DTI 5126: Fundamentals for Applied Data Science

Summer 2021 - Assignment 3

Name: Yomna Jehad Abdelsattar

Part A: Clustering

1) K-means Clustering

a) k = 4.

- Selected only “Sex” and “age”.

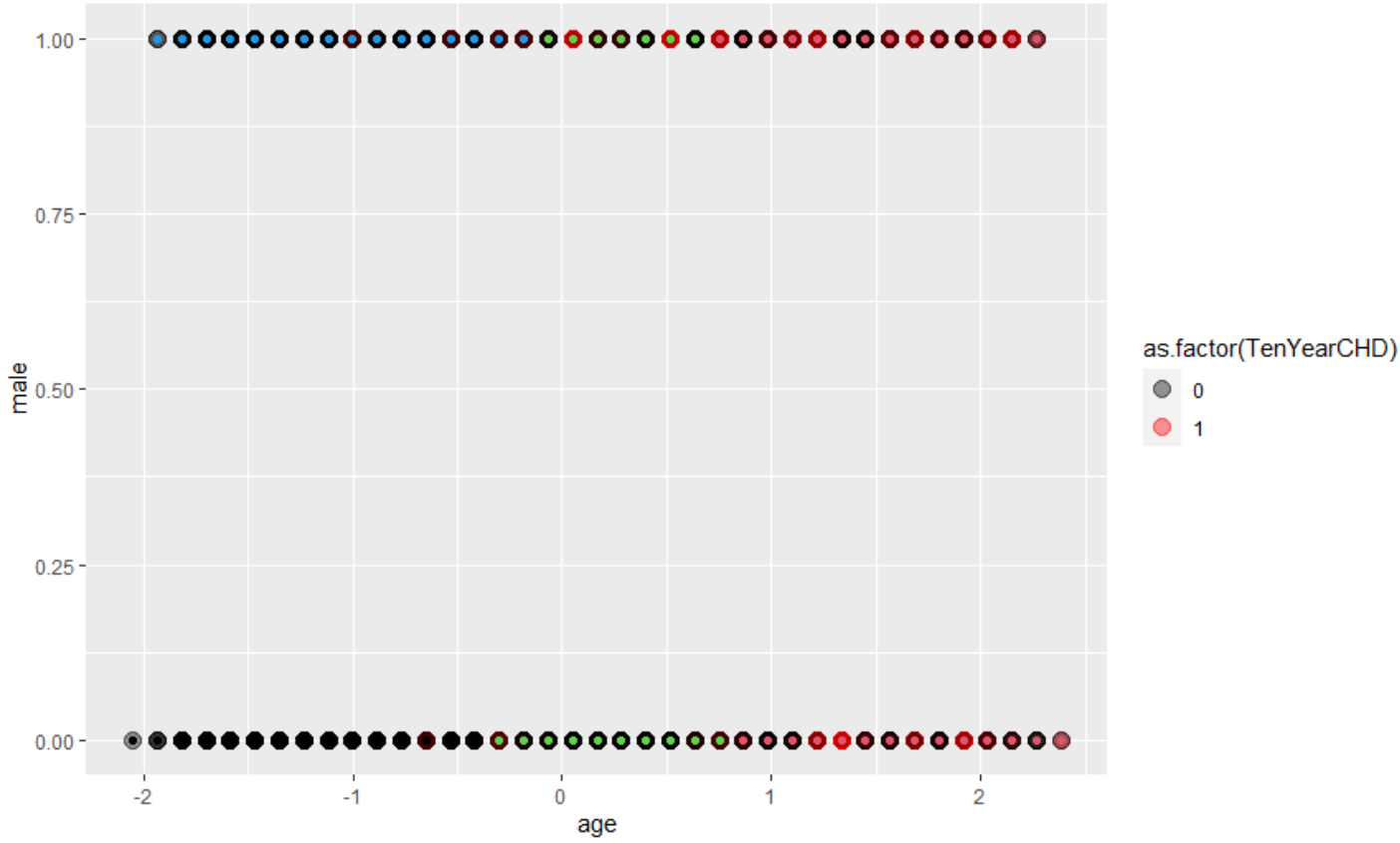
	male	age
1	1	39
2	0	46
3	1	48
4	0	61
5	0	46
6	0	43
7	0	63
8	0	45

- Standardized “age”.

	male	age
1	1	-1.23413741
2	0	-0.41761493
3	1	-0.18432280
4	0	1.33207609
5	0	-0.41761493
6	0	-0.76755314

- Specified number of classes (k) = 4 and applied k-means clustering.

```
> table(Cluster_kmean$cluster,fram$TenYearCHD)

     0    1
1  934   41
2  808  313
3 1029  194
4  825   96
```



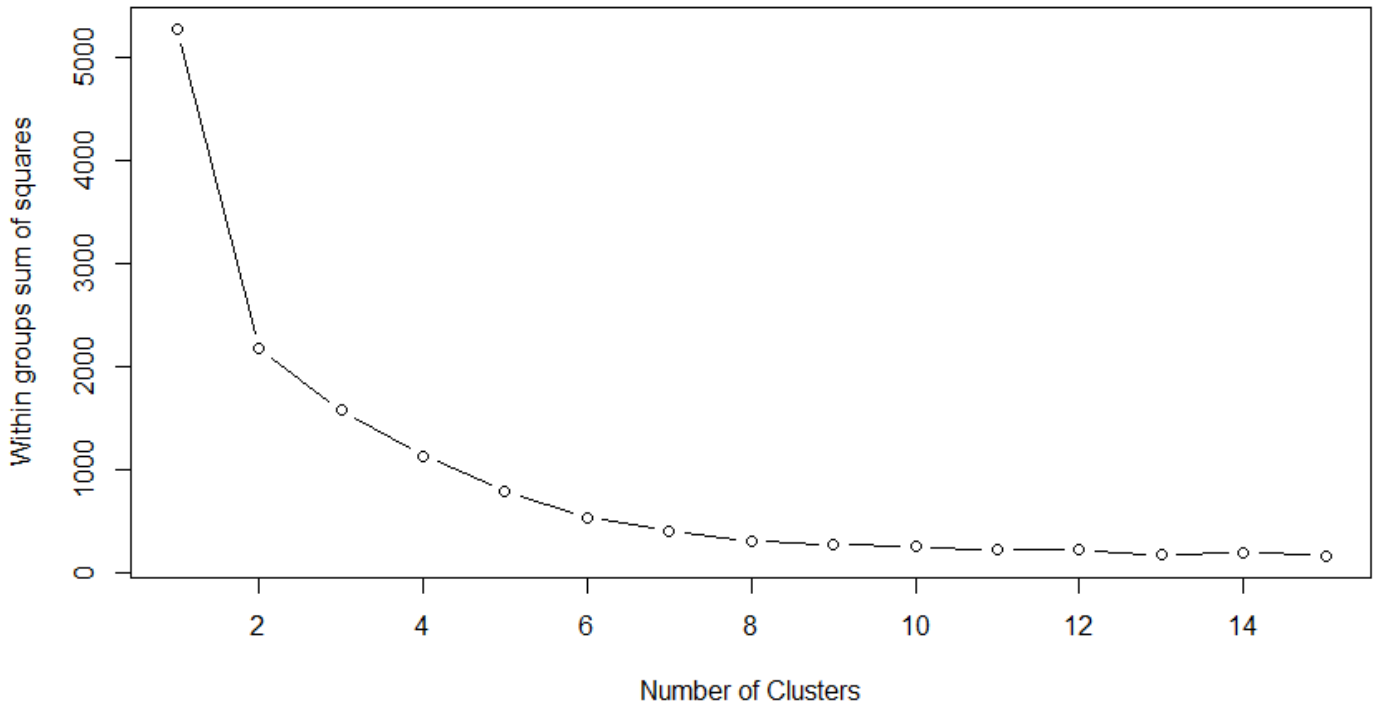
By looking at the plot, we can tell that the algorithm clustered the data in 4 groups:

- Group of young males
- Group of young females
- Group of middle aged people
- Group of old people

