Graph Mining & network science

Prédiction de liens (méthodes traditionnelles)

Machine learning sur les graphes

- **Prédiction de lien**

  un réseau évolue dans le temps : étant donné un graphe à un temps **t**, prédire les liens qui seront ajoutés dans l'intervalle **t → t'**

- **Identification de liens manquants**

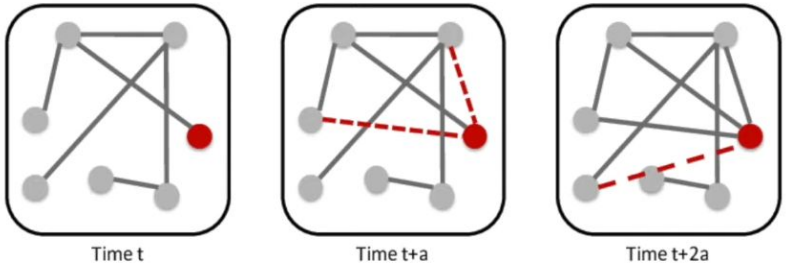
  étant un réseau, inférer les liens attendus par la structure mais manquants

- **Fiabilité des liens**

  estimer la fiabilité des liens dans un graphe

- **Quelles prédictions ?**

  existence, poids, type, etc.



Prédiction de liens

De nombreuses applications :

- recommandations d’amis sur Facebook
- recommandations de produits sur Amazon
- détection de fraude ou d’anomalies
- analyse des réseaux sociaux
- ...

Prédiction de liens

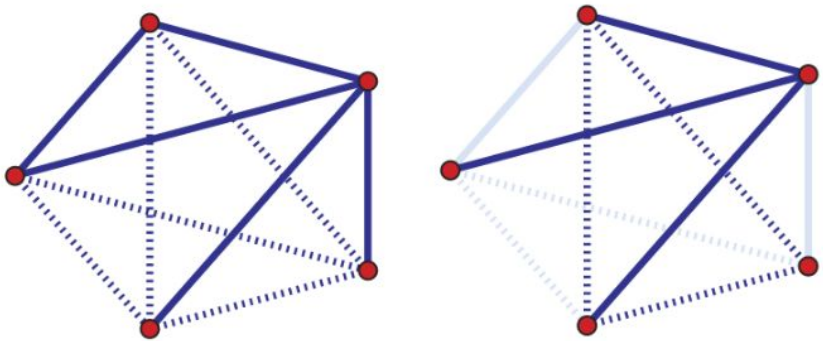
G = (V, E)

- Le nombre de liens potentiels :

  |V| . (|V| - 1) / 2 - |E|

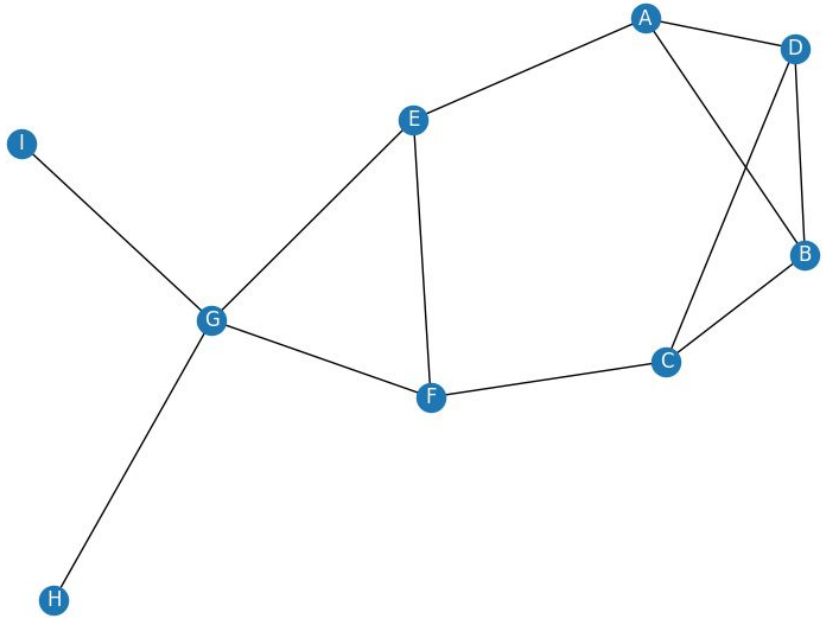
- Dans la grande majorité des cas, **$|E| << |V|^2$**

  et la probabilité de tirer au hasard le meilleur lien : **$O(|V|^{-2})$**



Prédiction de liens

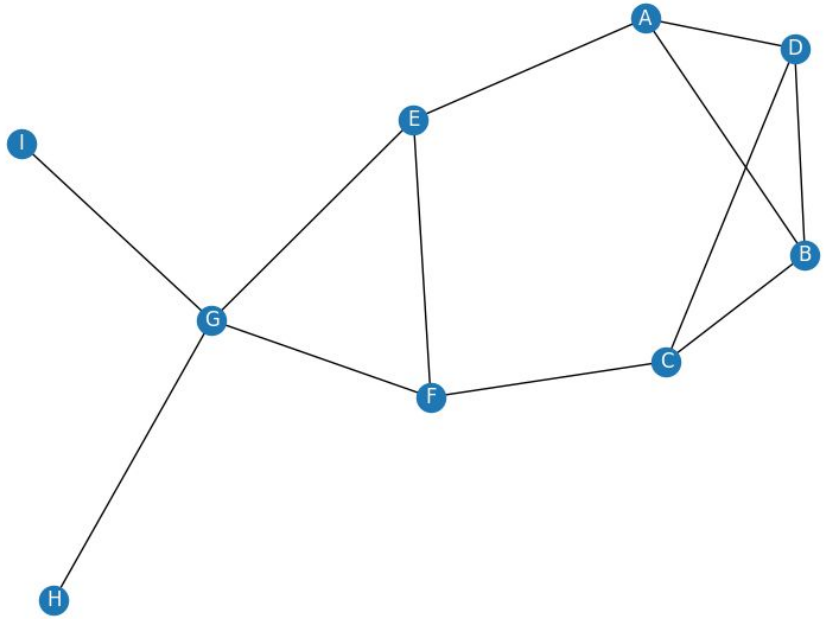
- Quels nouveaux liens sont susceptibles de se former dans le réseau ?
- Comment évaluer qu'une paire de noeuds est susceptibles de se connecter ?



Prédiction de liens

- Quels nouveaux liens sont susceptibles de se former dans le réseau ?
- Comment évaluer qu'une paire de noeuds est susceptibles de se connecter ?

Fermeture triadique : la tendance qu'ont les personnes qui partagent des connexions à se connecter.



