hypsiboas         1138
Ameerega           394
Dendropsophus      208
Leptodactylus      184
Scinax              90
Osteocephalus       82
Rhinella            50
```

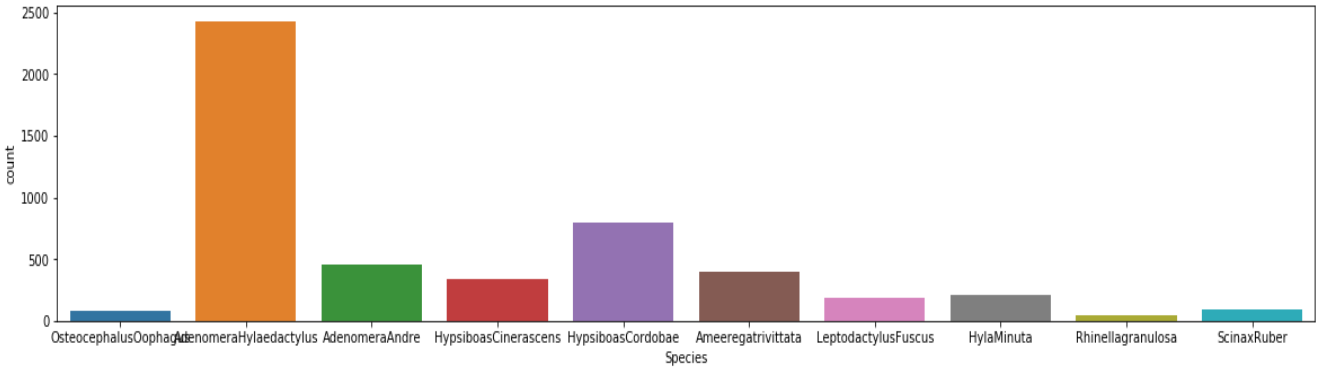
3. Species-Label

```
AdenomeraHylaedactylus    2430
HypsiboasCordobae          801
AdenomeraAndre             461
Ameeregatrivittata         394
HypsiboasCinerascens       337
HylaMinuta                 208
LeptodactylusFuscus        184
ScinaxRuber                 90
OsteocephalusOophagus       82
Rhinellagranulosa           50
```

Understanding Data Further

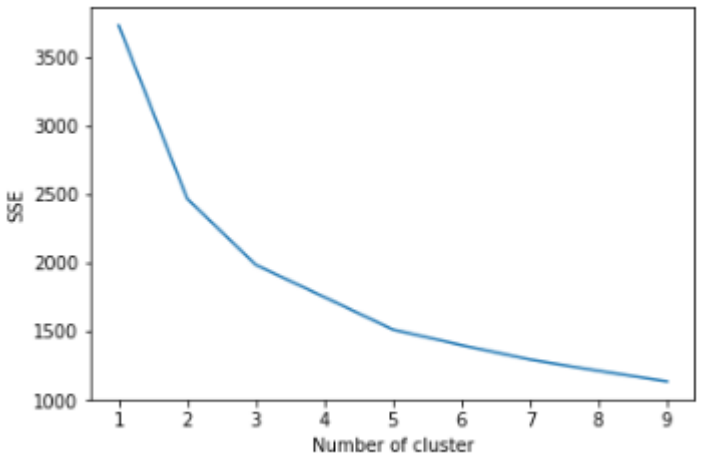






Determine the number of clusters:

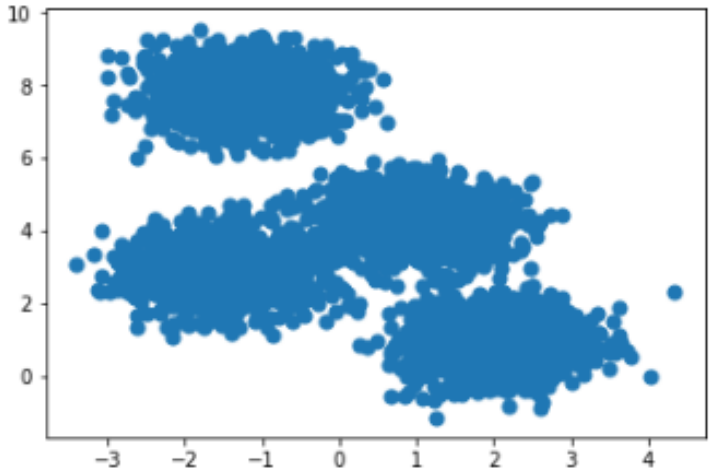
1. **Elbow Method:**



2. **Silhouette Coefficient**

```
For n_clusters=2, The Silhouette Coefficient is 0.348677841028
For n_clusters=3, The Silhouette Coefficient is 0.367692708115
For n_clusters=4, The Silhouette Coefficient is 0.378750934331
For n_clusters=5, The Silhouette Coefficient is 0.371653028394
For n_clusters=6, The Silhouette Coefficient is 0.264008021117
For n_clusters=7, The Silhouette Coefficient is 0.260615450976
For n_clusters=8, The Silhouette Coefficient is 0.270172962249
For n_clusters=9, The Silhouette Coefficient is 0.276188844822
For n_clusters=10, The Silhouette Coefficient is 0.281115232943
```

Since we observe that Silhouette Coefficient value is maximum when **cluster value = 4**, we choose.