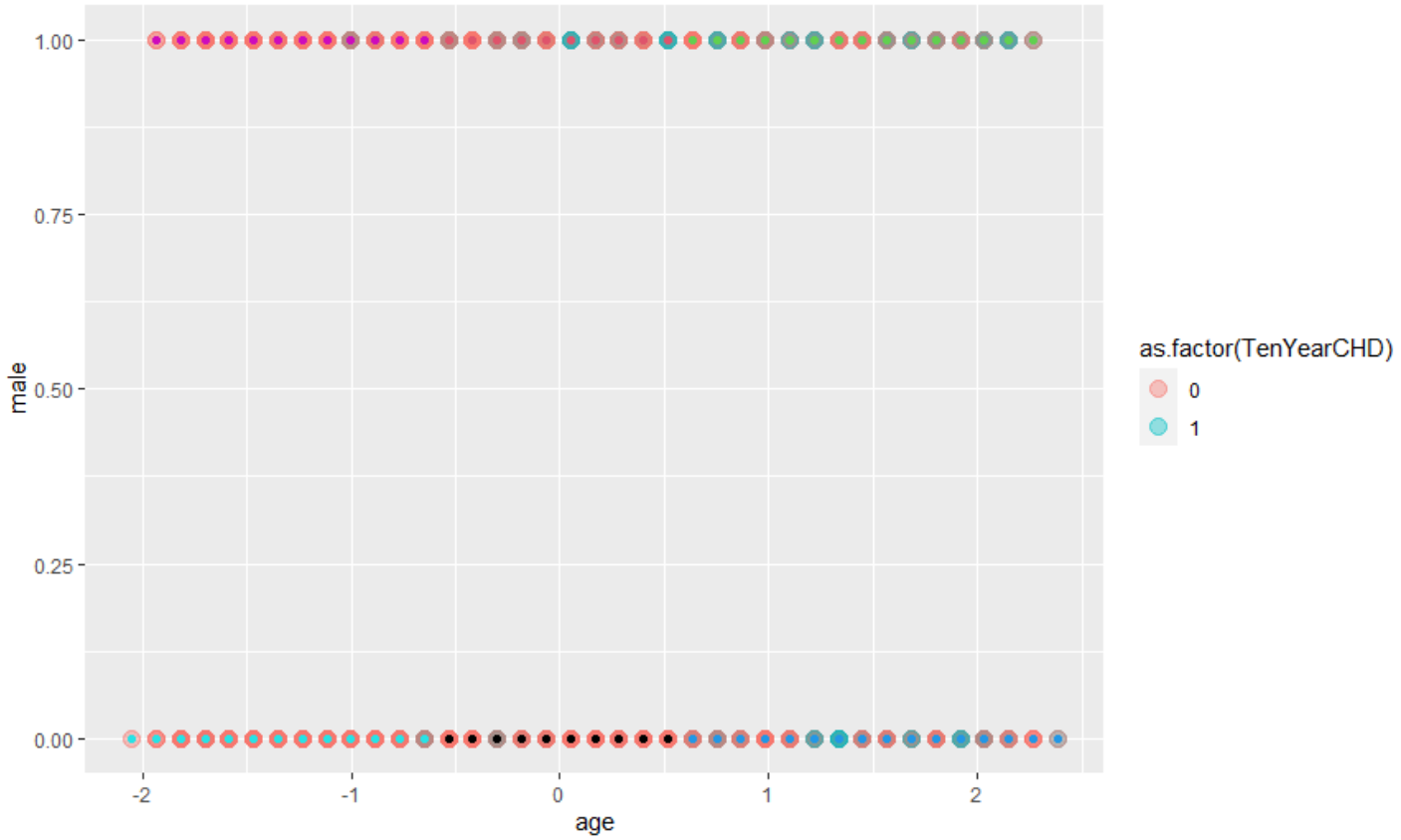
And it also shows that people with TenYearCHD are mainly old people.

b) Apply the elbow method to determine the best k.

- Best k result = 6 clusters (By looking at this graph)



- Apply k-means again.



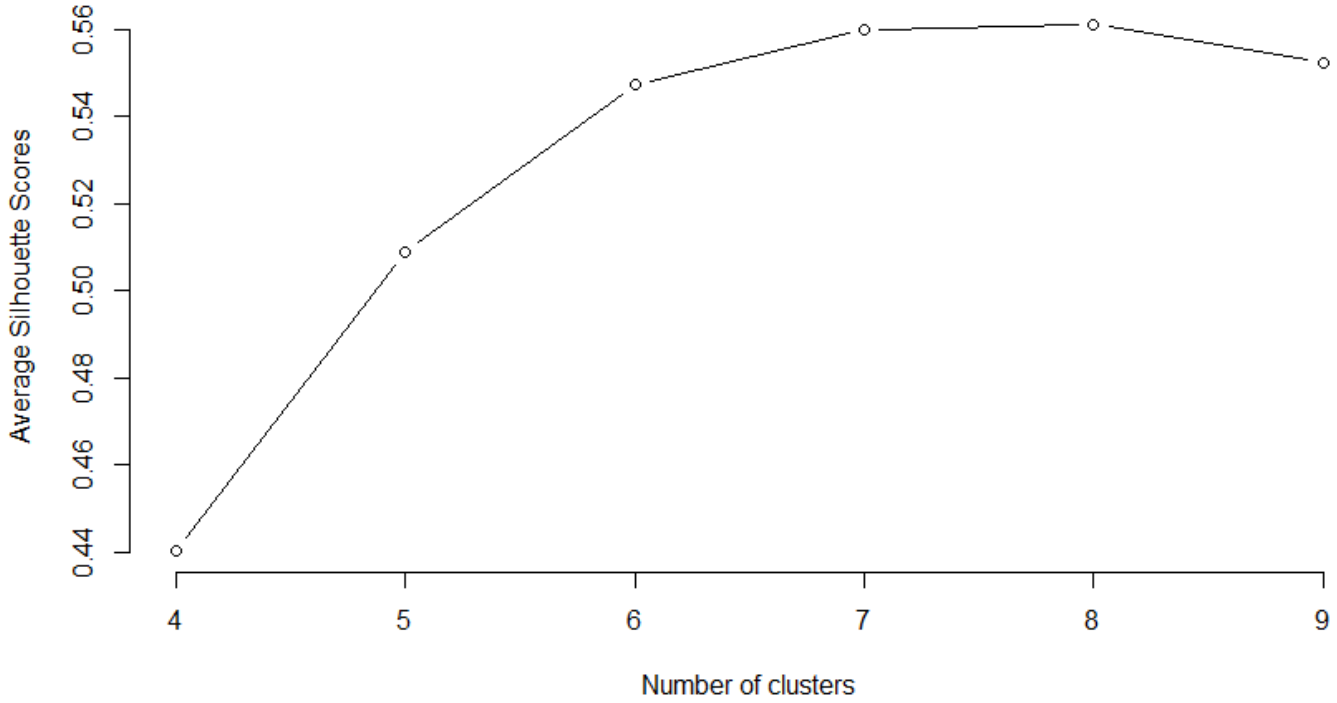
By looking at the plot, to me, this clustering makes sense (6 Groups):

- A group for young males.
- A group for young females.
- A group for middle aged males.
- A group for middle aged females.
- A group for old males.
- A group for old females.

And as previously mentioned, people with TenYearCHD are mainly in the old groups.

c) Evaluate quality of the cluster using the Silhouette Coefficient method.

