MQT-6021-A21 – Travail pratique 2

Michael Morin

8 octobre 2021

Instructions

- L'équipe doit être enregistrée dans le système de gestion des équipes.
- Les communications sur le sujet du travail ne sont permises qu'à l'intérieur d'une même équipe.
- Tous les fichiers nécessaires pour reproduire votre analyse doivent être déposés dans la boîte de dépôt (voir la section Livrables).
- Le travail est noté sur 100 points. La contribution du travail à votre note finale est telle qu'indiquée dans le plan de cours.
- Le plagiat est sanctionné par la note 0 et les mesures appropriées.
- N'oubliez pas de citer vos sources (s'il y a lieu) et d'accorder une importance particulière à la qualité de la langue et à la lisibilité.

Livrables

Ce travail pratique est constitué de deux livrables principaux :

- un rapport détaillé au format HTML, celui-ci devra être créé en utilisant R Markdown (voir les détails ci-dessous);
- l'ensemble des fichiers utilisés pour recréer votre analyse y compris les fichiers R Markdown.

Vous êtes notés à la fois sur les explications et sur le code. Vous devez appliquer de bonnes pratiques de programmation (code R le plus propre possible). Vous devez appliquer les bonnes pratiques pour l'analyse (même si ces pratiques ne sont pas toujours mentionnées sur l'énoncé). Par exemple:

- afficher un exemple de données dans le rapport;
- vérifier les valeurs manquantes après le chargement de données;
- vérifier les valeurs manquantes suite aux opérations de jointures de tables;
- revérifier les valeurs manquantes après leur traitement...

Rapport détaillé

Le travail pratique a plusieurs questions et chaque question pourrait avoir plusieurs sous-questions. Les questions sont indépendantes dans le sens où elles portent sur des données différentes.

Vous devez utiliser un R Markdown par question de façon à produire plusieurs petits rapports (un par question) au format HTML. Notez que toutes les sous-questions se rapportant à une même (grande) question devraient être dans le même R Markdown et dans le même fichier HTML une fois le R Markdown compilé.

Chaque fichier HTML qui correspond au rapport devra avoir un entête contenant:

- le titre contenant le numéro de votre équipe, le numéro du travail pratique et le numéro de la question;
- le nom de chacun des membres de votre équipe et leur numéro matricule entre parenthèses;
- la date.

Cet entête doit être généré via l'entête YAML du format R Markdown.

Le rapport, pour chaque (grande) question, doit contenir une section par sous-question. Pour chaque question et sous-question, détaillez et justifiez la réponse proposée par votre équipe. L'absence de justification pour une sous-question entraînera des pénalités sur la note finale du travail.

L'utilisation de R Markdown est évaluée dans ce travail de sorte que des pénalités sont attribuées pour la non-utilisation ou la mauvaise utilisation de R Markdown.

Fichiers R Markdown et autres fichiers

Vous devez fournir tous les fichiers nécessaires pour permettre de reproduire votre analyse pour chacune des questions et des sous-questions.

Vous serez pénalisés pour les fichiers manquants puisque la note de 0 sera attribuée à une sous-question pour laquelle nous ne sommes pas en mesure de reproduire votre analyse. Ceci peut arriver pour différentes raisons, les plus fréquentes sont:

- il manque un ou plusieurs fichiers de données ou de code;
- il y a une erreur dans le code qui crée un bogue.

S'il y a un aspect aléatoire à votre analyse (peu importe la raison), SVP le mentionner. Le cas échéant, nous en tiendrons compte lors de nos tentatives de reproduction de votre analyse.

Notez que les fichiers de données que nous avons distribués pourraient être remplacés par les originaux pour tester votre code. Il ne faut pas changer le nom de ces fichiers lorsque vous les utilisez dans votre code.

Nous tenterons d'exécuter le code du R Markdown dans la séquence qu'il apparaît dans RStudio et aussi de compiler le R Markdown en HTML. Le code doit fonctionner et le R Markdown doit compiler dans cette séquence. Testez sur un environnement R vide.