Determine which class in each label is the majority by reading the true labels.

Cluster_1:

pred_Family	pred_Genus	pred_Species
Hylidae	Hypsiboas	HypsiboasCordobae
Hylidae	Hypsiboas	HypsiboasCordobae
Hylidae	Hypsiboas	HypsiboasCordobae
Hylidae	Hypsiboas	HypsiboasCordobae
Hylidae	Hypsiboas	HypsiboasCordobae

Cluster_2:

pred_Family	pred_Genus	pred_Species
Leptodactylidae	Adenomera	AdenomeraHylaedactylus
Leptodactylidae	Adenomera	AdenomeraHylaedactylus
Leptodactylidae	Adenomera	AdenomeraHylaedactylus
Leptodactylidae	Adenomera	AdenomeraHylaedactylus
Leptodactylidae	Adenomera	AdenomeraHylaedactylus

Cluster_3:

pred_Family	pred_Genus	pred_Species
Dendrobatidae	Ameerega	Ameeregatrivittata
Dendrobatidae	Ameerega	Ameeregatrivittata
Dendrobatidae	Ameerega	Ameeregatrivittata
Dendrobatidae	Ameerega	Ameeregatrivittata
Dendrobatidae	Ameerega	Ameeregatrivittata

Cluster_4:

pred_Family	pred_Genus	pred_Species
Hylidae	Hypsiboas	HypsiboasCinerascens
Hylidae	Hypsiboas	HypsiboasCinerascens
Hylidae	Hypsiboas	HypsiboasCinerascens
Hylidae	Hypsiboas	HypsiboasCinerascens
Hylidae	Hypsiboas	HypsiboasCinerascens

Hamming loss

The Hamming loss is the fraction of labels that are incorrectly predicted.

In multiclass classification, the Hamming loss correspond to the Hamming distance between **y_true** and **y_pred** which is equivalent to the subset **zero_one_loss** function.

The Hamming loss is upperbounded by the subset zero-one loss. When normalized over samples, the Hamming loss is always between 0 and 1.

In multilabel classification, the Hamming loss is different from the subset zero-one loss. The zero-one loss considers the entire set of labels for a given sample incorrect if it does entirely match the true set of labels. Hamming loss is more forgiving in that it penalizes the individual labels.

```
Average Hamming Distance of Cluster 0: 0.444836865119408
Average Hamming Distance of Cluster 1: 0.028494020926756354
Average Hamming Distance of Cluster 2: 0.5150339476236664
Average Hamming Distance of Cluster 3: 0.14006514657980457
```

10.6.2 :-

(a). We already have :

	0.3	0.4	0.7
0.3		0.5	0.8
0.4	0.5		0.45
0.7	0.8	0.45	

Step 2 : i = 4 : We have i = 4 : We may see that 0.3 is the minimum dissimilarity, so we fuse observations 1 and 2 to form cluster (1, 2) at hight 0.3 . We now have the new dissimilarity matrix.