Prédiction de liens par score de similarité

Méthodologie :

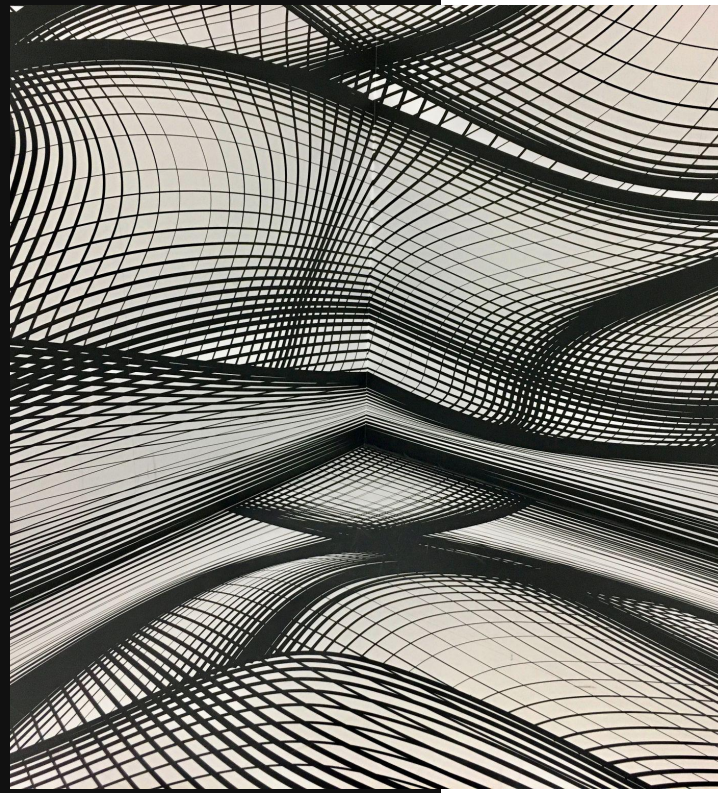
- pour chaque paire de noeuds **(x, y)**, **calculer un score s(x, y)**
- **trier** les paires par score décroissant
- **prédire** les top-n paires comme nouveaux liens
- **mesurer la qualité** de la prédiction en regardant la proportion de ces paires qui sont apparues au timestep ultérieur

The Link-Prediction problem for Social Networks,

Liben-Nowell and Kleinberg - 2003

Il nous faut des mesures de similarité entre deux noeuds !

Mesures de similarité entre 2 noeuds

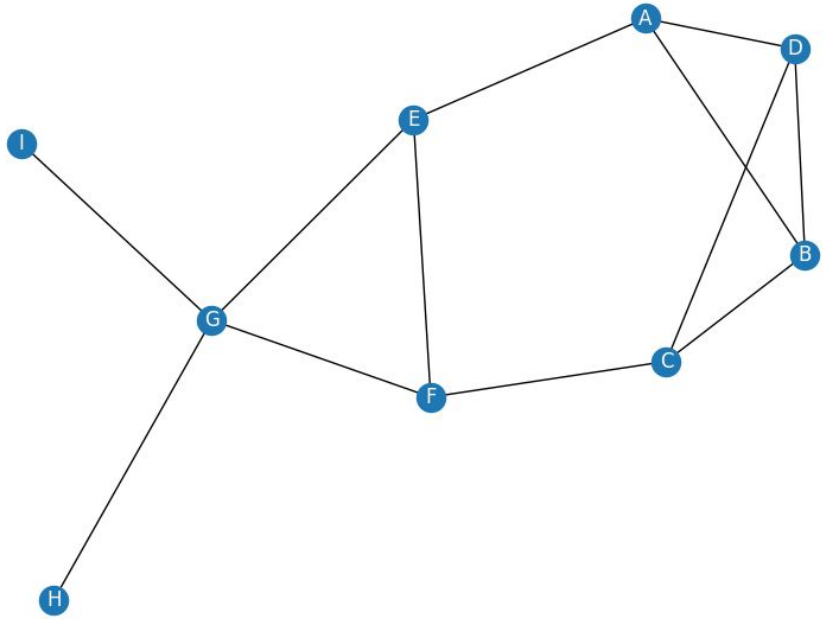


Score nodal - Preferential Attachment

- Dans le modèle d'attachement préférentiel, plus le degré d'un noeud est large, plus il attire de nouveaux liens.

$$s(i,j) = |N_i| \cdot |N_j|$$

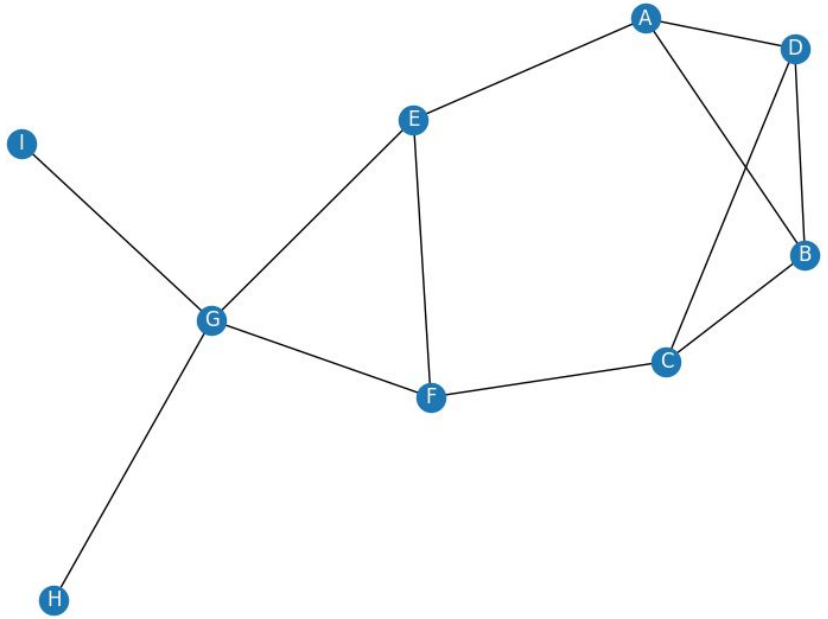
(ie. produit des degrés)