Veuillez suivre la convention "EQn_TPz_Qx_nomFichier.ext" pour nommer vos fichiers où:

- EQ est la chaîne de caractères EQ suivie du numéro de votre équipe n;
- TP est la chaîne de caractères TP suivie du numéro de TP z;
- Q est le caractère Q suivi du numéro de question x;
- nomFichier peut être remplacé par un nom qui décrit sommairement le contenu du fichier, par exemple, le numéro de sous-question si c'est un fichier relié à une sous-question;
- ext est l'extension du fichier (p. ex., Rmd pour un R Markdown, csv pour un fichier CSV).

Voici un exemple de fichier nommé correctement pour le R Markdown de l'équipe 3 pour la question 2 du TP 12: "EQ3_TP12_Q2_analyse.Rmd". Pour la sous-question 1 de cette même question, l'équipe a produit un fichier de données sur les courges au format CSV. Elle l'a nommé correctement: "EQ3_TP12_Q2_1dataCourges.csv".

Des points seront retirés si la convention n'est pas respectée ou s'il y a confusion dans les fichiers.

Sur l'utilisation de R

Attention! Si vous travaillez avec la commande *setwd* dans un Rmd, c'est le moment de vous en départir. Pour que le répertoire de travail soit le répertoire courant du fichier Rmd, vous pouvez créer un fichier Rproj par question. Ensuite, ouvrez d'abord le Rproj, puis le Rmd. Ceci facilitera l'importation de données à partir d'un chemin relatif, c'est-à-dire, un chemin à partir du répertoire courant. La commande *setwd* n'a pas besoin d'être utilisée et provoque généralement des erreurs étranges.

Si votre programme génère des fichiers, il ne doit pas écraser les fichiers distribués pour une question donnée.

De plus, les fichiers de données fournis (le cas échéant) ne doivent pas être modifiés directement (pas de tri de données dans Excel, pas de changement manuel ou à l'aide de code au contenu des fichiers du sous-répertoire *data*). Ceci est vrai pour tous les travaux dans le cadre de ce cours: il convient d'éviter les traitements manuels. Si des modifications doivent être faites aux données, il faut les faire avec notre outil: R. Celui-ci permet d'automatiser le traitement de données.

Archive et structure du répertoire

Pour remettre le travail, vous devrez créer une archive zip (dossier compressé).

Placez les fichiers de chaque (grande) question i dans leur propre répertoire (aussi nommé dossier) nommé Qi. Placez tous les répertoires contenant les questions dans un répertoire nommé "EQn_TPz" où n est le numéro de votre équipe et z est le numéro du TP. Placez vos HTML compilés à la racine du répertoire. Faites le ménage des fichiers qui ne sont pas nécessaires (les retirer). La Figure 1 présente un exemple de répertoire bien structuré et de l'archive résultant de la compression de ce répertoire. Le nom du répertoire parent est

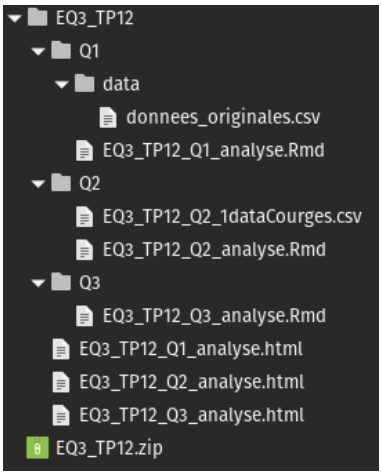

Figure 1: Exemple de répertoire bien structuré et de l'archive résultant de la compression

EQ3_TP12. Dans cet exemple, le travail pratique 12 avait 3 questions. L'équipe 3 a donc fait un répertoire par question: Q1, Q2, Q3. Dans chacun de ces répertoires, un fichier R Markdown a été créé. On remarque que chaque R Markdown a un HTML correspondant qui a été compilé par l'équipe et placé à la racine du répertoire EQ3_TP12. Chose intéressante, on remarque que des données étaient distribuées avec la question 1. Celles-ci sont disponibles dans l'archive. On remarque aussi que, pour la question 2.1, l'équipe a généré un fichier CSV nommé "EQ3_TP12_Q2_1dataCourges.csv".