I tried out many number of clusters to calculate their silhouette score and the plot was as follows:

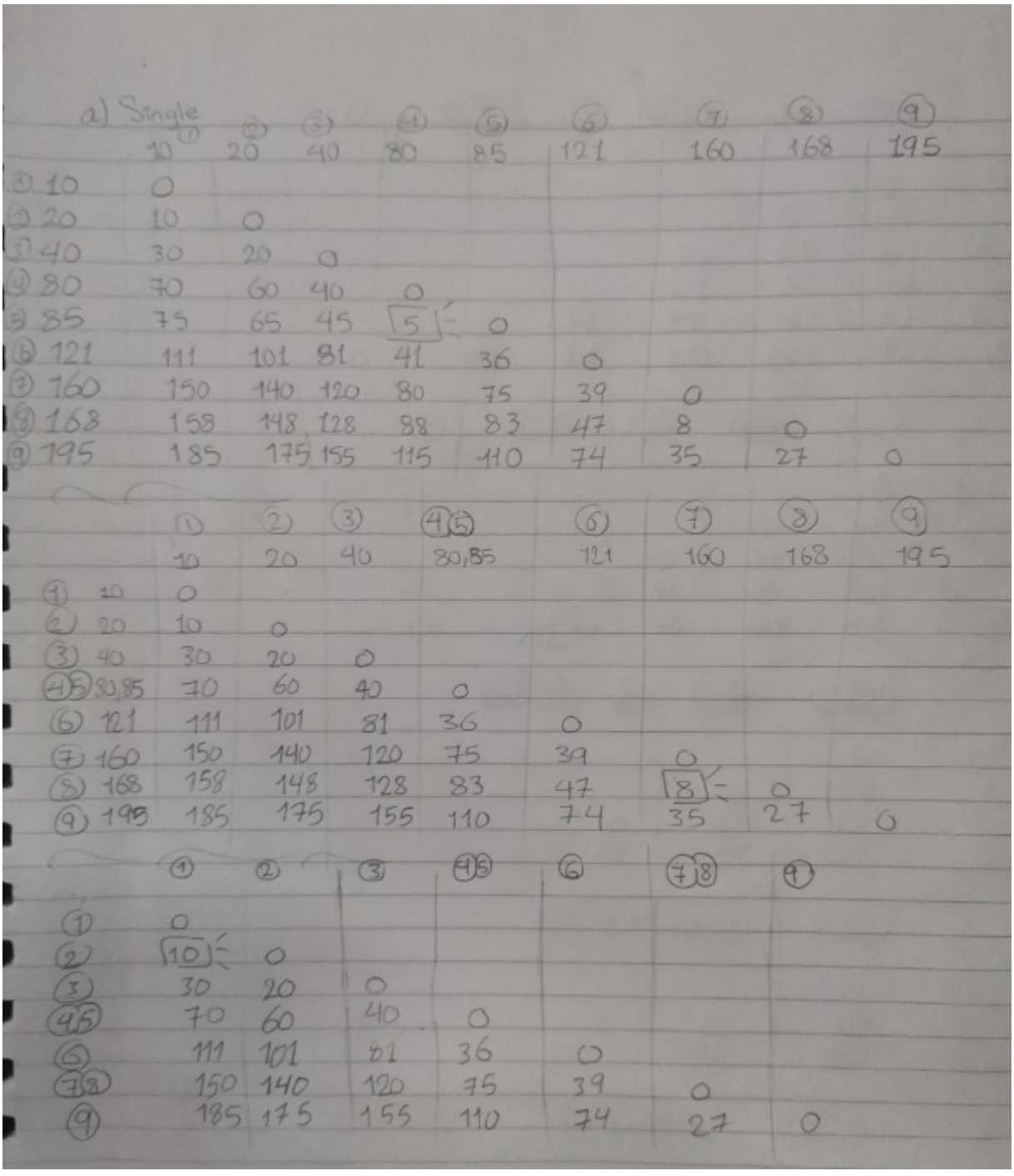


However, by carefully looking at these results, it turns out that the best number of clusters according to the highest silhouette score is either 7 or 8 clusters, and not 6 as I anticipated from the elbow plot.