Score nodal - coefficient de clustering

- Plus un noeud a un coefficient de clustering élevé, plus il a de chance de créer de nouvelles connexions :

$$s(i,j) = cc(i) \cdot cc(j)$$

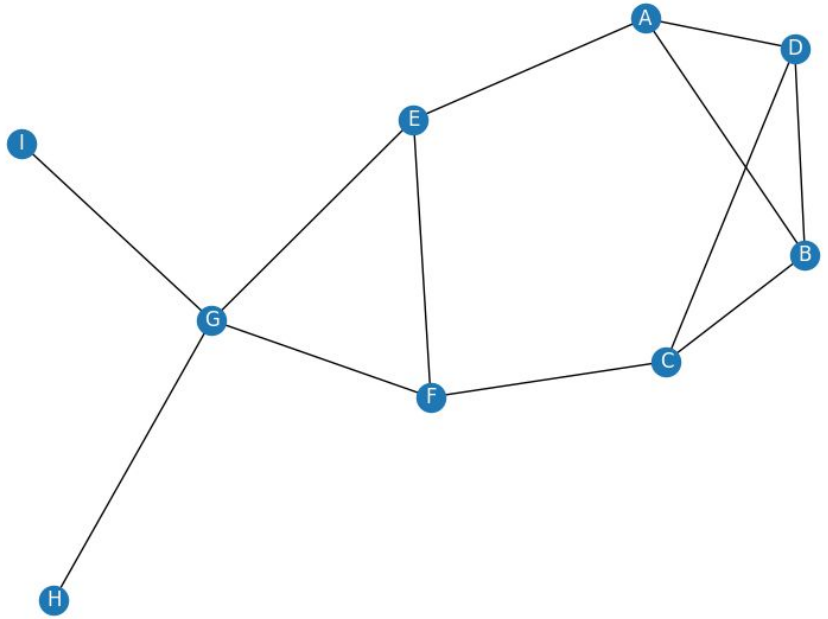


Score nodal - nombre de voisins communs

- Nombre de voisins en commun

$$s(i,j) = |N_i \cap N_j|$$

avec $\mathbf{N_i}$ l'ensemble des voisins de **i**.



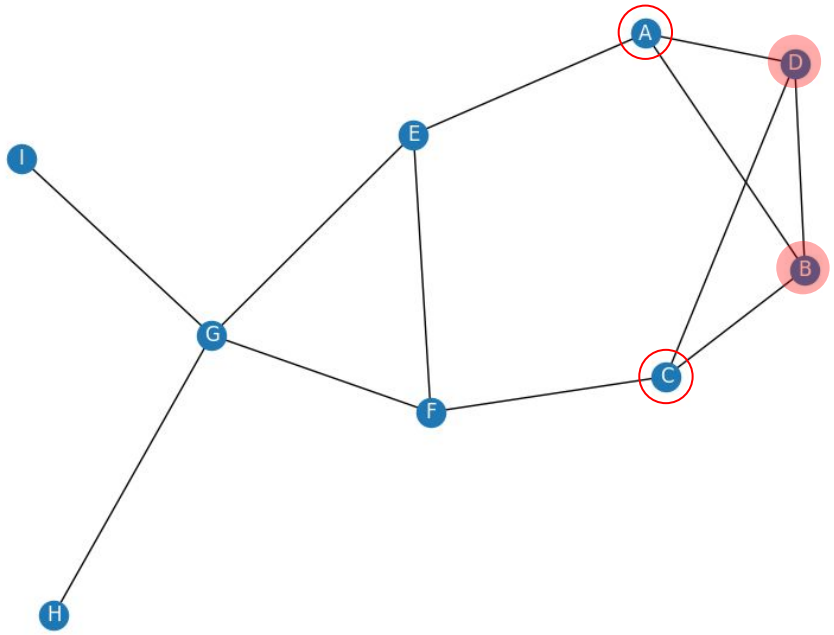
Score local - indice de Jaccard

- indice de Jaccard pour les noeuds i et j :

$$s(i,j) = \frac{|N_i \cap N_j|}{|N_i \cup N_j|}$$

Exemple :

$$s(A,C) = \frac{|\{B,D\}|}{|\{B,D,E,F\}|} = \frac{1}{2}$$

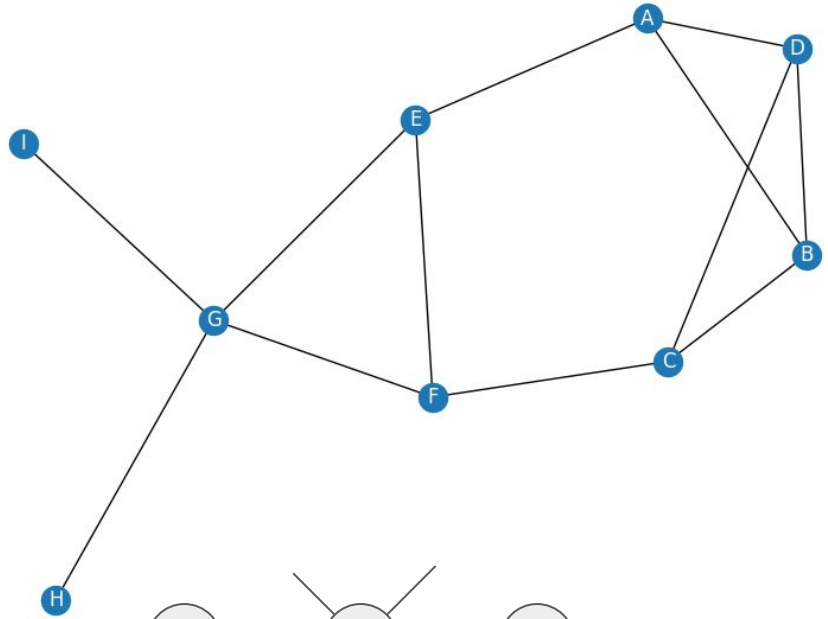


Ce score est similaire à l'edge-overlap de Granovetter

Score local - Resource Allocation Index

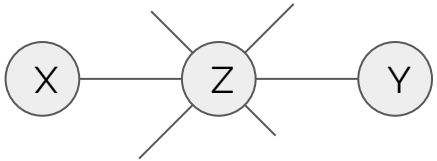
- La fraction d'une ressource qu'un noeud peut envoyer à un autre par l'intermédiaire de leurs voisins communs :

$$s(i,j) = \sum_{k \in N_i \cap N_j} \frac{1}{|N_k|}$$



Intuition :