Les fichiers HTML, c'est ce que nous lirons pour comprendre votre analyse. Les fichiers R Markdown, c'est ce que nous exécuterons puis compilerons en document HTML pour savoir si votre code fonctionne.

Je vous conseille de bien structurer vos répertoires dès le départ et de faire le ménage à la fin du travail pratique avant de créer l'archive. Pour faire le ménage et faire l'archive, travaillez sur une copie du répertoire (ainsi vous pourrez revenir en arrière si vous effacez quelque chose de trop). Ensuite, créez votre archive. Pour vous assurer que tout fonctionne, décompressez votre archive dans un répertoire temporaire et retestez vos fichiers R Markdown et vos scripts à partir d'un environnement R vide.

Note finale

Évitez d'utiliser des caractères spéciaux pour nommer vos fichiers et vos répertoires. Utilisez des lettres sans accents, des chiffres et le caractère _ (appelé underscore ou caractère de soulignement) pour remplacer les espaces.

Prenez bien le temps de lire l'énoncé. Notez que certaines questions sont plus courtes que d'autres.

Pour terminer, j'offre un remerciement spécial aux auxiliaires de MQT-6021 pour leur contribution aux questions de ce TP!

Bon travail!

Michael Morin

1 Estimation de l'efficacité d'une opération de recherche en mer (50 points)

Mise en situation (version réaliste): Vous êtes une équipe d'analystes en quête de richesses et vous venez de joindre un groupe de flibustiers à la recherche d'un bateau fantôme qui apparaît la nuit au large. Ce bateau est imprévisible dans sa position et sa dérive en plus d'être difficile à détecter visuellement (pour la recherche, le capitaine utilise un drone télécommandé longue distance avec vue à la première personne et vision de nuit).

La légende raconte que certains soirs les coffres du bateau sont remplis d'or. L'équipage a déjà cherché le bateau à maintes reprises. Ils l'ont même parfois trouvé, mais ses coffres étaient vident. Au total, ces 15 dernières années, ils ont complété un peu plus de 3500 recherches différentes... mais le capitaine ne se décourage pas!

En fait, il a dernièrement entendu parler de l'analytique prédictive. Il a aussitôt contacté votre équipe. Heureusement, l'ordinateur de bord a conservé les informations sommaires de chaque recherche sous forme de fichier CSV (voir `Q1_data.txt`). Ce fichier contient les caractéristiques (colonnes préfixées de `in`) et la probabilité de succès de l'opération (colonne `outcome`) pour toutes les recherches effectuées: même pour celles où ils n'ont rien trouvé l'ordinateur estime une probabilité de succès.

Vous décidez qu'il est temps d'exploiter ces données... L'ordinateur de bord ne peut pas évaluer la probabilité de succès des recherches qui n'ont jamais été complétées, mais un modèle de régression le pourrait! Vous décidez donc de créer un modèle de régression à partir des données connues qui devrait vous permettre d'estimer la probabilité de succès d'une recherche avec de nouvelles caractéristiques dans l'espoir de permettre au capitaine d'évaluer plusieurs façons de chercher le bateau sans devoir naviguer réellement.

Mise en situation (version farfelue): Vous travaillez comme analyste et offrez vos services à un groupe responsable de recherche et sauvetage maritime. ...

Pour cette question, vous aurez besoin du `tidyverse` (voir (Wickham and Grolemund 2017)) et de `caret` (voir (Kuhn 2019) et (Kuhn and Johnson 2013)).

```
library(tidyverse)
library(caret)
```

Tous les modèles doivent être entraînés à l'aide de `caret`. Vous pouvez charger les autres packages requis au besoin.