2) Hierarchical Clustering

a) Single linkage

- Derivation

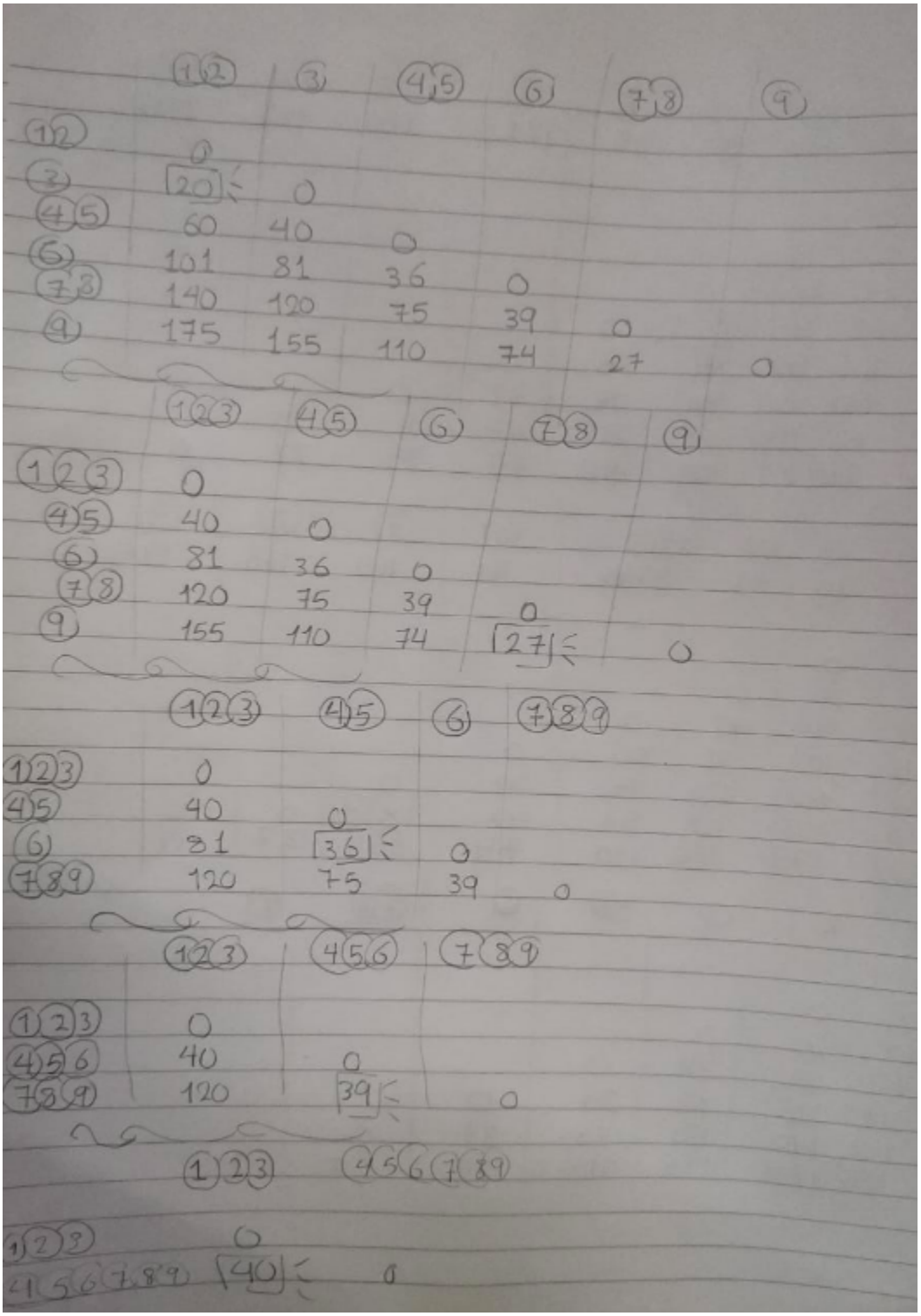


a) Single

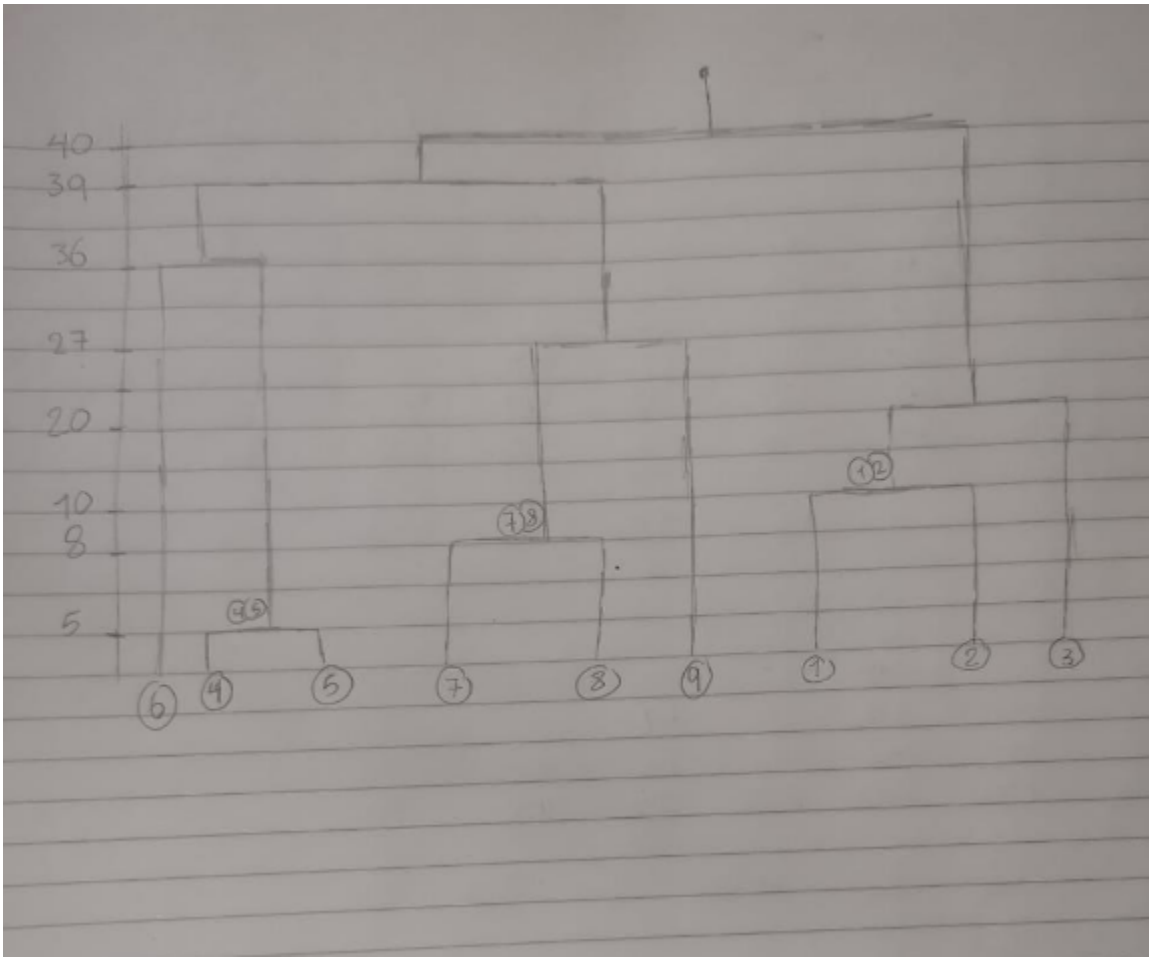
	① 10	② 20	③ 40	④ 80	⑤ 85	⑥ 121	⑦ 160	⑧ 168	⑨ 195
① 10	0								
② 20	10	0							
③ 40	30	20	0						
④ 80	70	60	40	0					
⑤ 85	75	65	45	[5] :=	0				
⑥ 121	111	101	81	41	36	0			
⑦ 160	150	140	120	80	75	39	0		
⑧ 168	158	148	128	88	83	47	8	0	
⑨ 195	185	175	155	115	110	74	35	27	0

	① 10	② 20	③ 40	④⑤ 80,85	⑥ 121	⑦ 160	⑧ 168	⑨ 195
① 10	0							
② 20	10	0						
③ 40	30	20	0					
④⑤ 80,85	70	60	40	0				
⑥ 121	111	101	81	36	0			
⑦ 160	150	140	120	75	39	0		
⑧ 168	158	148	128	83	47	[8] :=	0	
⑨ 195	185	175	155	110	74	35	27	0

	①	②	③	④⑤	⑥	⑦⑧	⑨
①	0						
②	[10] :=	0					
③	30	20	0				
④⑤	70	60	40	0			
⑥	111	101	81	36	0		
⑦⑧	150	140	120	75	39	0	
⑨	185	175	155	110	74	27	0

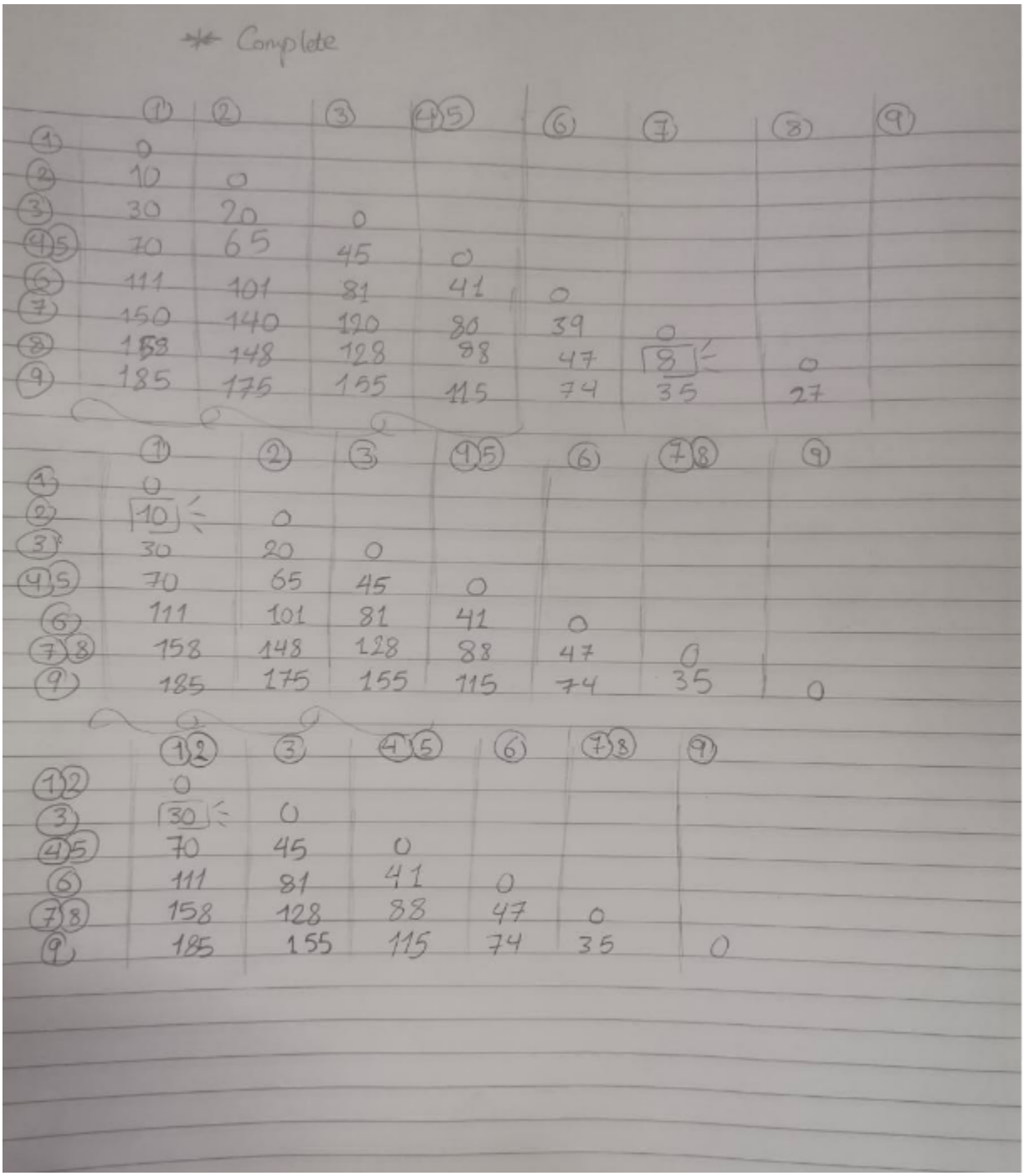

1 2 | 3 | 4 5 | 6 | 7 8 | 9
1 2
2 | 20 | 0
4 5 | 60 | 40 | 0
6 | 101 | 81 | 36 | 0
7 8 | 140 | 120 | 75 | 39 | 0
9 | 175 | 155 | 110 | 74 | 27 | 0
1 2 3 | 4 5 | 6 | 7 8 | 9
1 2 3 | 0
4 5 | 40 | 0
6 | 81 | 36 | 0
7 8 | 120 | 75 | 39 | 0
9 | 155 | 110 | 74 | 27 | 0
1 2 3 | 4 5 | 6 | 7 8 9
1 2 3 | 0
4 5 | 40 | 0
6 | 81 | 36 | 0
7 8 9 | 120 | 75 | 39 | 0
1 2 3 | 4 5 6 | 7 8 9
1 2 3 | 0
4 5 6 | 40 | 0
7 8 9 | 120 | 39 | 0
1 2 3 | 4 5 6 7 8 9
1 2 3 | 0
4 5 6 7 8 9 | 40 | 0