	0.5	0.8
0.5		0.45
0.8	0.45	

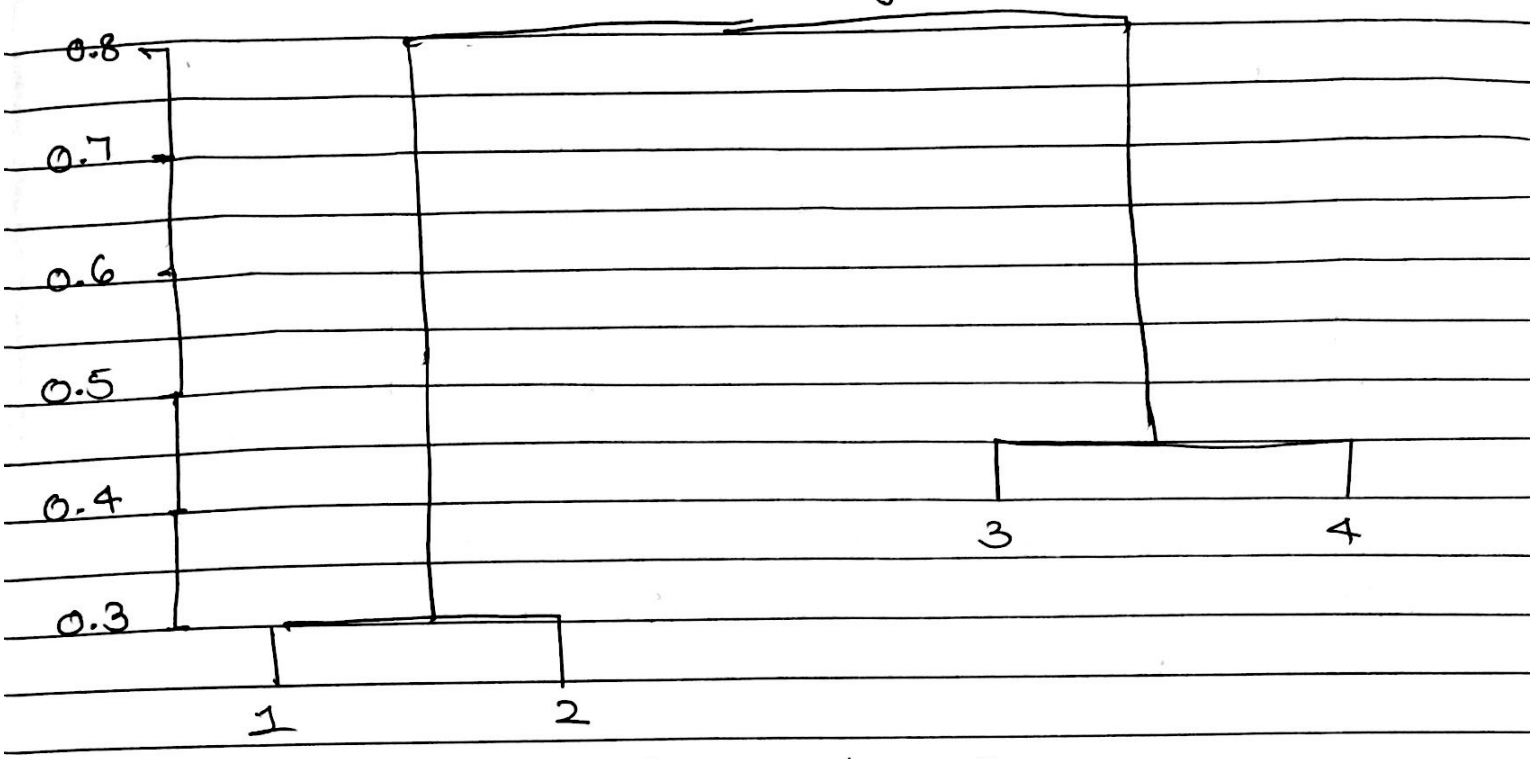
i = 3 : The min. dissimilarity is 0.45, so we merge observations (3) & (4) which results in a cluster (3, 4) with height 0.45.
