Approche de sélection de la modularité locale pour le calcul de communauté égo-centré

Ghassen MARRAKCHI Abdelmounaim BOUZERIRA

Mars, 2023

1 Introduction

Dans le présent projet, on s'intéresse à la relation entre les mesures de centralités et celles des modularités.

En effet, la qualité d'une communauté égo-centré sur un noeud dépend fortement du choix de la fonction qualité c.-à-d la modularité locale à optimiser. Il est donc probable que les mesures de centralités d'un noeud donnent l'information sur la mesure de modularité qui lui est la plus adéquate. Ainsi, dans ce projet, on cherche à proposer un système d'aide à la sélection de la modularité locale à utiliser en fonction des caractéristiques topologiques du noeud cible.

Pour ce faire, nous allons commencer par construire le jeu de données c.a.d transformer les graphes en données tabulaires (2). Ensuite, on construit des modèles d'apprentissage automatique pour la classification multi-classe (les classes étant les modularités locales)(3). Finalement, on exposera les résultats obtenus (4).

2 Jeux de données

Le processus de transformation des graphes en données tabulaires commence par du feature engineering sur chacun des noeuds des graphes utilisés (extraction des caractéristiques topologiques). Ensuite, les noeuds (les enregistrements) sont étiquetés suivant le calcul de modularité.

2.1 Graphes

On s'est basé sur 4 graphes :

— **Dolphins** : Un réseau social de grands dauphins. L'ensemble de données contient une liste de tous les liens, où un lien représente des associations fréquentes entre les dauphins.

— **Karate** : représente les relations d'amitié entre les 34 membres d'un club universitaire de karaté.
— **Football** : représente le réseau des matchs de football américain entre les collèges de la division IA. Les nœuds ont des valeurs qui indiquent à quelles conférences auxquelles ils appartiennent
— **Polbooks** : représente un réseau de livres sur la politique américaine publiés à l'époque de l'élection présidentielle de 2004. Les arêtes entre les livres représentent des co-achats fréquents de livres par les mêmes acheteurs.

2.2 Feature Engineering

Comme précisé précédemment, la transformation repose sur le calcul des caractéristiques topologiques des noeuds. On donc choisi ce qui est disponible dans la bibliothèque **igraph** :

— **Degré** (degree) : Le degré d'un nœud dans un graphe de réseau social est le nombre de liens qui y sont attachés.
— **Transitivité** (transitivity) : La transitivité mesure la probabilité que deux nœuds qui sont connectés à un troisième nœud soient également connectés l'un à l'autre.
— **Intermédiarité** (betweenness) : L'intermédiarité mesure le nombre de fois qu'un nœud est sur le chemin le plus court entre deux autres nœuds. Les nœuds avec une intermédiarité élevée sont considérés comme importants pour maintenir la connectivité du graphe.
— **Proximité** (closeness) : La proximité mesure la distance moyenne d'un nœud à tous les autres nœuds dans le graphe. Les nœuds avec une proximité élevée sont considérés comme plus centraux dans le graphe.
— **Centralité** (coreness) : La centralité mesure à quel point un nœud est important pour la structure en étoile du graphe. Les nœuds avec une centralité élevée sont considérés comme faisant partie du noyau central du graphe.
— **Vecteur propre** (eigenvector) : La centralité de vecteur propre mesure la capacité d'un nœud à se connecter à des nœuds importants dans le graphe. Les nœuds avec une centralité de vecteur propre élevée sont considérés comme importants pour la structure générale du graphe. Il s'agit aussi d'un cas particulier de la mesure **Katz**.
— **PageRank** : Le PageRank est une mesure utilisée par le moteur de recherche Google pour évaluer l'importance des pages web. Il utilise une version modifiée de la centralité de vecteur propre pour attribuer des scores de pertinence aux pages web en fonction de leur nombre de liens entrants et de la qualité de ces liens.
— **Harmonique** (harmonic) : La centralité harmonique mesure la capacité d'un nœud à se connecter à des nœuds éloignés dans le graphe. Les nœuds avec une centralité harmonique élevée sont considérés comme importants pour le maintien de la connectivité du graphe.
— **Contrainte** (constraint) : La contrainte mesure le degré de dépendance d'un nœud par rapport aux autres nœuds dans le graphe. Les nœuds avec une contrainte élevée sont considérés comme étant plus contraints par les autres nœuds du graphe.
— **Autorité** (authority) et Hub : L'autorité et le hub sont des mesures de centralité utilisées dans le cadre de l'algorithme HITS (Hyperlink-Induced Topic Search), qui

est utilisé pour l'analyse des liens hypertextes. L'autorité mesure l'importance d'une page en fonction du nombre de pages qui pointent vers elle, tandis que le hub mesure l'importance d'une page en fonction du nombre

— **Alpha** : La centralité alpha est une mesure de centralité qui combine les avantages de plusieurs autres mesures de centralité. Elle permet de prendre en compte la connectivité directe d'un nœud (mesurée par sa centralité de degré) ainsi que sa connectivité indirecte (mesurée par sa centralité de vecteur propre), en attribuant un poids alpha à chacune de ces deux mesures.