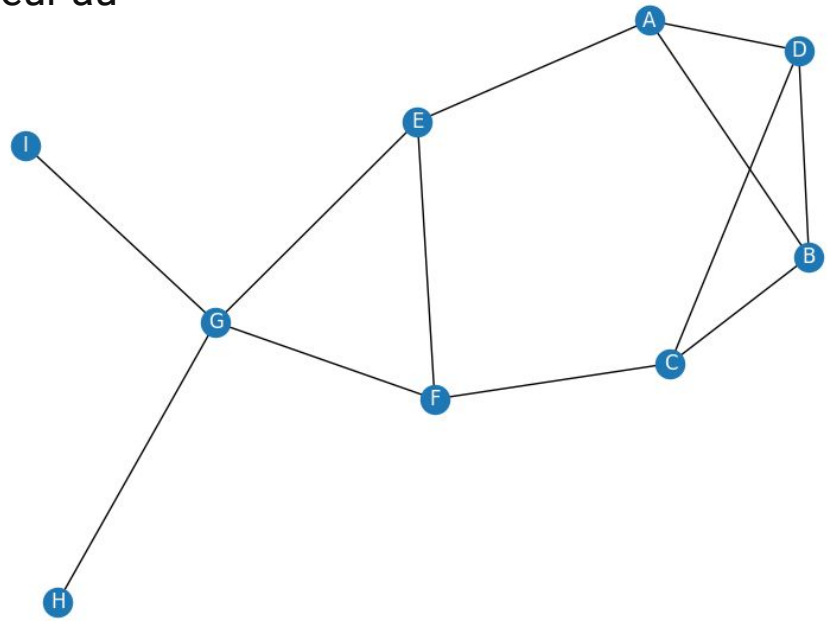
- **Z** a n voisins
- **X** veut envoyer un message à **Y**
- **Z** re-distribue uniformément ce message



Score local - Resource Allocation Index

- Similaire à l'index d'allocation de ressource, mais avec le dénominateur au logarithme

$$s(i,j) = \sum_{k \in N_i \cap N_j} \frac{1}{log(|N_k|)}$$



Scores par chemin

- Le plus court chemin entre deux noeuds : $s(i,j) = -shortest\_path(i,j)$
- Score de Katz : $s(i,j) = \sum_{l=0}^{\infty} \beta^l |path^{\langle l \rangle}(i,j)|$
- Hitting-time : $s(i,j) = -H_{ij}$
- Commute-time : $s(i,j) = -H_{ij} \cdot \pi_j - H_{ji} \cdot \pi_i$

Avec :

- H_{ij} = temps moyen pour une marche aléatoire partant de i d'atteindre j
- π = distribution stationnaire de la marche aléatoire : temps moyen passé à chaque noeud

Score par factorisation

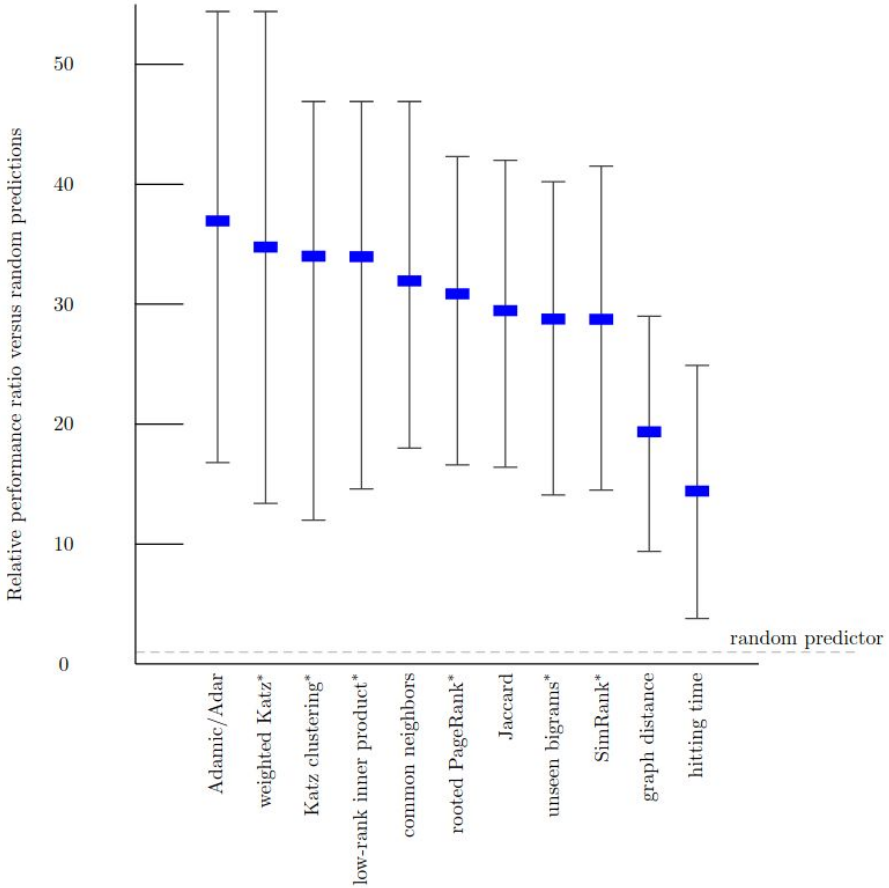
- Décomposition en valeurs singulières tronquée :

$$S = U_r \Sigma_r V_r^T$$

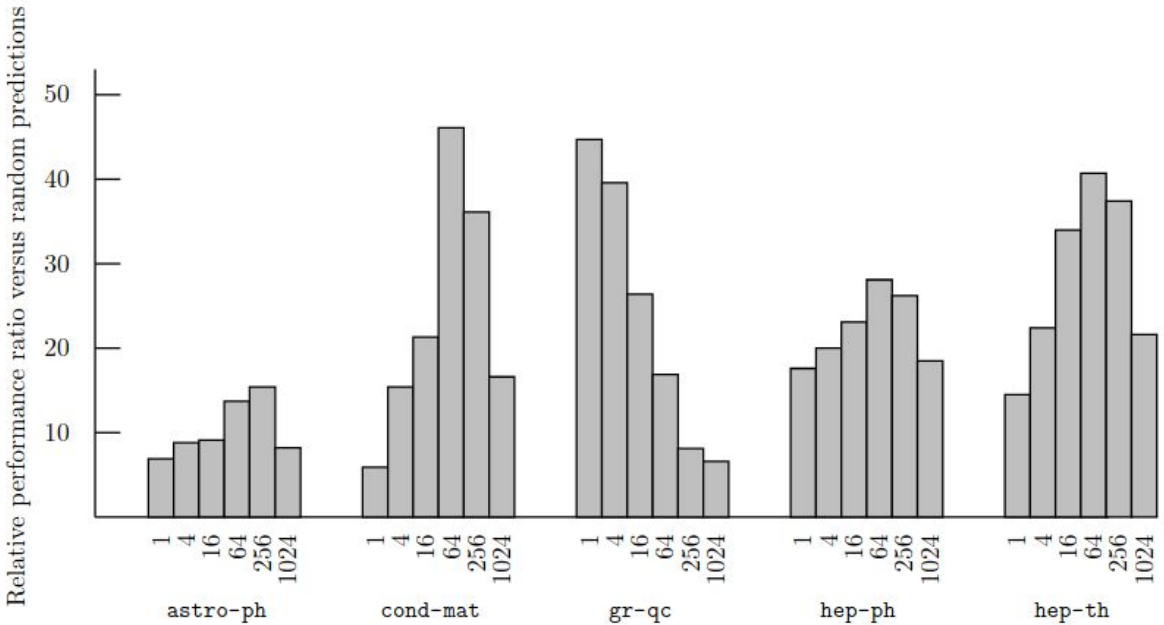
- Et :