1.1 Préparer les données

Chargez les données et préparez-les pour l'apprentissage d'un modèle de régression: par exemple, vous pouvez partitionner vos données en ensemble d'entraînement et de test.

Vous pouvez aussi retirer certaines colonnes des entrées. Rappelez-vous que d'autres opérations de prétraitement peuvent être faites lors de l'appel à la fonction `train`.

Décrivez sommairement les étapes de votre démarche.

N'oubliez pas les bonnes pratiques:

- Vérifiez les valeurs manquantes;
- Vérifiez quelques lignes du jeu de données;
- Vérifiez si les types de colonnes sont corrects.

1.2 Entraînement

Entraîner deux modèles pour la prédiction de la probabilité de succès d'une recherche. Vous devez absolument entraîner un *k-nearest neighbors* (k plus proche voisin). Pour ce modèle, fixez le paramètre `method` à `knn`, c.-à-d., `method = 'knn'`, lors de l'appel à la méthode train.

Pour le second modèle, vous pouvez utiliser un algorithme d'apprentissage tant qu'il est disponible dans `caret`.

Nous en demandons au moins deux, mais vous êtes libre de tester autant d'algorithmes d'apprentissage que vous le souhaitez.

Vous devez créer un objet avec `trainControl` pour vous permettre de spécifier les paramètres de la phase d'entraînement. Créez aussi, pour chaque modèle à entraîner, une grille de tuples pour tester plusieurs paramètres de l'algorithme d'apprentissage.

Après l'entraînement, vous devriez sauvegarder chaque modèle au format RDS. Si le fichier RDS correspondant à un modèle donné est présent, l'entraînement ne devrait pas se réexécuter et le modèle devrait être directement chargé à partir du RDS. L'entraînement ne devrait être exécuté que si le fichier RDS est absent.

Décrivez sommairement les étapes de votre démarche et les méthodes utilisées.

1.3 Évaluation de l'entraînement pour chaque modèle individuel

Affichez les meilleurs paramètres trouvés via l'entraînement pour chaque modèle.

Vérifiez la performance des modèles, pour au moins un des différents paramètres utilisés lors de l'entraînement, à l'aide de visualisations. La performance peut être visualisée en termes du critère de votre choix.

Affichez les résidus pour chaque modèle. Selon le modèle sélectionné, vous devrez les recalculer à partir des prévisions effectuées sur l'ensemble utilisé pour l'entraînement.

Décrivez sommairement les étapes de votre démarche.

1.4 Comparaison des deux modèles

Comparez vos modèles et stockez le meilleur (selon le critère de votre choix) dans un fichier RDS.

Le critère pour la comparaison de modèles est au choix, toutefois, vous devez choisir un critère bien fondé et qui a un sens en régression.

Produisez un graphique montrant la valeur observée par rapport à la valeur prédite pour chaque modèle.

Décrivez brièvement les étapes de votre démarche. Indiquez clairement quel est le meilleur modèle.

1.5 Compétition... en production (bonus)

Une compétition est lancée. Nous vous avons fourni un jeu de données complet pour l'entraînement du meilleur modèle possible.

Pour la compétition (qui est un bonus, donc facultative), nous testerons votre meilleur modèle sur un jeu de données qui vous est inconnu: le jeu de données de production. Ce jeu de données est similaire au jeu de données que nous vous avons fourni, mais il contient plus de 32000 observations.

Les modèles entrant dans la compétition seront testés sur le jeu de données de production. La compétition vise à déterminer, parmi tous les modèles en compétition, ceux qui sont les plus prometteurs.

Notre critère d'évaluation est le RMSE.

Il y a trois requis pour entrer dans la compétition (voir les sous-sections plus bas). S'il manque l'un des requis, votre modèle est exclu de la compétition. Si votre code ne fonctionne pas sur le jeu de production sur nos ordinateurs, vous serez exclus de la compétition.