2.3 Étiquetage

L'étiquetage est fait sur la base des modularités des noeuds. On a ainsi trois classes différentes :

1. **Modularité R** : Calcul de toutes les modularité.

$$R = \frac{B_{in}}{B_{in} + B_{out}}$$

avec :
- B_{in} : le nombre d'arêtes qui relient le noeud à d'autres noeuds de sa communauté
- B_{out} : le nombre d'arêtes qui relient le noeud à des noeuds d'autres communautés

2. **Modularité M** : Calcul de toutes les modularité.

$$M = \frac{D_{in}}{D_{out}}$$

avec :
- D_{in} : le nombre d'arêtes reliant deux noeuds appartenant à la même communauté
- D_{out} : le nombre d'arêtes reliant un noeud de la communauté à un noeud extérieur à la communauté (c'est exactement B_{out})

3. **Modularité L** : Calcul de toutes les modularité.

$$L = \frac{L_{in}}{L_{out}}$$

Tel que :

$$L_{in} = \frac{\sum_{i \in D} ||\Gamma(i) \cap D||}{||D||}$$

et

$$L_{out} = \frac{\sum_{i \in B} ||\Gamma(i) \cap S||}{||B||}$$

avec :
- L_{in} : la somme des poids des arêtes reliant deux noeuds appartenant à la même communauté
- L_{out} : la somme des poids des arêtes reliant un noeud de la communauté à un noeud extérieur à la communauté

Concernant le processus d'étiquetage, on effectue pour chaque noeud les étapes suivante :

1. Pour chaque modularité faire :
   (a) calculer la modularité
   (b) comparer à la vérité terrain
   (c) retourner un score
2. prendre la modularité avec le maximum de qualité comme label

Cela étant dit, il faut noter qu'on s'est retrouvé parfois dans des cas d'égalité entre les différentes modularités. Ainsi, pour résoudre ce problème, on a fait recours à un ordre de préférence. Autrement dit, après avois calculé les qualités des modularités et dans le cas d'une égalité entre deux valeurs, on avait choisi suivant un ordre de préférence.

En suivant cette approche, on a dû avoir 6 datasets différentes.

Dataset	Choix 1	Choix 2	Choix 3	Nombre de lignes	Nombre de colonnes
R_M_L	Modularité R	Modularité M	Modularité L	316	13
R_L_M	Modularité R	Modularité L	Modularité M	316	13
M_R_L	Modularité M	Modularité R	Modularité L	316	13
M_L_R	Modularité M	Modularité L	Modularité R	316	13
L_R_M	Modularité L	Modularité R	Modularité M	316	13
L_M_R	Modularité L	Modularité M	Modularité R	316	13

Il faut aussi noter que la distribution des labels a été fortement affecté par cet ordre de préférence (figure 1)

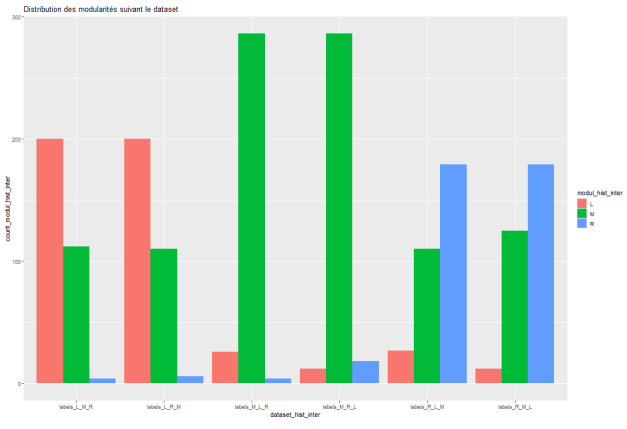

FIGURE 1 – Distribution des labels

3 Apprentissage automatique

L'objectif de ce projet peut être modélisé comme un problème de classification **multi-classes**.

