PROJET : CLASSIFICATION AUTOMATIQUE

A RENDRE POUR LE 15 MAI AU PLUS TARD

Consignes

1) Le projet est à traiter avec R et PyTorch.
2) Un seul document est à rendre, constitué de trois parties distinctes. La première partie porte sur le pré-traitement des données. La deuxième partie porte sur le travail demandé en R (cf. Dossier 1 ci-dessous), et la troisième sur l'apprentissage profond sous PyTorch (cf. dossier 2).
3) Le document à rendre doit être en format word ou pdf.
4) Les codes commentés sont à restituer dans le projet, les liens vers colab sont aussi à fournir.
5) Les tableaux des données pré-traitées sont aussi à rendre.
6) Les dossiers sont à rendre dans Teams. Tout dossier envoyé par mail ne sera pas corrigé.

Contexte

Le churn ou taux d'attrition correspond pour une entreprise à la proportion de clients perdus au cours d'une période. Par exemple, dans le secteur de télécommunication, il peut faire référence aux clients qui changent d'opérateur. Ainsi, le churn peut désigner la résiliation d'un forfait ou plus généralement d'un contrat qui lie le client et l'entreprise.
Le churn volontaire est souvent le résultat d'une insatisfaction sur l'offre ou la qualité des services, il faut alors détecter au plus tôt ces motifs d'insatisfaction afin d'y apporter une réponse appropriée, et au plus vite. Mais encore faut-il être prudent, car les actions commerciales anti-churn peuvent parfois être mal perçues par le consommateur qui s'étonnera par exemple d'une offre promotionnelle attractive qui interviendrait tardivement. Le recours aux algorithmes de prédiction permet ainsi de détecter très tôt des signaux faibles laissant penser à une attrition possible du client. Une fois le profil détecté, il faudra encore le scorer afin d'engager la démarche la mieux adaptée pour retenir le consommateur.
L'objectif de ce projet est d'analyser les données des clients d'une entreprise de télécommunication afin de créer des modèles prédictifs pour identifier les clients à haut risque de résiliation et identifier les principaux indicateurs de cette résiliation. Souvent, un client ne décide pas de passer instantanément à un concurrent, mais plutôt sur une certaine période de temps. Dans la prédiction du taux de résiliation, nous supposons que le cycle de vie du client comporte trois phases :

- La phase « bonne » : dans cette phase, le client est satisfait du service et se comporte comme d'habitude.
- La phase « action » : la satisfaction du client commence à se dégrader. Par exemple, il reçoit une offre convaincante d'un concurrent, fait face à des frais injustes, devient mécontent de la qualité du service, etc. Dans cette phase, le client montre généralement un comportement différent de celui des « bons » mois. En outre, il est essentiel d'identifier les clients à haut risque de résiliation dans cette phase, car certains « gestes câlins » peuvent être faits à ce stade.
- La phase de « résiliation » : dans cette phase, le client s'est désabonné.

Dans ce projet, nous disposons des informations concernant un groupe de clients sur une période de quatre mois consécutifs (juin, juillet, août et septembre). Les mois sont codés 6, 7, 8 et 9. Nous allons considérer que les deux premiers mois correspondent à la phase « bonne », le troisième mois à la phase « action », et le quatrième mois à la phase « résiliation ».

Données

La base de données comporte 99999 clients/lignes et 226 informations observées chez ces clients/colonnes sur les quatre mois codés 6, 7, 8 et 9.

La variable à expliquer y qui prend une valeur binaire (y=1 si le client a résilié au mois 9 et 0 sinon) est à construire comme suit : les clients qui n'ont effectué aucun appel entrant (ic) ou sortant (og) ET n'ont pas utilisé les données mobiles (2g_mb ou 3g_mb) le mois de septembre (9) sont ceux qui ont résilié (y=1). Les variables que vous devez utiliser pour construire y sont donc « total_ic_mou_9 », « total_og_mou_9 » « vol_2g_mb_9 » et « vol_3g_mb_9 » avec mou= Minute Of Usage.

Après avoir labélisé les churners, supprimer toutes les variables correspondant à la phase de churn (toutes les variables ayant «_9», dans leurs noms). En effet, vue que le taux de résiliation est défini en fonction du mois 9, et que la prédiction est faite au mois d'août (8), le mois d' « action », les données du mois 9 ne seront pas disponibles pour la prédiction.

Partie 1 : Préparation des données

Cette partie peut être traité en R ou en Pytorch. Avant d'utiliser vos données, vous devez les nettoyer (enlever les doublons s'il y en a, détecter les valeurs aberrantes et retirer les clients concernés, supprimer les variables qui présentent plus que 10% de valeurs manquantes NAN, remplacer les valeurs manquantes NAN dans les variables contenant moins de 10% de NAN par 0 si la variable est binaire et par la valeur médiane de cette variable autrement, supprimer les variables non numériques, etc. A la fin de ces traitements, vous obtenez une base à n lignes et p colonnes.