$$s(i, j) = S_{ij}$$

Performances

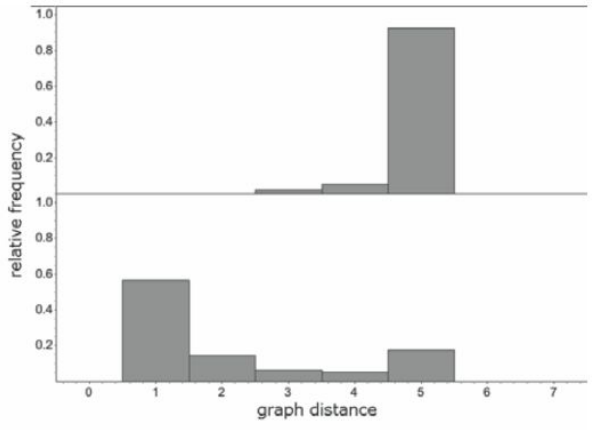


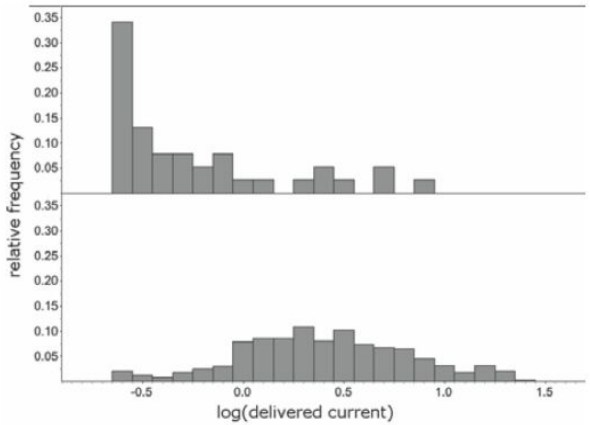
Performances

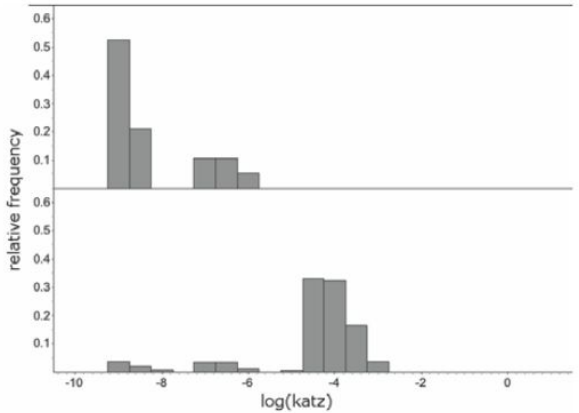


Performances

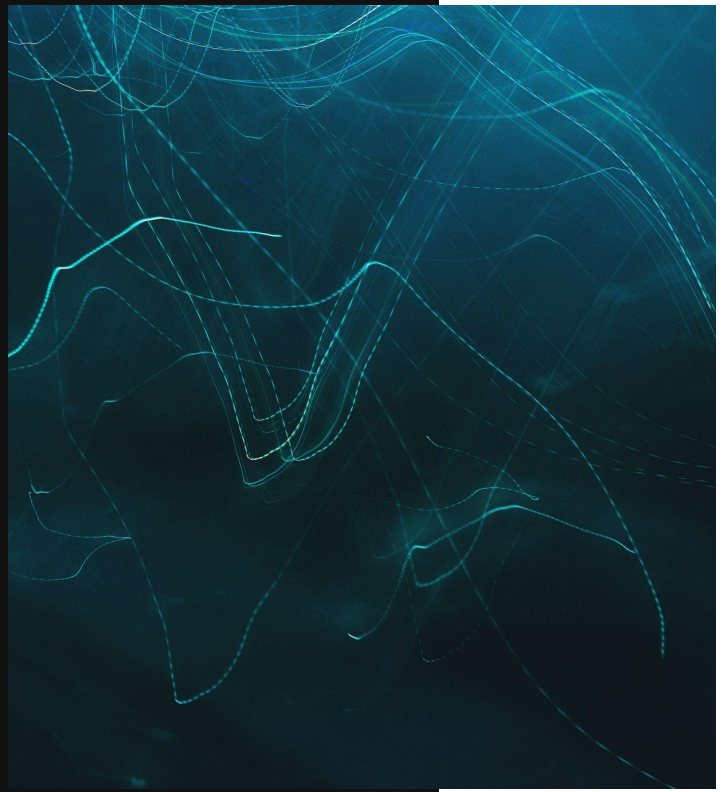
- De l’importance d’avoir des features discriminantes







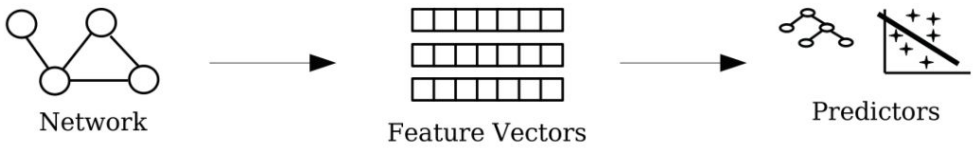
Link prediction : apprentissage supervisé



Apprentissage supervisé : création des features

Procédure :

- générations des features et des datasets (training et test sets)
- choix d'un classifieur : LogReg, Decision Tree, Random Forest...
- entraînement du classifieur sur les données
- évaluation du modèle



