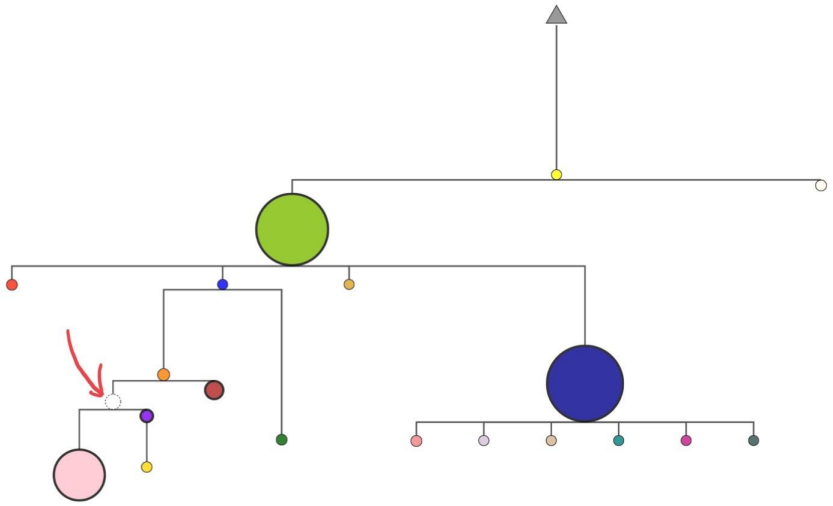
Now, our function **print_height_depth (file_fasta, tree, outputfile)** writes a warning in the metrics file that there was an empty node on the path when the height or depth parameters were calculated.

For example, for **dataset9_1_simplifie** when we have empty node on the path to calculate depth and height:

	H1	H2	H3	D1	D2	D3
dataset9_1_simplifie	1	5	0	3	2	6



dataset9_1_simplifie Elbow tree

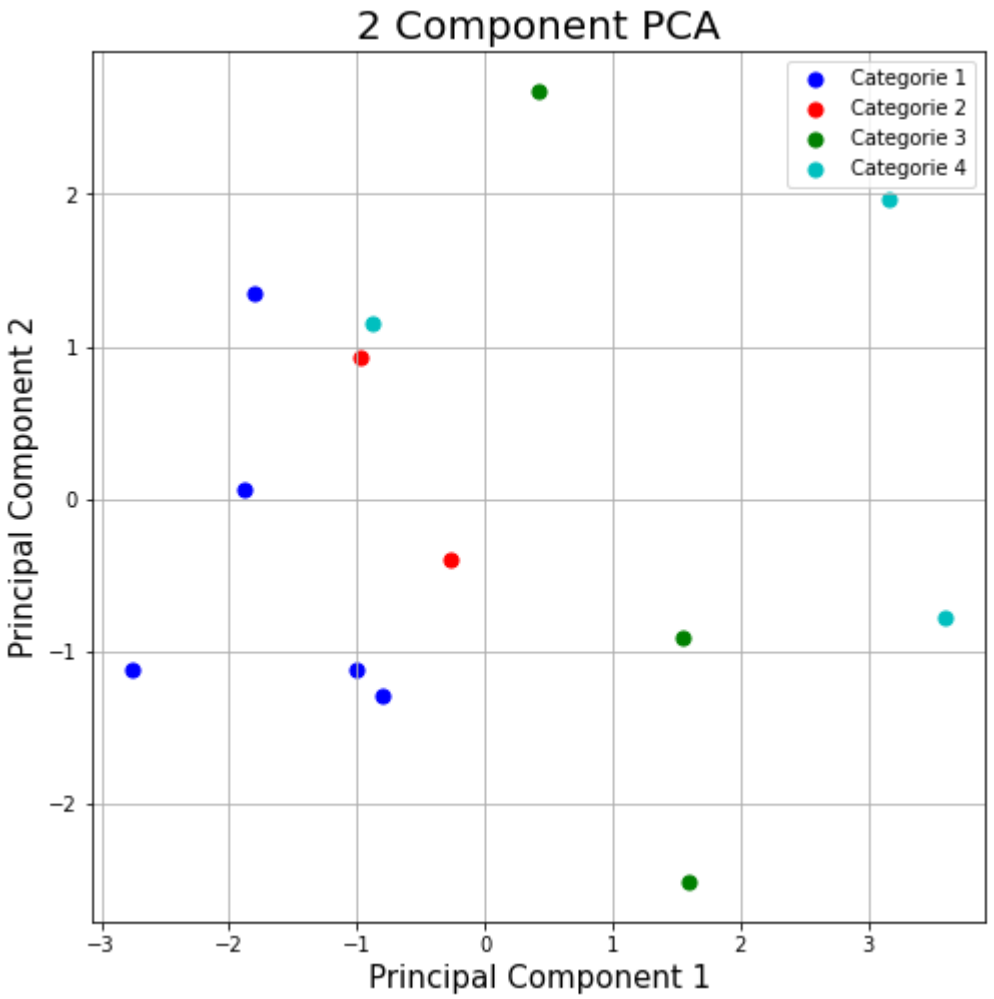
We will have these warnings in the Metrics file

