Upsell/Cross sell LAB

Ce Lab est constitué de questions pratiques sur la datascience, de niveau basique à moyen pour vérifier votre aptitude à maîtriser les bases de la datascience (machine learning et feature engineering).

Ne pas passer plus de 30min sur chaque point théorique (sauf si vous voulez approfondir le sujet hors cadre du LAB)
Ne travaillez pas sur un dataset de plus de 5000 lignes afin de ne pas attendre trop longtemps les résultats

Le lien de la data : https://www.openml.org/search?type=data&sort=runs&id=42759&status=active (~1.7GB)

Questions :
PREPROCESSING

1. En utilisant grep, éliminer les lignes de type commentaire (starts with « % ») et écrire le résultat dans un autre fichier. Avec un outil unix, comptez le nombre de lignes dans chaque fichier.
2. En utilisant uniquement des outils unix de bash, séparer la donnée tabulaire de la donnée header de deux manière différentes: a/ identifiez la ligne ou commence la data et séparer grâce au numéro de ligne et b/ une autre méthode de votre choix en utilisant par exemple un séparateur entre les deux blocs.
3. En utilisant uniquement des outils unix de bash, construire une liste en ligne de type des 15.000 variables sous force de { 'Var1': int, 'Var2': float, ... } (c'est un dtype)
4. Refaire la création du dtype en utilisant python et sans charger l'ensemble de la donnée
5. Charger la donnée en utilisant directement une api de type load ARFF (chercher les librairies correspondantes).
6. Charger la donnée (à partir de la balise @Data) sous une forme brute en utilisant uniquement pandas sans utiliser de dtype spécifique (pandas va inférer tout seul). Quel est le nombre maximal de lignes qu'il est possible de charger avec 1GB de RAM ?
   Point théorique : Types of memory speed
7. Charger la donnée correspondante à 1GB de RAM à présent avec un dtype qui part de la liste générée précédemment avec le header. Quel est, cette fois-ci, le nombre maximal de lignes qu'il est possible de charger avec 1GB ?
   Points théoriques : pandas dtypes
8. Optionnel : Tester le dtype category de pandas pour économiser au maximum de la mémoire, mais attention, ce format peut poser des problèmes par la suite.
9. Existe-il des variables avec un nombre important de missing values ? Choisissez un threshold de 10% de variables manquantes et supprimez les variables correspondantes.
10. Pour les variables ayant moins de 10% de valeurs manquantes, utilisez un fill par une valeur constante de votre choix. Produisez ensuite le dataframe résultant sans les variables qui ont plus de 10%.
11. Refaites le même travail avec
    a. un forward fill sur les valeurs manquantes
    b. un fill des missing values en utilisant la valeur majoritaire
    c. une suppression de toute ligne ayant une valeur manquante. Quel pourcentage de ligne reste-il après cette dernière opération ? Est-ce plus avantageux d'utiliser un fill missing ou de faire un drop missing dans ce cas ?
12. Quelles autres techniques de missing values sont possibles ?
    Point théorique : lire et comprendre la librairie datawig

13. Choisissez un pourcentage de train test split et appliquez le au dataset.
14. Persistez votre data de train et votre data de test aux formats parquet, csv et h5. Quelle est la différence en termes de taille ? et d'utilisabilité entre ces 3 formats ?
    Point théorique : Format HDF5, Format Parquet

FEATURE ENGINEERING 1 - XGBOOST

1. Pourquoi XGBOOST peut fonctionner avec des variables catégorielles ?
   Point théorique : XGBoost enable_categorical
2. Utilisez un xgboost avec les paramètres par défaut et calculez la precision, recall et accuracy avec la fonction predict classique. Quel threshold utilise cette fonction pour donner des prédictions ? Essayer de recalculer la precision, le recall et l'accuracy en changeant le threshold et trouver un threshold optimal qui équilibre precision et recall.
   Point théorique : precision, recall, accuracy
3. Changer la métrique d'évaluation pour une AUC de XGBoost et relancer le code, pour comparer les métriques finales.
4. Calculer l'AUC et visualisez la ROC curve à présent avec le même modèle et trouvez ce qui semble l'optimal de la ROC Curve, quitte à augmenter le nombre d'arbres du xgboost et la profondeur maximale.
   Point théorique : AUC, ROC-curve
5. Augmenter la profondeur maximale et constatez si l'AUC s'améliore. Quel est le risque de trop augmenter max_depth des algorithmes de ce type ?
   Point théorique : Vapnik-Chervonenkis dimension
6. Recalculer un score de cross validation 5 folds stratifiée multithread sur le train dataset et sortez les 5 AUC correspondantes. Quel est le risque en cas de non utilisation d'une cross-validation non stratifiée ? Pour quel type de dataset ce risque peut-il se présenter ? Est-il possible d'utiliser l'ensemble du dataset (train et test) pour la cross-validation ou faut-il se limiter au train dataset ?
   Point théorique : StratifiedFold, cross-validation
7. Sauvegardez le modèle et observez ce qu'il contient.
8. Calculez cette métrique sur le test set uniquement