3.1 Modèles

On a choisi trois principaux modèles suivant leurs capacités à traiter ce genre de problématique et leurs facilités d'implémentation :

- **KNN** : qui a été entrainé en faisant varier la valeur de K entre 1 et 5. On remarquera, tout à l'heure, dans les résultats, que la valeur $\mathbf{K = 1}$ a donné les meilleures valeurs mais **dans le sens d'un surentrainement**.
- **Arbre de décision** : qui a été entrainée avec les paramètres de base disponibles.
- **Régression Logistique** (ou régression logistique multimodale) : qui est une adaptation de la régression logistique pour la classification **multi-classes**

Il est à noter que les modèles **KNN** et **Régression Logistique** ont été entrainé avec une approche "**OneVsAll**". En effet, pour parvenir à faire une classification multiple, on reformule le problème en une multitude de problèmes de classifications **binaires**. Ainsi, pour chaque classe, le modèle est entrainé avec la valeur **1** si la classe en question et **0** sinon. La figure ci-dessous explicite l'idée.

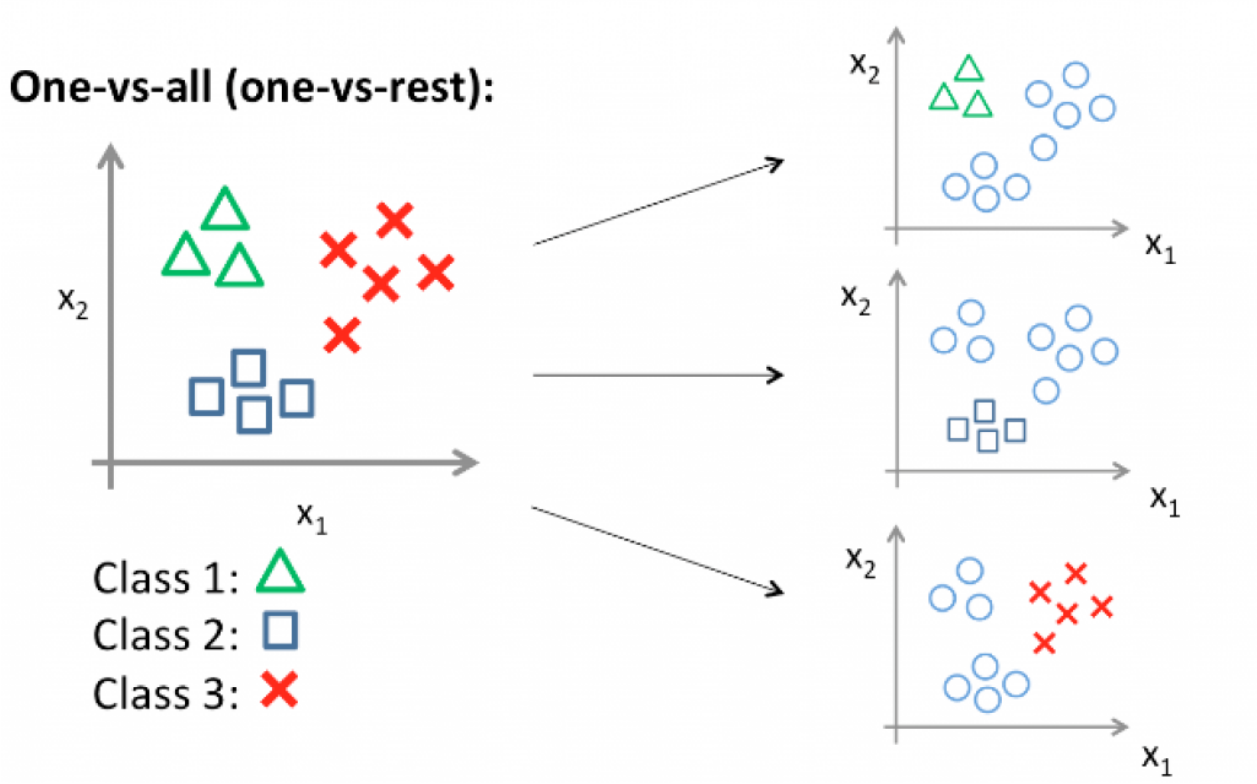

FIGURE 2 – Classification One-vs-All

3.2 Métrique

- **Precision** : La précision est le ratio d'observations positives correctement prédites sur le nombre total d'observations positives prédites.
- **Macro Average Precision** : Calcule la précision pour toutes les classes individuellement, puis les moyenne.
- **Micro Average Precision** : Calcule les vrais positifs et les faux positifs pour chaque classe, puis les utiliser pour calculer la précision globale.
- **Recall** : Le recall est le ratio des observations positives correctement prédites parmi toutes les observations de la classe réelle "oui".
- **Macro Average Recall** : Calcule le rappel pour chaque classe individuellement et ensuite les moyenne.
- **Micro Average Recall** : Calcule les vrais positifs et les faux négatifs pour chaque classe individuellement, puis utiliser cela pour calculer le rappel global.
- **F1** : Le score F1 est la moyenne pondérée de la précision et du rappel.
- **Macro Average F1** : Calcule le score F1 de chaque classe individuellement, puis en faire la moyenne.
- **Micro Average F1** : Calcule le score F1 de chaque classe individuellement et ensuite les moyenne.