Apprentissage supervisé : création des features

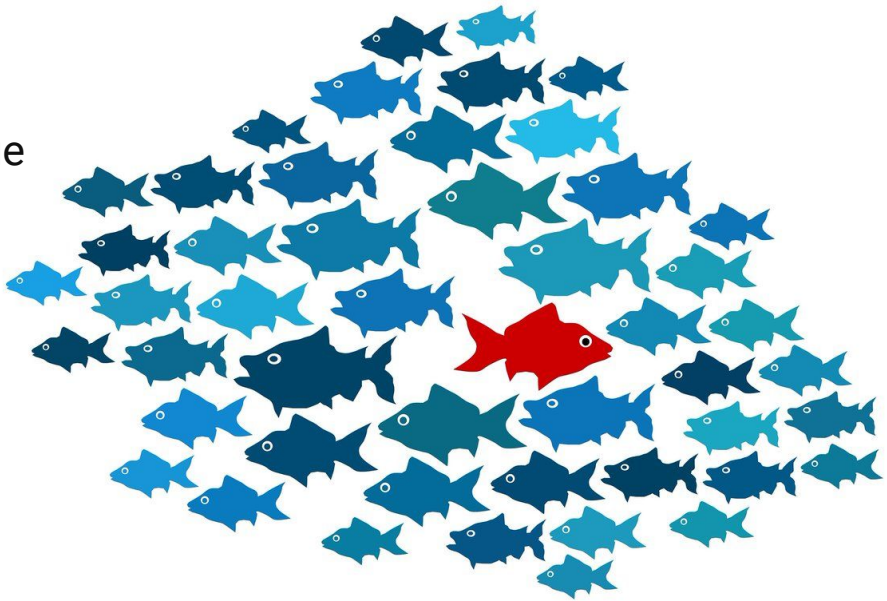
- On crée des features pour chaque paire **(i, j)** :
  - scores de proximité topologique vus précédemment
  - agrégation/combinaison de features des deux noeuds
  - features basées sur le contenu / attributs des noeuds

Difficultés : calcul lourd

- **Challenge 1 : un coût calculatoire des features très élevé**, puisqu'il faut évaluer la proximité d'un nombre très élevé de paires de noeuds : évolution quadratique des paires avec le nombre de noeuds

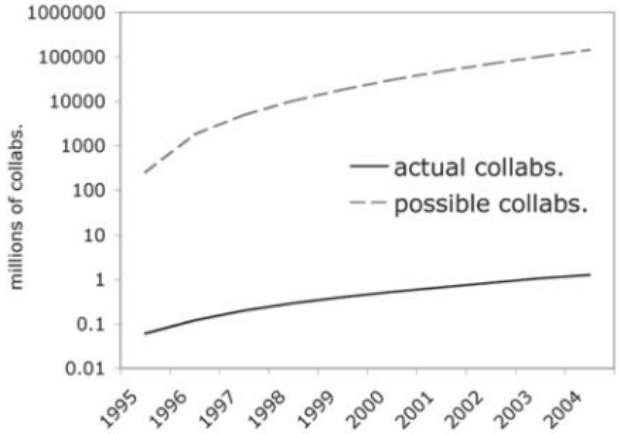
Difficultés : déséquilibre de classes

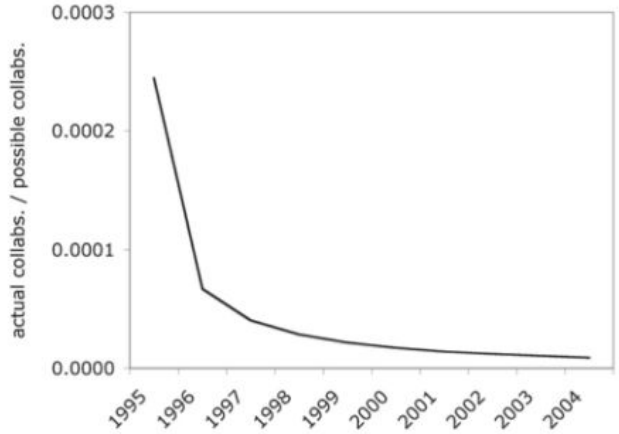
- **Challenge 2 :** un fort **déséquilibre dans la distribution des classes :**
  le nombre d'exemples positifs augmente beaucoup moins vite que le nombre d'arêtes possibles (exemples négatifs)



Difficultés : déséquilibre de classes

Collaborations réelles et possibles dans le dataset DBLP :





[The case for anomaly detection, Rattigan & Jensen - 2004]

Difficultés : déséquilibre de classes

Downsampling :
Créer un sous-ensemble aléatoire issu de la classe majoritaire, de même taille que la classe minoritaire.

ou

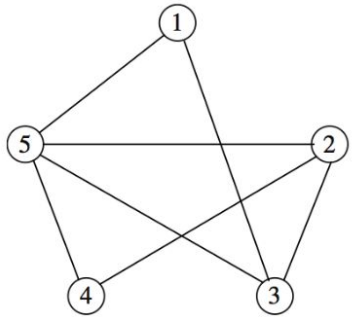
Upsampling :
Tirer un échantillon de la classe minoritaire, avec remise, jusqu'à ce que la classe minoritaire soit de même taille que la classe majoritaire.

Difficultés : déséquilibre de classes

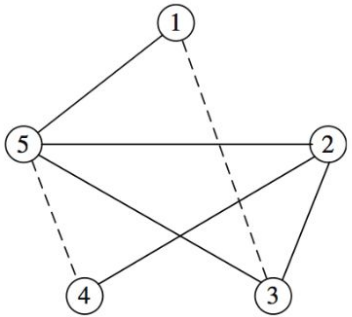
- **Challenge 3 : peu de data sets complets** en pratique

Solution :