Pour vous assurer que votre code est fonctionnel pour la compétition, nous vous suggérons de le tester sur un ordinateur différent de celui utilisé pour le développer de même que sur un environnement R vide. Finalement, votre code ne doit en aucun cas changer la colonne de sortie ou utiliser la colonne de sortie pour la prévision sinon le modèle sera exclu de la compétition. Si vous manipulez les données au niveau des colonnes pour

votre entraînement (disons, si vous retirez une colonne), vous devez traiter les mêmes colonnes avec votre code de prétraitement des données de production.

Parmi, tous les modèles qui entrent dans la compétition (respectant les trois requis), nous choisirons les 5 meilleurs. Les créateurs de ces modèles auront un bonus de 15 points sur 100 sur la note du TP2. Tout compétiteur a un bonus de 5 points sur 100 sur la note du TP2. Si vous respectez les requis 1 et 2, mais pas le troisième, vous aurez un bonus de 2 points sur 100 sur la note de votre TP2.

Vous ne pouvez pas avoir plus de 100% dans la note du TP. Votre note au TP, Z, est calculée ainsi: $Z = min\{y + b, 100\}$ où y est votre note originale sans bonus et b est le bonus à la Q1.

1.5.i Premier requis

Vous devez nous fournir votre meilleur modèle au format RDS. Vous devez nommer ce fichier `Q1_challenger_Tx_nomModele.rds` où:

- `x` est le numéro de votre équipe;
- `nomModele` est le nom de votre modèle.

Voici un exemple de nom de fichier correct: `Q1_challenger_T6021_Houdini.rds`. Le nom du modèle est `Houdini`. C'est le modèle de l'équipe numéro 6021.

1.5.ii Deuxième requis

Attention! Pour entrer dans la compétition, vous devez aussi fournir du code pour permettre la mise en production de votre modèle en plus du fichier RDS mentionné ci-dessus.

Le code pour la mise en production doit permettre de:

- charger votre modèle à partir du fichier RDS;
- charger un jeu de données dans le même format que `Q1_data.txt` et le traiter pour qu'il soit compatible avec votre modèle;
- rouler votre modèle, déjà entraîné, sur le jeu de données chargé, mais qui contient des lignes différentes.

Voici un code qui permet de charger le modèle de l'équpe 6021:

```
challenger_model_path <- 'Q1_challenger_T6021_Houdini.rds'
if(file.exists(challenger_model_path)) {
  challenger_model <- read_rds(challenger_model_path)
}
```

1.5.iii Troisième requis

Pour entrer dans la compétition, un modèle devra battre le modèle `Houdini` de l'équipe `6021` sur les données de production. Le RMSE d'`Houdini` est de 0.0280841 sur le jeu de production. Vous n'avez pas accès au jeu de production, mais en appliquant de bonnes pratiques pour l'apprentissage automatique et en testant un bon nombre d'agorithmes d'apprentissage et leurs paramètres: vous pouvez battre ce modèle.

2 Commercialisation et prévisibilité du succès de produits (50 points)

La commercialisation de produits alimentaires est un problème complexe. Même si une part du succès initial d'un produit est lié au marketing, un mauvais produit ne passe normalement pas le test du temps.

Vous faites partie d'une entreprise qui offre ses services pour l'évaluation et la prévision du succès de produits alimentaires étant donné leurs caractéristiques. Votre entreprise ne s'intéresse pas à l'aspect marketing. Elle

met plutôt l'accent sur les caractéristiques du produit par rapport aux produits du même genre. Votre dernier gros contrat est dans le domaine vinicole.

Contrat: Raisin Cola (R Cola), une compagnie de boissons gazeuses, souhaite se lancer dans la production de vin après avoir acheté plusieurs producteurs. R Cola n'y connaît rien en vins, mais elle aimerait développer une nouvelle gamme de produits et prévoir leurs succès à long terme.