Il est à noter que les meilleures métriques pour notre problème seront les métriques moyennes de chaque classe.

4 Analyse des résultats

Le modèle **KNN** a été sur-entrainé pour tous les datasets. Cela dit, on remarque que c'est **l'arbre de décision** qui "remporte" pour la plus part des datasets : **la régression logistique** étant meilleure pour les datasets **M-R-L**, **M-L-R** et sur le niveau global pour le dataset **L-M-R**.

— Jeu de données **R-M-L** : avec le meilleur KNN (**K = 1**)

	Precision	Micro Avg	Macro Avg
Logist	0.7482	0.7233	**0.6484**
KNN	1	1	0.9999
Tree	**0.8741**	**0.8616**	0.5716

	Recall	Micro Avg	Macro Avg
Logist	0.8106	0.7233	0.5927
KNN	1	1	0.9999
Tree	**0.8992**	**0.8616**	**0.6013**

	F1	Micro Avg	Macro Avg
Logist	0.7781	0.7233	**0.6107**
KNN	1	1	0.9999
Tree	**0.8865**	**0.8616**	0.5852

	Accuracy
Logist	0.7233
KNN	1
Tree	**0.8616**

— Jeu de données **R-L-M** : avec le meilleur KNN (**K = 1**)

	Precision	Micro Avg	Macro Avg
Logist	0.8111	0.7075	0.4633
KNN	1	1	0.9999
Tree	**0.8951**	**0.8537**	**0.7731**

	Recall	Micro Avg	Macro Avg
Logist	0.7785	0.7075	0.5090
KNN	1	1	0.9999
Tree	**0.9014**	**0.8537**	**0.7453**

	F1	Micro Avg	Macro Avg
Logist	0.7945	0.7075	0.4847
KNN	1	1	0.9999
Tree	**0.8982**	**0.8537**	**0.7560**

	Accuracy
Logist	0.7075
KNN	1
Tree	**0.8537**

— Jeu de données **M-R-L** : avec le meilleur KNN (**K = 1**)

	Precision	Micro Avg	Macro Avg
Logist	**0.6428**	**0.9367**	**0.8985**
KNN	1	1	0.9999
Tree	0.5	0.9169	0.5220

	Recall	Micro Avg	Macro Avg
Logist	**0.75**	**0.9367**	**0.6099**
KNN	1	1	0.9999
Tree	0.6363	0.9169	0.4941

	F1	Micro Avg	Macro Avg
Logist	**0.6923**	**0.9367**	**0.6638**
KNN	1	1	0.9999
Tree	0.56	0.9169	0.50513

	Accuracy
Logist	**0.9367**
KNN	1
Tree	0.9169

— Jeu de données **M-L-R** : avec le meilleur KNN (**K = 1**)

	Precision	Micro Avg	Macro Avg
Logist	**1**	**0.9407**	**0.9795**
KNN	1	1	0.9999
Tree	0	0.9130	0.4901

	Recall	Micro Avg	Macro Avg
Logist	**1**	**0.9407**	**0.7619**
KNN	1	1	0.9999
Tree	-	0.9130	0.4515

	F1	Micro Avg	Macro Avg
Logist	**1**	**0.9407**	**0.8042**
KNN	1	1	0.9999
Tree	-	0.9130	0.4664

	Accuracy		
Logist	0.9407		
KNN	1		
Tree	**0.9130**		

— Jeu de données **L-R-M** : avec le meilleur KNN (**K = 1**)

	Precision	Micro Avg	Macro Avg
Logist	0	0.7391	0.4787
KNN	1	1	0.9999
Tree	0	**0.8656**	**0.5678**

