Il est important que les sociétés de cartes de crédit soient en mesure de reconnaître les transactions frauduleuses par carte de crédit afin que les clients ne soient pas facturés pour les articles qu'ils n'ont pas achetés.

Les jeux de données contiennent des transactions effectuées par des titulaires de carte européens. Cet ensemble de données présente les transactions qui ont eu lieu en deux jours, où nous avons moins de 500 fraudes sur presque 285.000 transactions. L'ensemble de données est très déséquilibré, la classe positive (fraudes) représente 0,2% de toutes les transactions.

Les fonctionnalités V1, V2,… V28 sont les principaux composants obtenus avec PCA, les seules fonctionnalités qui n'ont pas été transformées avec PCA sont 'Time' et 'Montant'. La fonction «Time» contient les secondes écoulées entre chaque transaction et la première transaction de l'ensemble de données.

La fonction 'Classe' est la variable de réponse et prend la valeur 1 en cas de fraude et 0 sinon.

Le dataset est téléchargeable sur :

https://wetransfer.com/downloads/1992fa4c2a53099d36cc27ee4ffc1c2920200415202716/8a1227d9efe35688dc85300dc91f649c20200415202731/8b8cfc

Dans un premier temps importez les données en utilisant les scripts Python afin de les charger pour démarrer la classification.

1. Quelles les prétraitements obligatoires avant d'appliquer l'algorithme Random Forest de Python (Package SkLearn)?

2. Trouver le meilleur paramétrage afin d'avoir le taux d'erreur minimum sur le classifieur.

    a. *class* `sklearn.ensemble.`**`RandomForestClassifier`**(*n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='auto', max_leaf_nodes=None, min_impurity_decrease=0.0, min_impurity_split=None, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None*)

    b. Quel sont les meilleurs paramètres utilisé afin que le modèle Random Forest soit le plus performant ?

    c. Justifier le choix des valeurs utilisées lors du build du modèle Random Forest

3. Quel est le taux d'erreur de votre modèle random forest ?

4. Quel est le meilleur classifieur (ou estimator) de votre random forest ?

5. Affichez le taux d'erreur de tous les estimator de votre modèle Random Forest

6. Prenez deux estimators de votre modèle Random Forest, l'estimator le plus performant et l'estimator le moins performant :

    a. Quels le taux d'erreur des deux estimatiors ?

7. Quelle est la différence entre la moyenne des taux d'erreur des deux estimators et le taux d'erreur de votre meilleur modèle random Forest ?

8. Affichez les deux arbres correspondants au deux estimators

9. Ecrire la liste des règles de décision extraites des deux estimator sous forme de Si A , B => C (x %) ?

10. Quelles les règles dont un besoin le mauvais estimator pour qu'il s'approche des performances du bon estimator ?

Les package python sont à mettre dans une annexe de votre document et à joindre dans la réponse à ce projet.

Le travail est prévu pour 2 personnes, s'il vous est impossible, une personne peut soumettre le travail seul, mais il sera noté de la même façon.

Pour information, ceux qui ne peuvent utiliser Python auront la possibilité d'utiliser R , pour ce une explication est requise .