TP2.2 : états de marché

13 janvier 2023

TP à rendre.

1 Détermination d'états de marché

— Télécharger le code R ou Python pour le clustering par la méthode de Louvain
— Télécharger les données us_equities.csv.gz

1. Charger les données.
2. Choisir une taille de fenêtre de calibration T.
3. Choisir un univers de titres (colonnes). Rappel : pour la détermination des états de marché, $N > T$ est indispensable.
4. Pour chaque fenêtre de calibration
   (a) supprimez les colonnes avec des NA
   (b) appliquer l'algorithme de Louvain selon la méthode de MacMahon et Garlaschelli pour obtenir les états μ_t, $t \in \{t_0, \cdots, t_1\}$.
5. Tracez le nombre d'états de marché en fonction du temps.

2 Stratégies basées sur les états de marchés (à choix avec 3)

Il s'agit d'inventer des stratégies qui exploitent les états de marchés.

1. Pour une fenêtre temporelle de longueur bien choisie, déterminer les états de marchés.

2. Etant donné l'état de marché du dernier pas temporel de la fenêtre de calibration, trouver un critère pour déterminer dans quels titres investir. Note : on peut imaginer tenir la position un jour, ou plus.

3. Pour appliquer cette cette stratégies les jours suivants, trois possibilités,
   (a) décaler la fenêtre temporelle de 1 pas et appliquer les points ci-dessus ;
   (b) utiliser une approche des plus proches voisins par rapport aux rendements moyens d'un état de marché ;
   (c) un mélange des deux.
4. Calculer la performance cumulée de cette stratégies (avec au moins 100 fenêtres de calibration).

3 Prédiction de la direction d'un titre financier avec de l'apprentissage machine (à choix avec 2)

Le but est d'utiliser l'apprentissage machine pour trouver une stratégie qui exploite les états de marché.

1. Choisir un titre (une colonne) dont prédire les rendements R_{t+1} au temps t

2. Pour chaque fenêtre temporelle de calibration $\{t_0, t_1\}$, construire une matrice de prédicteurs à partir des K derniers états de marché.

$$P_{t_0,t_1} = \begin{pmatrix} \mu_{t_0+K-1} & \mu_{t_0+K-2} & \cdots & \mu_{t_0} \\ \vdots & \vdots & \vdots & \vdots \\ \mu_{t_1} & \mu_{t_1-1} & \cdots & \mu_{t_1-K+1} \end{pmatrix}$$

3. Utiliser un paquet d'apprentissage machine (`e1071, randomForestSRC, scikit-learn`, etc.) de votre langage préféré et pour chaque t, apprendre le signe des rendements de R

$$\begin{pmatrix} \mu_{t_0+K-1} & \mu_{t_0+K-2} & \cdots & \mu_{t_0} \\ \vdots & \vdots & \vdots & \vdots \\ \mu_{t_1-1} & \mu_{t_1-2} & \cdots & \mu_{t_1-K} \end{pmatrix} \sim \begin{pmatrix} \text{sign } R_{t_0+K} \\ \vdots \\ \text{sign } R_{t_1} \end{pmatrix}$$

   Notez que la dernière ligne de P_{t_0,t_1}est gardée en réserve pour prédire R_{t_1+1}

4. Prédire sign R_{t_1+1} (fonction `predict` appliquée à la dernière ligne de P_{t_0,t_1})
5. Notez cette prédiction x_{t_1+1}. C'est également le pari à prendre au temps t_1, donc le rendement de la stratégie est $g_{t_1} = x_{t_1+1} r_{t_1+1}$.
6. Itérez sur le temps. Voir point 3. de l'exercice 2.
7. Calculez et tracez $G_t = \prod_{t' \leq t}(1 + g_{t'})$ et commentez.