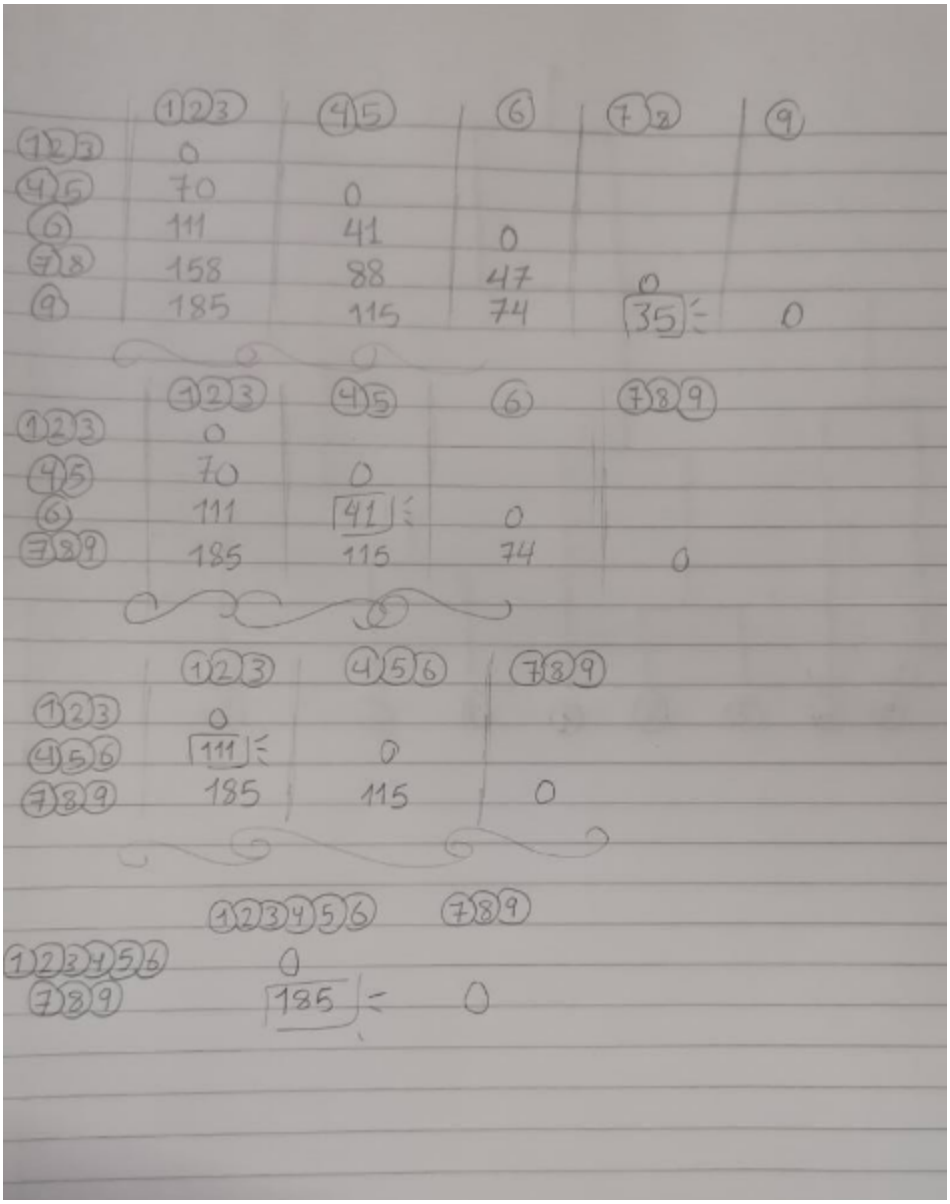
- Dendrogram



b) Complete linkage

- Derivation





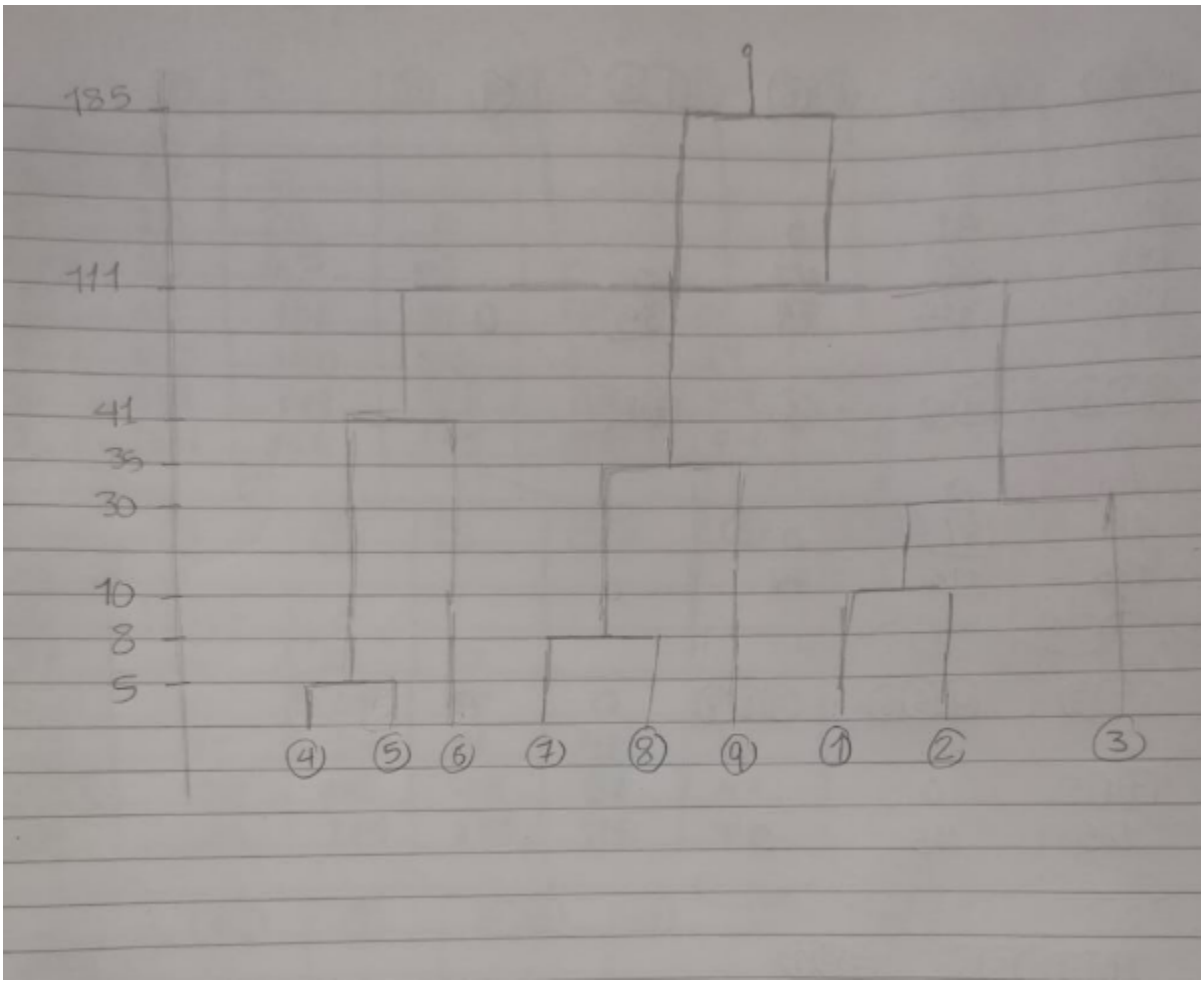
(1)(2)(3)	(4)(5)	(6)	(7)(8)	(9)
(1)(2)(3)	0			
(4)(5)	70	0		
(6)	111	41	0	
(7)(8)	158	88	47	0
(9)	185	115	74	35 = 0

(1)(2)(3)	(4)(5)	(6)	(7)(8)(9)
(1)(2)(3)	0		
(4)(5)	70	0	
(6)	111	41 =	0
(7)(8)(9)	185	115	74 0