dataset9_1_simplifieMetricPD.txt

```
Number of branches : 19
Sum of all branches length (PD) : 74.0
Sum of all branch length divided by the number of clonotypes of the lineage (avPD) : 3.89

#=====================================================#

Height

The most abundant clonotype - seq29 : height = 1
The second most abundant clonotype - seq2 : height = 5
The third most abundant clonotype - seq83 : height = 0

The height parameter for seq2 was calculated using a node that wasn't in the original sequence file.

#=====================================================#

Depth

The most abundant clonotype - seq29 : depth = 3
The second most abundant clonotype - seq2 : depth = 2
The third most abundant clonotype - seq83 : depth = 6

The depth parameter for seq83 was calculated using a node that wasn't in the original sequence file.
```

PCA for our data

We use all our parameters for calculating two principal components - ‘Number of branches', 'PD - sum of length of all branches', 'avPD - average sum of length of all branches', 'Height 1', 'Height 2', 'Height 3', 'Depth 1', 'Depth 2', 'Depth 3'.



Then we calculate **explained_variance_ratio_** for our components (percentage of variance explained by each of the selected components).

pca.explained_variance_ratio_ = [0.39320932, 0.23249094]

If a sum of **explained_variance_ratio_** for 2 components is more than 0.85, these components use enough data. As we have limited data, we have sum = 0.62570026.

KNN

En utilisant PCA et les paramètres ['Number of branches','Average of Branches', 'H1', 'H2', 'H3', 'D1', 'D2','D3'], on a obtenu les composants principaux pour chaque arbre. On a gardé les deux premières pour chacun et on les a gardé avec le titre PC1 et PC2 dans la Table 2.

Avec les résultats de PCA on a utilisé KNN algorithme pour essayer de classifier les arbres selon les caractéristiques choisies. On a utilisé deux approches pour l'initialisation de notre algorithme: Dans le premier cas, on a créé une matrice en zéros. Dans le deuxième cas, on a utilisé la moyenne des PC1 et PC2 de chaque classe pour initialiser l'algorithme.

Après l'initialisation, on a modifié les centres des classes en suivant l'algorithme KNN, et la modification avec alpha=0.05. On a arrêté l'apprentissage quand la distance entre la matrice antérieure des centres et la matrice nouvelle des centres était inférieure à 0.001.
On a ajouté le résultat des centres initialisées avec zero dans la Table 2 avec le titre 'KNN_zero' et celui avec les moyennes, avec le titre 'KNN_moy' dans la Table 2.

On a calculé la sensitivité et spécificité pour KNN_zero (Table 3) et KNN_moyenne (Table 4)

	PC1	PC2	Categorie	KNN_zero	KNN_moy
0	1.663964	1.345121	1	1.0	1.0
1	1.687469	0.047713	1	1.0	1.0
2	1.087055	-1.146615	1	1.0	1.0
3	1.013657	-1.320312	1	1.0	1.0
4	3.158271	-1.201718	1	1.0	1.0
5	-0.184434	-0.365643	2	4.0	2.0
6	0.526174	0.953448	2	3.0	2.0
7	-1.009050	2.730705	3	3.0	4.0
8	-1.845943	-2.473893	3	2.0	3.0
9	-2.193221	-0.839032	3	2.0	3.0
10	-1.764859	1.899486	4	3.0	4.0
11	0.779805	1.155897	4	3.0	2.0
12	-2.918889	-0.785157	4	2.0	3.0

Table 2. PCA et classification

	True P	True N	False P	False N	Sensitivity	Specificity
1	5.0	8.0	0.0	0.0	1.000000	1.000000
2	2.0	10.0	1.0	0.0	1.000000	0.909091
3	2.0	9.0	1.0	1.0	0.666667	0.900000
4	1.0	9.0	1.0	2.0	0.333333	0.900000

Table 3. Erreur pour KNN_zero

	True P	True N	False P	False N	Sensitivity	Specificity
1	5.0	8.0	0.0	0.0	1.000000	1.000000
2	2.0	10.0	1.0	0.0	1.000000	0.909091
3	2.0	9.0	1.0	1.0	0.666667	0.900000
4	1.0	9.0	1.0	2.0	0.333333	0.900000

Table 3. Erreur pour KNN_moyenne