Génération synthétique de données

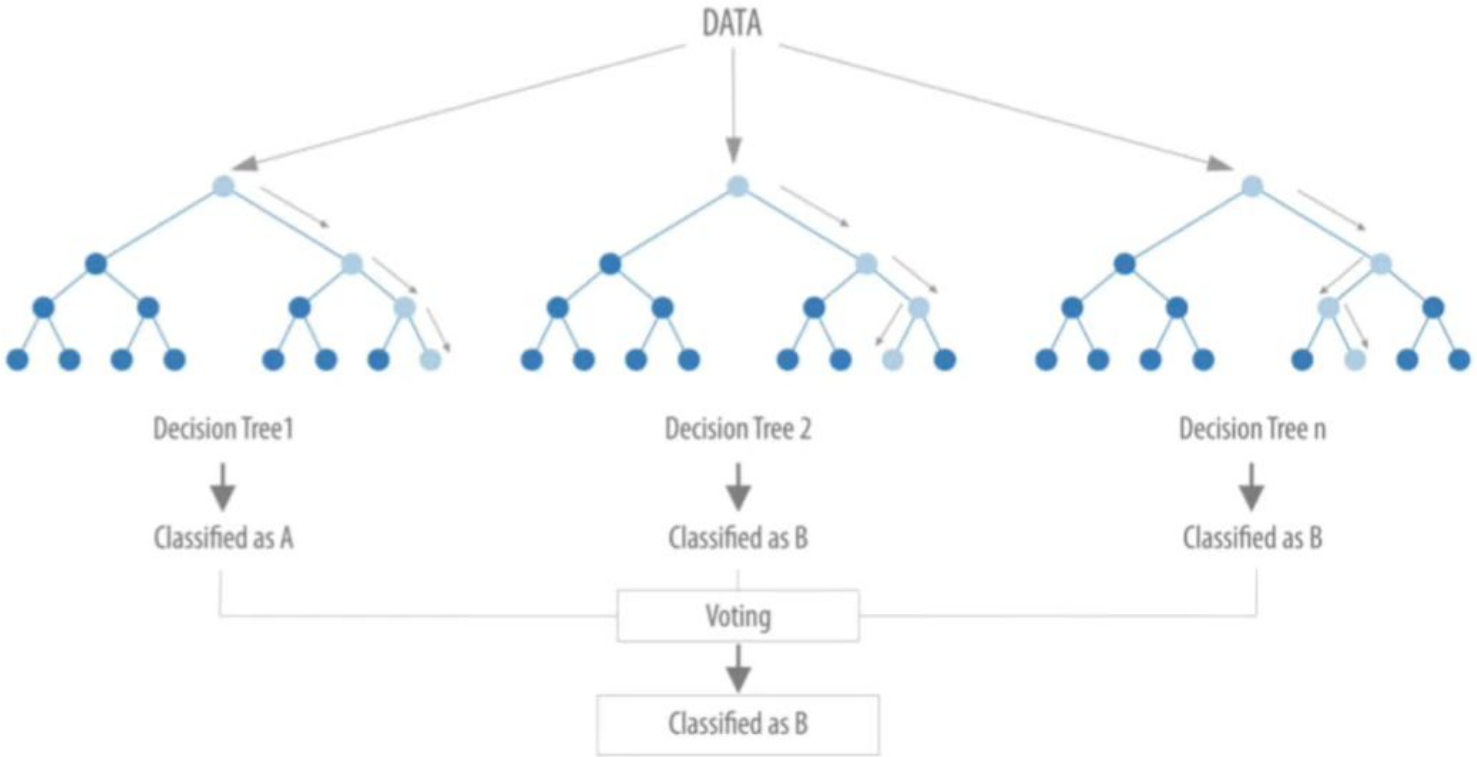


Whole graph

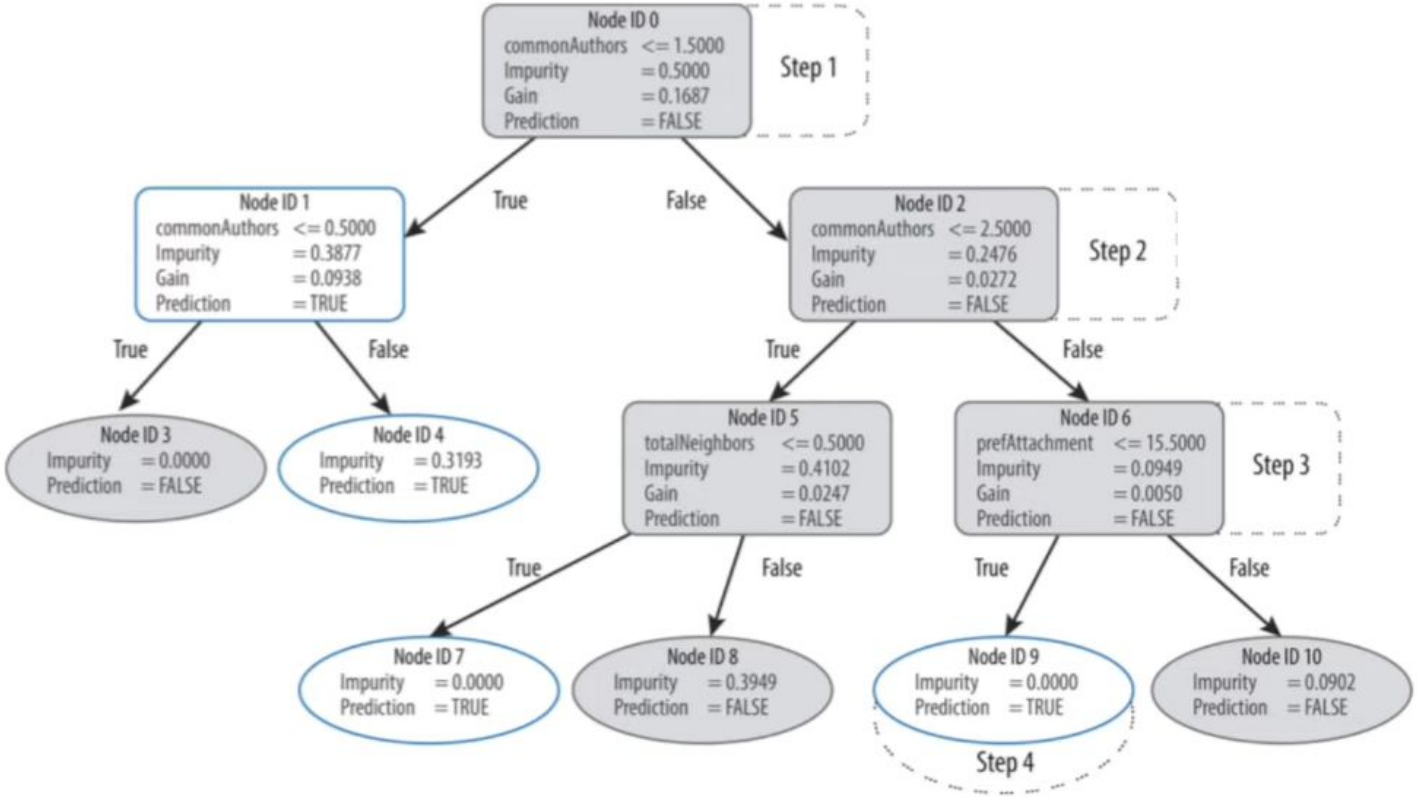


Training graph

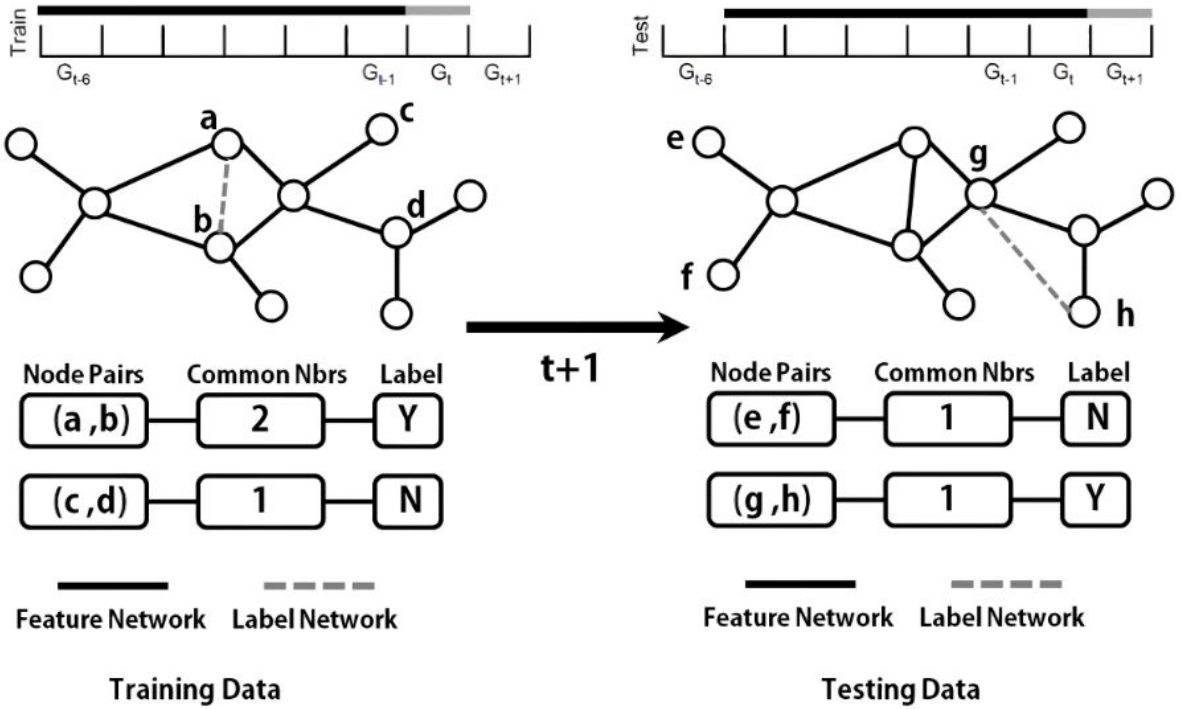
Classifieur : exemple avec un Random Forest



Classifieur : exemple avec un Random Forest



Evaluation - Training et testing data



Evaluation - métriques

$$Precision = \frac{TP}{TP + FP}$$

$$Recall = \frac{TP}{TP + FN}$$

$$F_1 = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$$

AUC / courbe ROC

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

Apprentissage supervisé - performances

Al Hasan - 2006

Classification model	Accuracy	Precision	Recall	F-value	Squared Error
Decision Tree	90.01	91.60	89.10	90.40	0.1306
SVM(Linear Kernel)	87.78	92.80	83.18	86.82	0.1221
SVM(RBF Kernel)	90.56	92.43	88.66	90.51	0.0945
K_Nearest Neighbors	88.17	92.26	83.63	87.73	0.1826
Multilayer Perceptron	89.78	93.00	87.10	90.00	0.1387
RBF Network	83.31	94.90	72.10	81.90	0.2542
Naive Bayes	83.32	95.10	71.90	81.90	0.1665
Bagging	90.87	92.5	90.00	91.23	0.1288

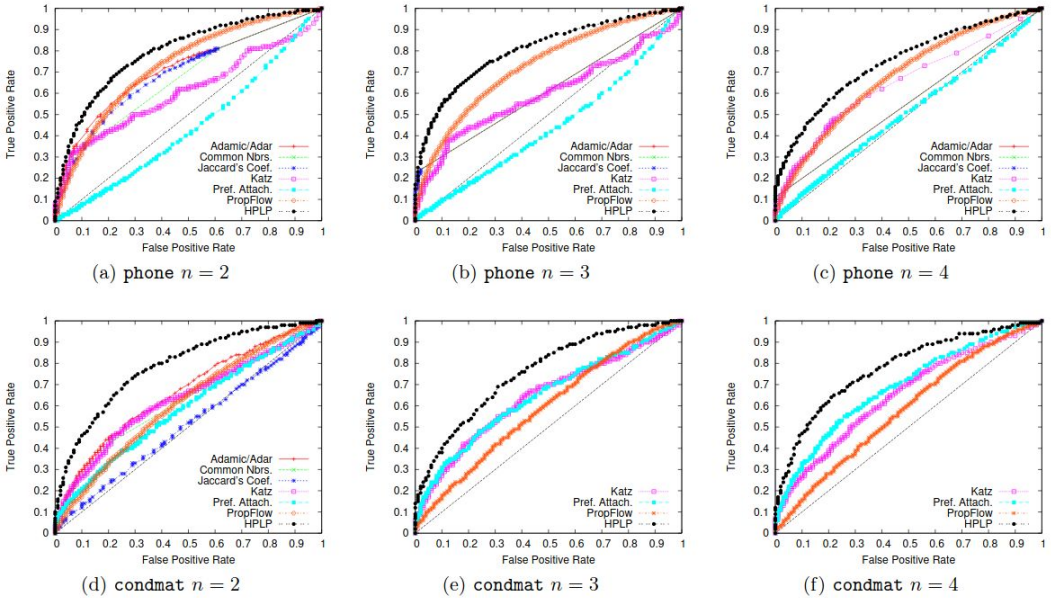
Table 2: Performance of different classification algorithms for BIOBASE database