Dissimilarity matrix :

	0.8
0.8	

i = 4 : Fusing clusters (1, 2) and (3, 4) to form cluster ((1,2), (3,4)) at height 0.8

```
a = as.dist(matrix(c(0, 0.3, 0.4, 0.7,
                     0.3, 0, 0.5, 0.8,
                     0.4, 0.5, 0.0, 0.45,
                     0.7, 0.8, 0.45, 0.0), nrow=4))
plot(hclust(a, method = "complete"))
```

Cluster Dendrogram



(b). Using Simple linkage clustering

Step 1: We already have:

	0.3	0.4	0.7
0.3		0.5	0.8
0.4	0.5		0.45
0.7	0.8	0.45	

Step 2: i=4 : 0.3 is the minimum dissimilarity, thus we fuse observations '1' & '2' to form cluster (1,2) at hight 0.3

Dissimilarity Matrix:

0.4		0.7
0.4		0.45
0.7	0.45	

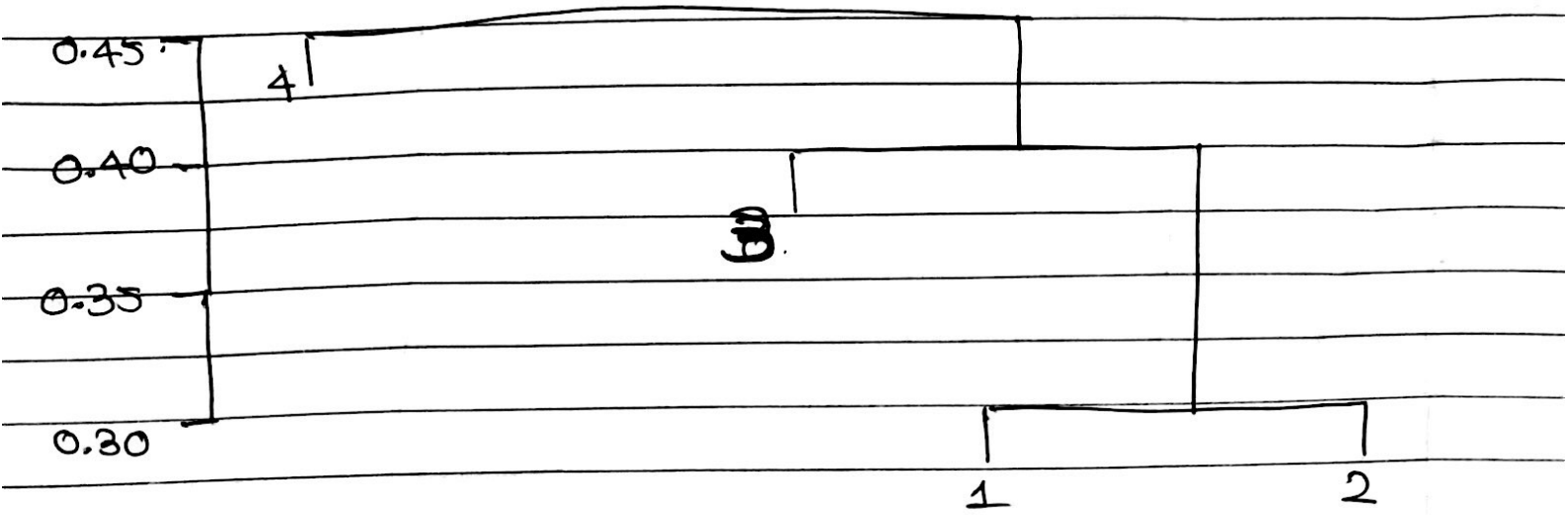
i= 3 : 0.4 is the minimum dissimilarity, so we fuse cluster (1,2) and observation 3 to form cluster 8 to form cluster ((1,2),3) at height 0.4

Dissimilarity Matrix:

0.45
0.45

i=4 : It remains to fuse cluster ((1,2),3) and observation 4 to form cluster (((1,2),3),4) at height 0.45

Cluster · Dendrogram



c. Suppose we cut the dendrogram obtained in (a) that two clusters result, we have clusters: (1,2) and (3,4)

d. In this case we have: clusters ((1,2),3) and (4)

e. Cluster Dendrogram