	Recall	Micro Avg	Macro Avg
Logist	-	0.7391	0.4816
KNN	1	1	0.9999
Tree	-	**0.8656**	**0.5892**

	F1	Micro Avg	Macro Avg
Logist	-	0.7391	0.4796
KNN	1	1	0.9999
Tree	-	**0.8656**	**0.5774**

	Accuracy		
Logist	0.7391		
KNN	1		
Tree	**0.8656**		

— Jeu de données **L-M-R** : avec le meilleur KNN (**K = 1**)

	Precision	Micro Avg	Macro Avg
Logist	**0.6666**	0.7312	**0.6945**
KNN	1	1	0.9999
Tree	0	**0.8814**	0.5804

	Recall	Micro Avg	Macro Avg
Logist	**0.6666**	0.7312	**0.6909**
KNN	1	1	0.9999
Tree	-	**0.8814**	0.5861

	F1	Micro Avg	Macro Avg
Logist	**0.6666**	0.7312	**0.6925**
KNN	1	1	0.9999
Tree	-	**0.8814**	0.5832

	Accuracy		
Logist	0.73122	-	-
KNN	1	-	-
Tree	**0.8814**	-	-

L'atteinte de telles performances reste toujours révélateur d'une certaine qualité des données. En effet, on serait tenté d'expliquer ces résultats par les corrélations existants entre les différentes variables.

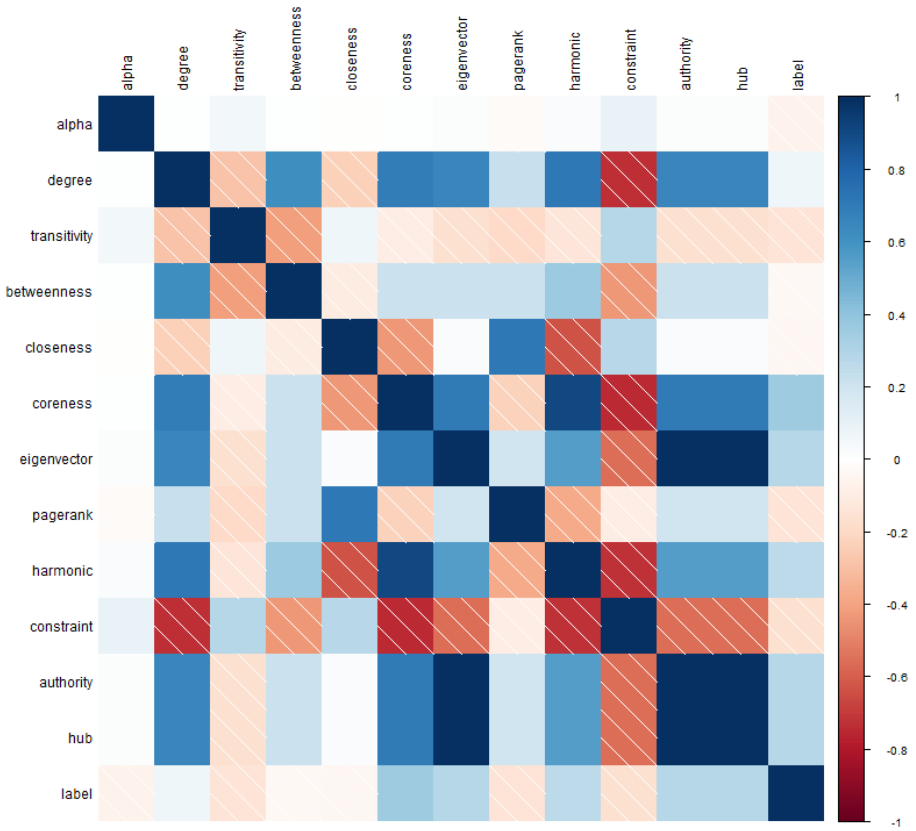

FIGURE 3 – Corrélation des différentes variables

Deux blocs de fortes corrélations positives sont remarquables dans la figure 3. D'abord, entre les variables **authority** et **hub**. Ceci revient à leurs formules respectives. Ensuite, il y a une forte corrélation positive entre ces deux variables et la centralité **en vecteur propre**.

D'un autre côté, on remarque trois blocs de forte corrélation négative : **contrainte-harmonique**, **contrainte-degré** et **contrainte-centralité**. Au fait, on peut expliquer ces deux dernières par la corrélation **positive** entre le degré d'un noeud et sa centralité.