Classification model	Accuracy	Precision	Recall	F-value	Squared Error
Decision Tree	82.56	87.70	79.5	83.40	0.3569
SVM(Linear Kernel)	83.04	85.88	82.92	84.37	0.1818
SVM(RBF Kernel)	83.18	87.66	80.93	84.16	0.1760
K_Nearest Neighbors	82.42	85.10	82.52	83.79	0.2354
Multilayer Perceptron	82.73	87.70	80.20	83.70	0.3481
RBF Network	78.49	78.90	83.40	81.10	0.4041
Naive Bayes	81.24	87.60	76.90	81.90	0.4073
Bagging	82.13	86.70	80.00	83.22	0.3509

Table 3: Performance of different classification algorithms for DBLP dataset

Apprentissage supervisé - performances

Lichtenwalter - 2010



(a) phone $n = 2$ (b) phone $n = 3$ (c) phone $n = 4$

(d) condmat $n = 2$ (e) condmat $n = 3$ (f) condmat $n = 4$

Table 2: Feature Listing

Name	Parameters	HPLP	HPLP+
In-Degree(i)	-	✓	✓
In-Volume(i)	-	✓	✓
In-Degree(j)	-	✓	✓
In-Volume(j)	-	✓	✓
Out-Degree(i)	-	✓	✓
Out-Volume(i)	-	✓	✓
Out-Degree(j)	-	✓	✓
Out-Volume(j)	-	✓	✓
Common Nbrs(i,j)	-	✓	✓
Max. Flow(i,j)	$l = 5$	✓	✓
Shortest Paths(i,j)	$l = 5$	✓	✓
PropFlow(i,j)	$l = 5$	✓	✓
Adamic/Adar(i,j)	-		✓
Jaccard's Coef(i,j)	-		✓
Katz(i,j)	$l = 5$, $\beta = 0.005$		✓
Pref Attach(i,j)	-		✓