(1)(2)(3)	(4)(5)(6)	(7)(8)(9)
(1)(2)(3)	0	
(4)(5)(6)	111 =	0
(7)(8)(9)	185	115 0

(1)(2)(3)(4)(5)(6)	(7)(8)(9)
(1)(2)(3)(4)(5)(6)	0
(7)(8)(9)	185 = 0

- Dendrogram

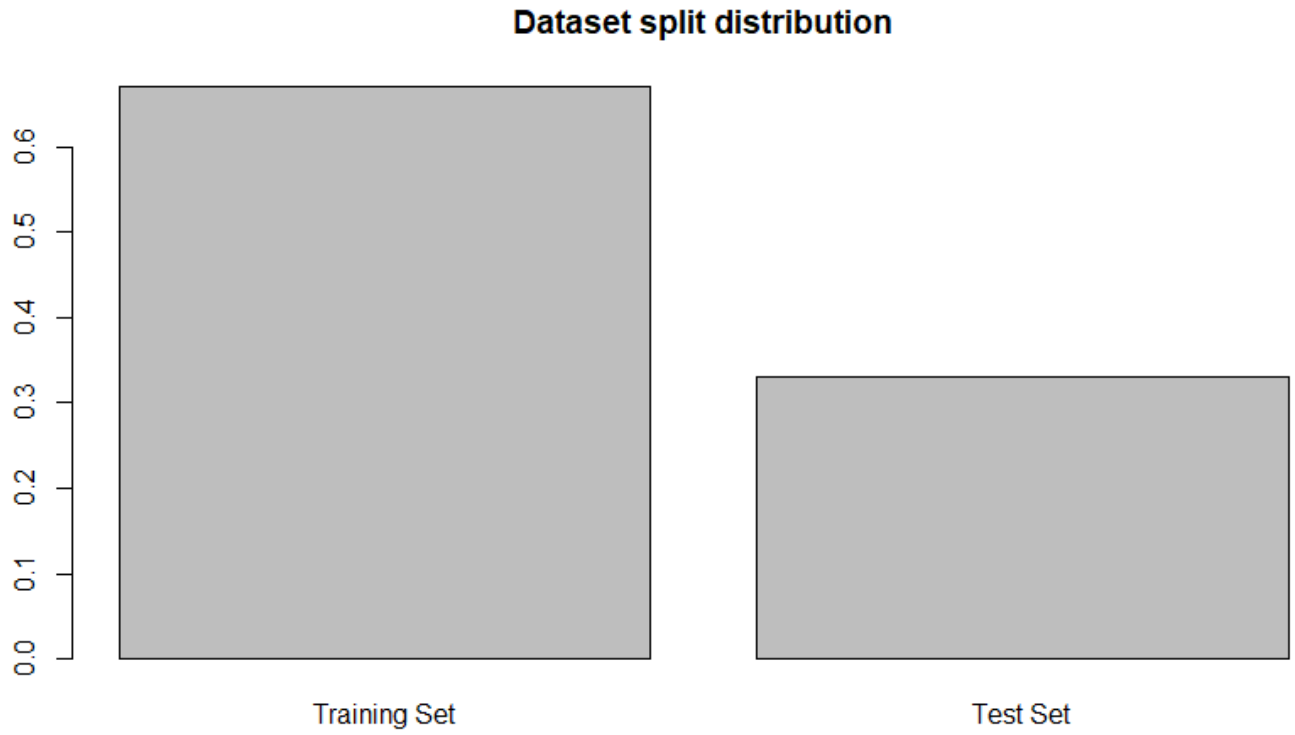


Part B: Model Evaluation & Performance Improvement

a) Partition data 67% training set and 33% test set.

test_set	2324 obs. of 21 variables
train_set	4719 obs. of 21 variables

- Bar graph with the results.



As shown in the bar graph, the ratio between training set and test set as requested.

b) Rebalance data.

- Identify total number of training set records.

```
> nrow(train_set)
[1] 4719
>
```

- Identify the number of “YES” Churn values in the training set.

old_yes_train_set_no	Named int 1252

- In order to have a “YES” Churn percentage of 30% in the training dataset, calculate the number of “YES” Churn records that need to be resampled.

```
# ----------------------------

old_yes_train_set_no = table(train_set$Churn)["Yes"]

old_yes_train_set_ratio = old_yes_train_set_no / nrow(train_set)
no_to_yes_resample_ratio = 0.3 - old_yes_train_set_ratio
no_to_yes_resample_no = no_to_yes_resample_ratio * nrow(train_set)
new_yes_train_set_no =  nrow(train_set) - table(train_set$Churn)["No"] + no_to_yes_resample_no
new_yes_train_set_ratio = new_yes_train_set_no/ nrow(train_set)
```

As a result of these calculations, the number of “Yes” Churn records that need to be resampled = 164

new_yes_train_set_no	Named num 1416
new_yes_train_set_rat…	Named num 0.3
no_to_yes_resample_no	Named num 164
no_to_yes_resample_ra…	Named num 0.0347
old_yes_train_set_no	Named int 1252
old_yes_train_set_rat…	Named num 0.265

c) Resample and confirm.

After resampling using the ROSE package,

```
> table(data.balanced.under$Churn)

  No  Yes
3461 1473
```

To further confirm the desired ratios:

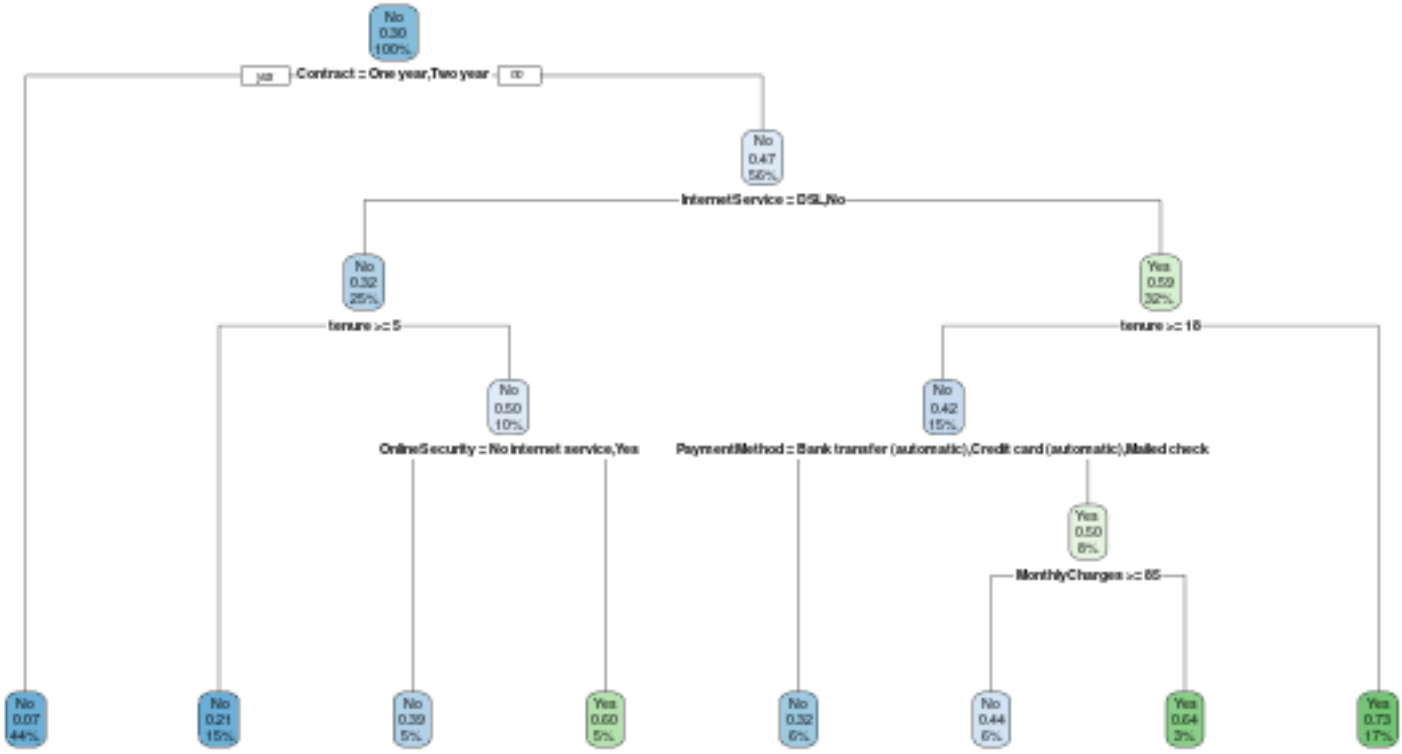
```
new_new_yes_train_set_no = table(data.balanced.under$Churn)["Yes"]

new_new_yes_train_set_ratio = new_new_yes_train_set_no / nrow(data.balanced.under)
```

new_new_yes_train_set_no	Named int 1473
new_new_yes_train_set_ratio	Named num 0.299

