TP 1 - Méthodes d'inférence de réseaux basées sur des mesures de co-expression

Olivier Goudet

February 7, 2024

Chaque TP de cette UE est noté. Il est à faire individuellement et à déposer avant la fin du semestre sur le dépôt Moodle de l'UE.

Exercice 1 : prise en main des données

Questions

1. Ouvrez en local un éditeur Python comme PyCharm et créez un projet avec l'environnement python3. Téléchargez les données d'entraînement sur Moodle.

2. Importez sous python les fichiers data1.csv et target1.csv avec la librairie pandas. Le fichier data1.csv correspond à la matrice d'expression des variables et le fichier target1.csv correspond à la liste des liens directs du réseau dirigé qui a permis de générer les données de façon artificielle.

3. Quel est le nombre de variables et le nombre de liens directs du premier réseau ? On notera d le nombre variables. Quel est le nombre de points d'observation pour ce dataset ?

4. Remplacez les données manquantes en complétant chaque valeur manquante d'une variable donnée par une valeur tirée aléatoirement parmi l'ensemble des données non manquantes pour cette même variable.

5. Normalisez les données pour que chaque variable soit de moyenne nulle et d'écart type 1.

6. Pour chaque lien direct du réseau entre deux variables, faites un affichage du nuage de points (en 2D). Par exemple pour le lien direct $V0 \rightarrow V6$ on affichera le nuage de points qui correspond aux points ($V0$,$V6$).

7. Avec quel type de modèle a été généré le réseau 1 selon vous ?

8. Ecrire une fonction qui permet de vérifier si le réseau 1 contient des cycles ou pas (cf slide 5 cours 3).

9. Répéter les étapes précédentes pour les 4 autres réseaux du dataset.

Exercice 2 : calculs de corrélation non signée

Questions

1. Avec le premier réseau (data1.csv), calculez toutes les valeurs absolues des coefficients de corrélation entre chaque paire de variable et afficher les résultats dans une matrice (matrice de corrélation non signée). On pourra utiliser la fonction pearsonr du module stats de la librairie scipy.

2. Affichez ensuite ces corrélations entre chaque paire de variable sous la forme d'une liste de lien "Cause","Effet","Score" dans un fichier "predictions_network1.csv", avec un tri suivant le score en valeur absolue du plus fort au plus faible.

3. Est-ce que cette méthode permet d'inférer un graphe dirigé selon vous ? Est-ce que les liens obtenus vous semblent cohérents avec le vrai réseau qui a généré les données (fichier target1.csv) ?

4. Affichez la courbe précision/recall obtenue avec cette méthode de corrélation dans le cas où le problème d'inférence de réseau est vue comme un problème de classification binaire de d^2 arcs, avec deux classes possible : 0, absence du lien $X_i \to X_j$, et 1, présence du lien $X_j \to X_i$. Voir les fonctions precision_recall_curve et PrecisionRecallDisplay de scikit-learn pour répondre à cette question.

5. Calculer le score de précision moyenne correspondant (cf. fonction average_precision_score de scikit-learn).

6. Appliquer ces mêmes étapes pour les 4 autres réseaux du dataset. Discuter les scores obtenus.

Exercice 3 : implémentation de la méthode WGCNA

L'objectif de cette partie est de réimplémenter la méthode WGCNA (*Weighted Gene Co-expression Network Analysis*) (voir slides 9-12, cours 4) à partir de la matrice de corrélation non signée calculée dans la partie précédente.

1. Appliquer un seuillage sur la matrice de corrélation non signée de façon à obtenir la matrice d'adjacence du réseau (cf. slide 10 cours 4).

2. Calculer la matrice TOM (*Topological Overlap Matrix*) (cf. slide 11 cours 4).

3. A partir de cette matrice TOM, effectuer une recherche de modules (ou clusters) avec une méthode de classification ascendante hiérarchique (voir la méthode AgglomerativeClustering de scikit-learn, avec l'option affinity="precomputed").

4. Est-ce que ces modules pour semble cohérents compte tenu du vrai réseau qui a généré les données ?

5. Une fois les clusters obtenus, réorganisez la matrice TOM pour faire un affichage comme présenté sur le slide 12.

Exercice 4 : information mutuelle

1. Calculez les informations mutuelles entre chaque paire de variables et affichez les résultats dans une matrice. On pourra effectuer un *binning* (cf. slides 35-36 du cours 2) avec la méthode *histogram2d* de numpy, puis calculer l'information mutuelle avec la méthode mutual_info_score de scikit-learn (champs *contingency*).

2. Calculer les scores de précisions moyennes obtenus pour les différents réseaux avec cette méthode de calcul d'information mutuelle paire à paire.

3. Comparer les scores obtenus avec ceux obtenus dans l'exercice 2 avec les corrélations. Pour quels datasets les calculs d'information mutuelle semblent mieux ou moins bien marcher que la corrélation de Pearson ? Pourquoi ?

Exercice 5 : implémentation de la méthode ARACNE

1. Implémentez la méthode ARACNE (cf. slides 18-20) qui prend en entrée une matrice d'information mutuelle ou bien une matrice de corrélation non signée, comme celles calculées dans les exercices précédents.

2. Afficher les liens supprimés par la méthode DPI (cf. slide 18).

3. Est-ce que cette suppression de liens vous semble cohérente par exemple pour le réseau 1 ?