On peut aussi, en visualisant l'ensemble des points, remarquer la relation de ces caractéristiques avec la **la meilleure modularité choisie** (figure 3).

D'abord, on remarque la forte discrimination de la variable **hub** telle que chaque modularité forme un cluster ce qui peut être dû au calcul de celle-ci : elle mesure l'importance d'un noeud en fonction de la quantité et de la **qualité** de ses liens vers des noeuds considérés comme des "hubs" dans le réseau.

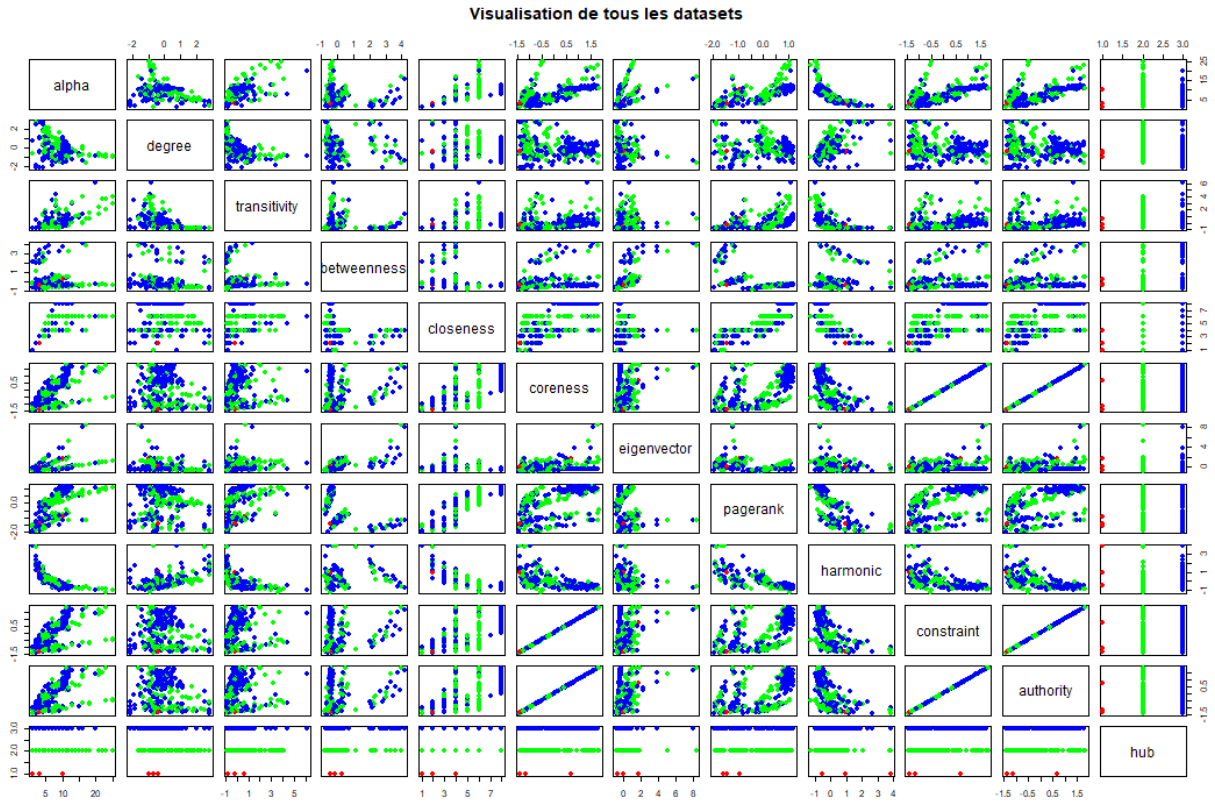

FIGURE 4 – Affichage des différentes variables

De plus, on remarque une "*pseudo-discrimination*" pour la centralité **proximité** telle que les noeuds ayant une proximité égale à **6** donnent une meilleure qualité avec la modularité **M**. Cependant, on n'arrive pas à remarquer un pattern clair pour les autres modularités.

5 Conclusion

De ce fait, on a pu proposer un modèe qui permettrait d'aider au choix de la meilleure modularité d'un noeud étant donnée ces caractéristiques topologique. De plus, l'atteinte de ces résultats permet aussi de confirmer la relation entre les aractéristiques topologiques et la modularité. Finalement, on porpose d'agrandir le champs de la question en se penchant sur la détection de la communauté locale et ce en connaissant les caractéristiques topologiques non seulement du noeud mais aussi du graphe.