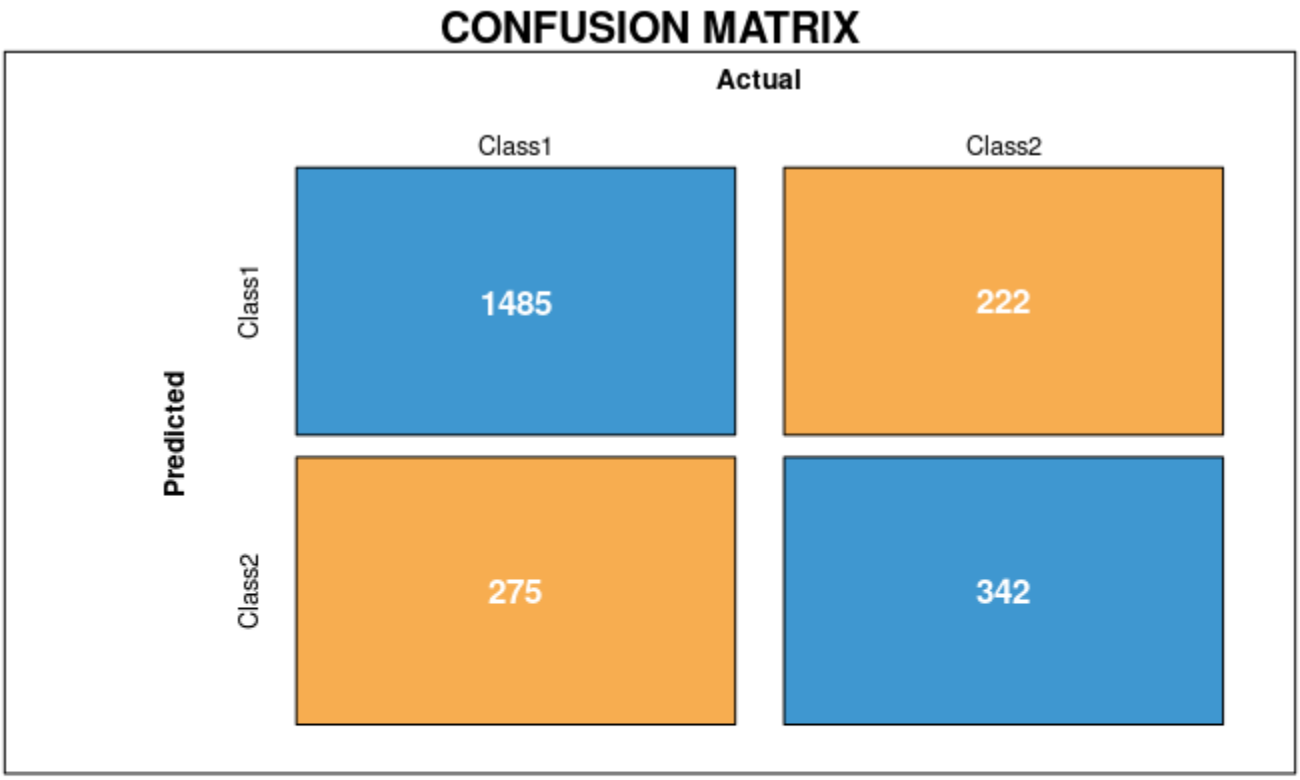
New percentage is 0.299 (approximately 30%).

d) Decision tree

- Tree plot



- Confusion matrix



DETAILS

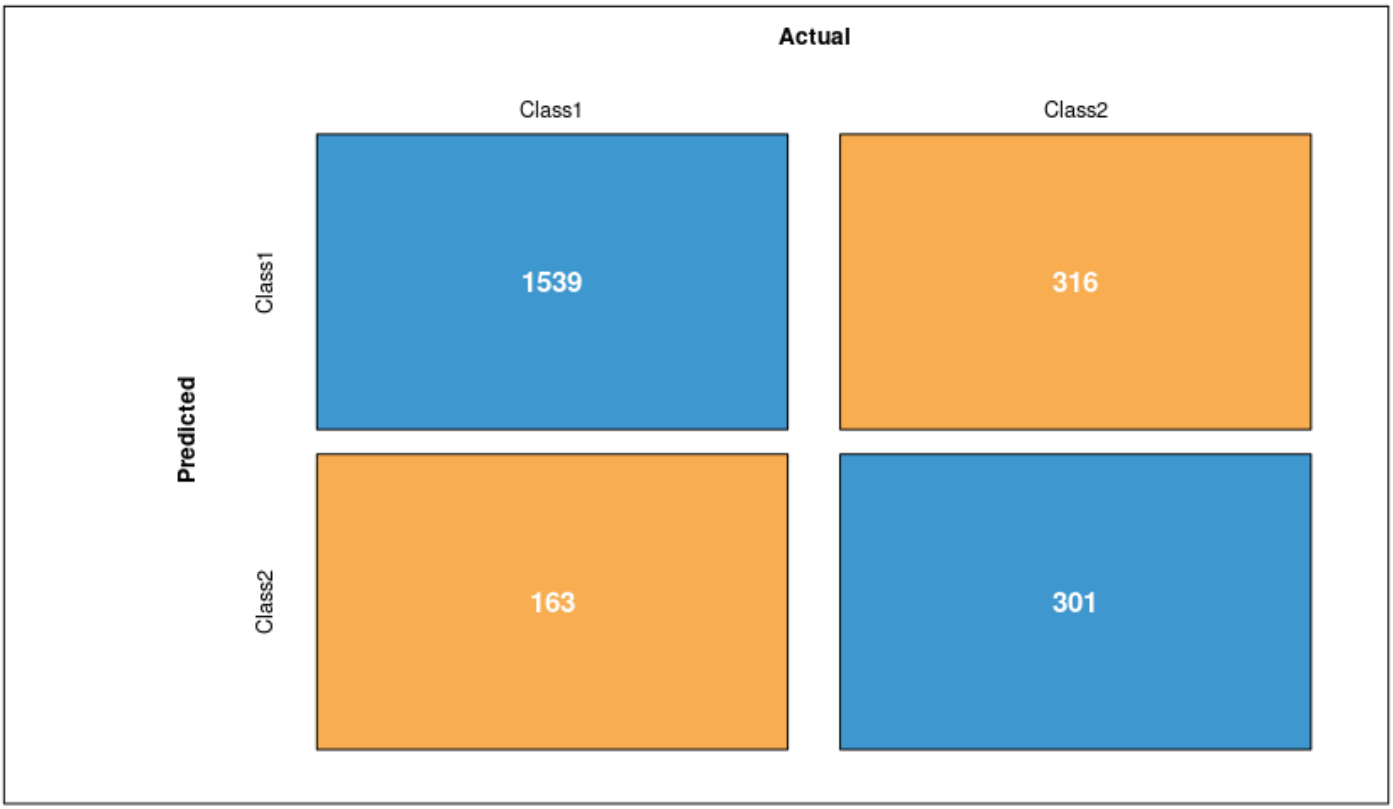
Sensitivity	Specificity	Precision	Recall	F1
0.844	0.606	0.87	0.844	0.857

Accuracy	Kappa
0.786	0.436

e) Ensemble method: Random forest.