Données: Vous avez accès à des données historiques sur le succès et l'échec à long terme de la commercialisation de plusieurs vins (fichier `Q2_data.csv`). Dans les données, on vous indique si oui ou non un vin a survécu sur le marché durant plus de 5 ans (colonne `Class`). Les autres colonnes (précédées de `in`) sont les caractéristiques des vins suivantes (dans l'ordre):

- couleur
- tanin
- densité optique de 420 nm
- densité optique de 520 nm
- densité optique de 620 nm
- acidité de type fixe
- acidité de type volatile
- acidité de type citrique
- sucre
- chlorures
- dioxyde de soufre libre
- densité de dioxyde de soufre totale
- densité
- pH
- sulfates
- teneur en alcool
- prix moyen ajusté en dollars canadien

2.1 Préparer les données

Chargez les données et préparez-les pour l'apprentissage d'un modèle de classification:

- Vous devez partitionner le jeu de données en ensemble d'entraînement et de tests.

Vous pouvez aussi retirer certaines colonnes des entrées. Rappelez-vous que d'autres opérations de prétraitement peuvent être faites lors de l'appel à la fonction `train`.

Décrivez sommairement les étapes de votre démarche.

N'oubliez pas les bonnes pratiques:

- Vérifiez les valeurs manquantes;
- Vérifiez quelques lignes du jeu de données;
- Vérifiez si les types de colonnes sont corrects.

2.2 Entraînement

Entraînez deux modèles pour la prédiction du succès des vins. Vous devez absolument entraîner un arbre de décision (*decision tree*). Pour ce modèle, fixez le paramètre `method` à `rpart` ou `rpart2`, p. ex., `method = 'rpart2'`, lors de l'appel à la méthode train. L'autre modèle est au choix, mais il doit être différent d'un arbre de décision.

Nous en demandons au moins deux, mais vous êtes libre de tester autant d'algorithmes d'apprentissage que vous le souhaitez.

Vous devez créer un objet avec `trainControl` pour vous permettre de spécifier les paramètres de la phase d'entraînement. Créez aussi, pour chaque modèle à entraîner, une grille de tuples pour tester plusieurs paramètres de l'algorithme d'apprentissage.

Après l'entraînement, vous devriez sauvegarder chaque modèle au format RDS. Si le fichier RDS correspondant à un modèle donné est présent, l'entraînement ne devrait pas se réexécuter et le modèle devrait être directement chargé à partir du RDS. L'entraînement ne devrait être exécuté que si le fichier RDS est absent.

Décrivez sommairement les étapes de votre démarche et les méthodes utilisées.

2.3 Afficher l'arbre de décision appris

Vous avez entraîné un arbre de décision. Utilisez la librairie `rpart.plot` pour afficher l'arbre appris (fonction `prp`).

2.4 Évaluation de l'entraînement pour chacun des modèles

Affichez les meilleurs paramètres trouvés via l'entraînement pour chaque modèle.

Vérifiez la performance des modèles, pour au moins un des différents paramètres utilisés lors de l'entraînement, à l'aide de visualisations. La performance peut être visualisée en termes du critère de votre choix.

Décrivez sommairement les étapes de votre démarche.

2.5 Comparaison des modèles

Comparez vos modèles.

Le critère pour la comparaison de modèles est au choix, toutefois, vous devez choisir un critère bien fondé et qui a un sens en classification.

Produisez, pour chaque modèle, une courbe ROC et affichez l'aire sous la courbe ROC. Je vous conseille le package `pROC` pour calculer les valeurs de courbes ROC à partir des probabilités d'appartenance aux classes. Ensuite, je conseille d'utiliser `ggplot` pour tracer vos courbes.

Décrivez brièvement les étapes de votre démarche. Indiquez clairement quel est le meilleur modèle.

Références

Kuhn, Max. 2019. "The Caret Package." 2019. https://topepo.github.io/caret/index.html.

Kuhn, Max, and Kjell Johnson. 2013. *Applied Predictive Modeling*. Vol. 26. Springer. https://link.springer.com/book/10.1007%2F978-1-4614-6849-3.

Wickham, Hadley, and Garrett Grolemund. 2017. *R for Data Science: Import, Tidy, Transform, Visualize, and Model Data*. " O'Reilly Media, Inc.".