FEATURE ENGINEERING 2 - XGBOOST

1. Est-ce intéressant d'ajouter du one-hot encoding sur XGboost ?
   Point théorique : max_cat_to_onehot, max_cat_threshold
2. Est-il possible d'entrainer des modèles out-of-core sur XGBoost ?
   Point théorique : out-of-core
3. Est-ce nécessaire d'enlever les variables fortement corrélées entre elles dans le cas d'un XGBoost ? Dans le cas d'un Random Forest ?
   Point théorique : multicollinearity

FEATURE ENGINEERING - Dummy

Vous pouvez vous inspirer des fonctions de ce notebook

1. Sélectionnez les variables de type catégorie et enlevez toutes les variables qui ont plus de 99% de variables identiques.
2. Enlevez également toutes les variables qui ont moins de 6 valeurs uniques.
3. Construisez les dummy variables à partir des variables restantes et travaillez avec le dataset résultant

FEATURE ENGINEERING - Random Forest

1. Après une discussion avec le métier, vous comprenez que les variables Var4, Var5 et Var6 (INTEGER) correspondent en fait à des variables catégorielles de segment client, les variables Var7, Var12 et Var22 correspondent à des montants dépensés. Utilisez les variables segments et montants pour créer de nouvelles features en appliquant la règle suivante : pour chaque variable montant vm1 et pour chaque variable catégorie vc1, la nouvelle feature est v1' = sqrt(abs((v1)^2 - Moyenne(v1 pour les enregistrements de type c1) ^2))
   Combien de nouvelles variables ont été créées ?
2. Utilisez un algorithme de feature selection de votre choix (sans utiliser la target) pour garder uniquement les 10% de variables les plus pertinentes. Combien reste-il de variables nouvellement créées ?
3. Sauvegardez le dataset nouvellement créé
4. Calculez les scores de cross-validation et de test set en utilisant une random forest avec des paramètres par défaut et comparez avec les deux versions de XGBOOST
5. Est-il possible de calculer un score de validation après avoir fait une feature selection qui utilise la target ?

HYPERPARAMETRIZATION - Random Forest

1. Utilisez la grid search sklearn GridSearchCV (maximum 20 recherches) de pas constant pour chercher le meilleur set d'hyper-paramètre de Random Forest sur le dernier dataset utilisé en utilisant le test set comme validation. Quel est le meilleur set de paramètre dans votre grid search ?
   Quel est le problème computationnel rencontré si on ajoute des paramètres dans notre grid search ? Est-il possible de changer le dataset utilisé pour la validation de la grid search ? Est-il possible de changer la fonction de score de la grid search ?
2. Est-ce possible d'utiliser le test set comme dataset de validation de la grid search ? Dans quelles conditions cela peut être autorisé ?
3. Quels sont les inconvénients d'augmenter max_depth dans un Random Forest ?
4. Utilisez la librairie hyperopt (ou autre) pour rechercher le meilleur paramètre dans le même espace. Quelle est l'avantage d'une telle approche par rapport à une grid classique ?
   Point théorique : Gradient descent, Optimization Algorithm

PIPELINE

1. Repartez du chargement de la donnée brute (après le passage par le cast des dtypes) et, après avoir fait un train/test split, en utilisant les pipelines et les ColumnTransformer, faites une imputer de missing values, un one-hot encoding, un scaler de variables et terminez par un classifier (par exemple random forest). Produisez la ROC curve sur le dataset de test.
2. Optionnel : Ajoutez un ColumnTransformer dans votre pipeline en appliquant la normalisation L1 sur les variables Var7, Var12 et Var22.
3. Quels avantages y a-t-il à utiliser un pipeline au lieu de code plain ? Êtes-vous convaincu par l'usage des pipelines ?

MISC

1. L'équipe soupçonne qu'un ancien data scientist malintentionné a inversé certains des target de -1 vers 1 et de 1 vers -1 pour se venger de son manager. En utilisant une approche de type cross-val-predict (5 folds par exemple), identifiez les candidats possibles à l'inversion du plus probable au moins probable avec la méthode predict_proba de l'estimateur du cross-val-predict. Trouvez les 3 labels upselling les plus suspects.

2. Utilisez TPOT pour trouver le meilleur pipeline de machine learning adapté à notre dataset de train. Calculez les métriques et comparez les par rapport à xgboost et random forest
   Point théorique : Automated Machine Learning
3. Quelles différences fondamentales y a-t-il entre la démarche artisanale et TPOT ?