- Initial model with default hyperparameters

CONFUSION MATRIX



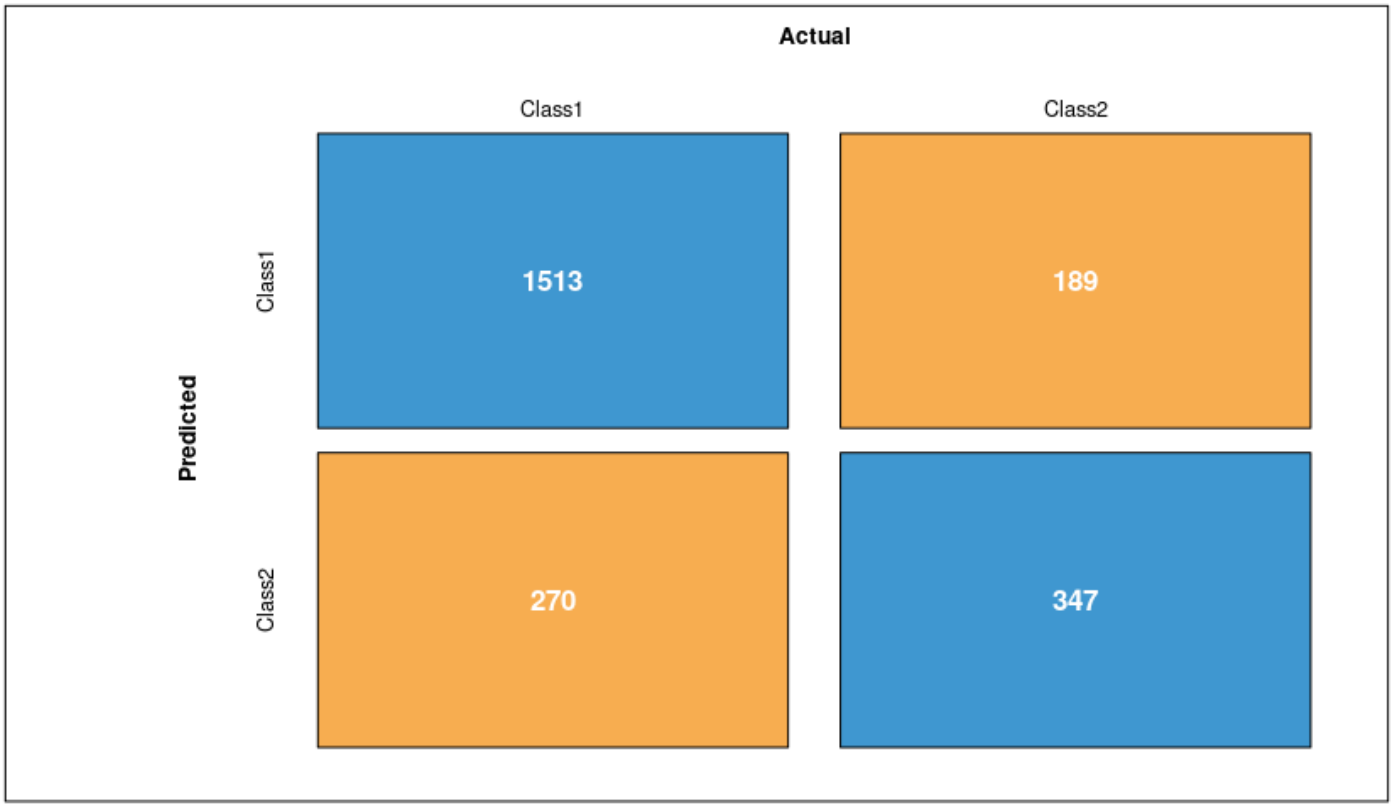
DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.904	0.488	0.83	0.904	0.865

Accuracy	Kappa
0.793	0.426

- Hyperparameters tuning (increased ntree = 100)

CONFUSION MATRIX



DETAILS

Sensitivity	Specificity	Precision	Recall	F1
0.849	0.647	0.889	0.849	0.868

Accuracy	Kappa
0.802	0.471

- Model comparison
  By simple tuning to the hyperparameter the accuracy, we managed to increase the overall accuracy and specificity. However we notice that sensitivity (recall) has decreased. This is the effect of the ntree hyperparameter I chose to tune.

I tried to tune the model at ntree= 20 and ntree = 200 as well, the accuracy turned out to be 0.791 and 0.797 respectively, the other metricies did not vary significantly either. So i concluded that ntree = 100 is quite a good place to avoid overfitting as it's known that generally increasing the number of trees or depth of a tree makes it prone to overfitting.

f) Ensemble method vs Decision tree: Confusion matrix

- Accuracy, Sensitivity and Specificity.
  By looking at the confusion matrix provided after each model,
  We can conclude that the ensemble model (Random Forest) got better overall accuracy and F1 Scores than the normal decision tree model.
  However when it comes to sensitivity and specificity it significantly got better which is a good sign for model generalization ability.

- Best model

  To choose the best model, we need to have a criteria. For the Churn prediction problem. I believe that the company would like to accurately predict the number of customers who will actually churn (True positives) and decrease the falsely predicted as positive churn customers (False Positive) because the company will most probably send marketing items to the churn potential clients which will cost them money.
  That being said; I think the most important metric here is the one that minimizes the False Positive: Precision.
  By looking at the confusion matrices:
  Decision Tree: 0.87
  Initial Random Forest: 0.83
  Tuned Random Forest: 0.889

  Thus, the best model is the Tuned Random Forest.

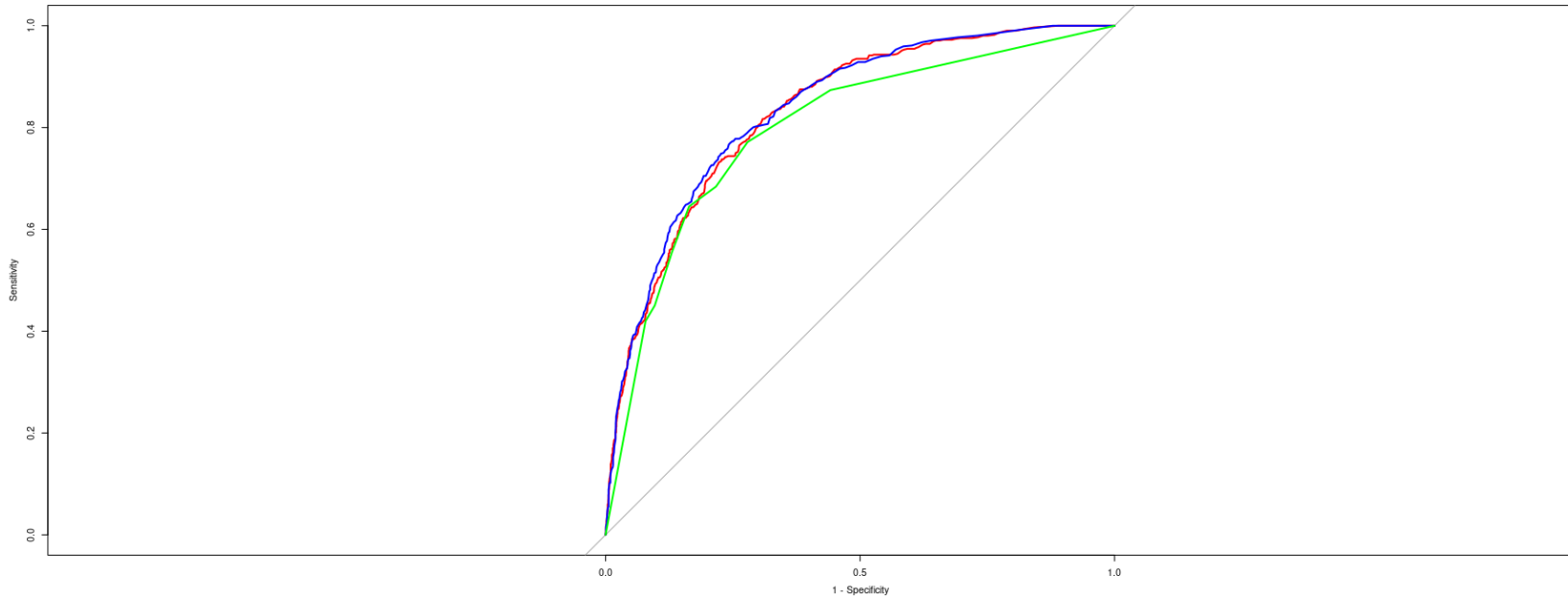
- Worst model is : Initial random forest.

g) Ensemble method vs Decision tree: ROC analysis

- Decision Tree: GREEN

- Ensemble without hyperparameter tuning: RED

- Ensemble with hyperparameter tuning: BLUE



The best model will be the model with the biggest area under the curve (AUC). By looking at the overlapped plot, the Blue one has the largest area, thus, the ensemble with hyperparameter tuning is the best model.