Diviser la base de données en n'=80%*n observations pour l'apprentissage et n''=20%*n observations pour le test (n=n'+n''). Sauvegarder ces répartitions dans deux fichiers excel (à restituer avec les codes) et utiliser les dans toute la suite du travail.

Partie 2 : Travail à effectuer sous R

1. Centrer-réduire vos données d'apprentissage.

2. Utiliser une ACP pour réduire la dimension p en p' composantes principales que vous allez fixer en justifiant votre choix.

3. Sur cette nouvelle base de données à n' observations, p' variables, construisez un classifieur.

    a. D'abord un SVM, linéaire, à noyau gaussien, à noyau polynomial, tout en optimisant tous les paramètres nécessaires, à savoir le seuil de tolérance C et les paramètres du noyau. Listez dans un tableau le jeu de paramètres choisi avec les performances obtenues. Retenez le meilleur SVM et expliquez votre choix.

    b. Ensuite un réseau de neurones, en variant le nombre de couches cachées et les fonctions d'activation. Reportez les performances des modèles appris dans un tableau et retenez le meilleur réseau.

4. Centrer-réduire les données de test en utilisant la moyenne et l'écart-type calculer sur l'ensemble d'apprentissage. Projetez les données de test sur le plan factoriel que vous avez trouver en 2) pour réduire la dimension de la base de test de p à p' variables. Ensuite :

a. Appliquer le SVM retenu sur la base de test à n'' observations et p' variables et donner les performances. Commenter ce résultat.

b. Idem pour le Réseau de neurones.

5. Refaire les étapes 3 et 4 sur les données centrées-réduites à p variables (sans faire l'ACP).

6. Comparer les performances des quatre classifieurs (SVM, Réseau de neurones, ACP+SVM, ACP+Réseau de neurons).

Partie 3 : Travail à effectuer sous PyTorch

Pour cette partie, il est fortement conseillé de commencer par les tutoriels sur Colab et sur l'apprentissage profond avant de passer à la partie travail à réaliser.

I. Environnement : Utilisation de la plateforme Colab de Google

Pour limiter les problèmes matériels, il est recommandé d'utiliser le service Colab proposé par Google : https://colab.research.google.com.
Colab permet l'exécution jusqu'à 24h consécutives de notebooks jupyter sur des serveurs mis à disposition par Google. Les notebooks seront à placer dans un Google Drive, ainsi que tous les fichiers utilisés dans le notebook (ie : le dataset). Les sorties du programme sont aussi stockées sur le Google Drive (attention donc à ne pas lire/écrire de données trop volumineuses). Pour choisir l'accélération matérielle GPU, rendez-vous dans le menu Modifier/Paramètres du notebook...

Un petit guide sur Colab :
https://ledatascientist.com/google-colab-le-guide-ultime/

II. Initiation au Deep Learning avec PyTorch

PyTorch fournit les modules et classes conçus torch.nn, torch.optim, Dataset et DataLoader pour vous aider à créer et à former des réseaux de neurones dans une forme concise et flexible. Afin de mieux comprendre ces modules et pouvoir les personnaliser pour votre problème, je vous recommande le tutoriel ci-dessous, qui va vous permettre :

- De comprendre la différence entre Torch et Pytorch et vous apprendre à installer PyTorch (si vous souhaiter travailler en local),
- Comprendre les étapes d'un cycle d'apprentissage par Pytorch,
- Développer un réseau profond pour la régression, la classification binaire et multiple, et un réseau convolutif profond (CNN profond) pour la classification des images (CNN facultatitf).

https://machinelearningmastery.com/pytorch-tutorial-develop-deep-learning-models/

III. Travail demandé

Commencer par un petit récapitulatif sur les compétences acquises par les tutoriels.

Ensuite, centrer-réduire la base d'apprentissage à n' observations et p variables et apprenez un réseau profond. Décrire le réseau en détails : nombre d'entrées, de sorties, de couches cachées, de neurones par couche, fonction d'activation utilisée dans chaque couche, fonction coût choisie ainsi que l'algorithme de rétropropagation utilisé. Entrainer votre modèle en mode CPU, GPU et TPU. Reporter à chaque fois les performances ainsi que le temps de calcul.

Centrer-réduire la base de test à n'' observations et p variables en utilisant la moyenne et l'écart-type calculés sur la base d'apprentissage. Calculer les performances obtenues par votre modèle sur la base du test. Comparer ces résultats à ceux obtenus avec les méthodes d'apprentissage classiques.