Avant toute chose, un peu de préparation de données…

Bonjour à toutes et à tous, la semaine dernière nous vous avions conseillé sur quel type de modèle de machine learning choisir pour votre iRoboat mais cette semaine nous vous proposons de vous donner des pistes sur comment bien préparer la donnée brute qui vous a été fournie dans le kit n°2 avant de la fournir à votre super IA.

Comme vous vous doutez surement, il n'est **pas optimal** de donner de la donnée brute, non-travaillée, à une IA, vous devez d'abord effectuer quelques étapes dites de **pre-processing**. Il n'y a pas une règle d'or qui permette d'avoir la donnée optimale quelque soit le type d'IA utilisé (ça serait beaucoup trop simple, n'est-ce pas ?) mais il existe quelques « règles » / idées qui pourront vous guider. Nous allons tenter de vous donner nos best tips pour que vous puissiez y parvenir.

Normalement, vous avez déjà accès au code iRoboat Accenture présent ici. Nous allons donner quelques exemples présents sur le notebook de data pre-processing que vous trouverez ici qui donnent quelques pistes de comment préparer vos données. Ce code ne représente pas la meilleure manière de faire, nous vous recommandons de vous **approprier le code**, de tester d'autres manières de faire, etc.

Création de nouvelles variables, suppression des doublons, etc.

Une première bonne pratique est de **créer de nouvelles variables** plus parlantes, plus proches des KPIs (valeurs importantes pour notre cas d'usage) que nous voulons étudier. Quelques exemples présents dans le notebook : *angleFromCoordinate* (convertit la longitude et latitude de deux points en angle en degré), *compute_target_angle* (calcule l'angle en degré à partir d'une liste de longitude et latitude suivant un pas de temps), *vr_trigo_bijection* (renvoie la bijection d'un angle en degré).

Il est important de vérifier que vous n'avez **pas de doublons** dans vos données et si c'est le cas de les retirer (voir cellules 15 et 16 du notebook).

On peut également regarder la **corrélation** entre les variables (voir cellules 11 et les deux suivantes) car certains modèles n'aiment pas avoir des variables trop corrélées. L'idéal est d'avoir des variables le moins possibles corrélées de sorte à représenter le plus d'informations possible sans répétition.

Une autre bonne pratique est de **représenter ses données avec des graphiques** (voir fin du notebook). Cela permet de repérer de potentielles erreurs mais également de mieux comprendre ce que la donnée a à nous dire.

Que signifie train, validation, test set ?

Dans le vocabulaire du machine Learning, il est très courant d'entendre les termes train, validation et test set. En effet, ces notions renvoient à des partitions distinctes et hétérogènes d'un jeu de données initial pour effectuer des tâches bien précises dans une activité d'apprentissage automatique.

Le **training set** est la portion de données utilisée pendant l'étape de **training,** cette étape consiste à entrainer un modèle ayant des entrées et une sortie, on parle en général d'inputs X et de labels y. L'action de training peut se résumer mathématiquement en une opération y = f(X) effectuée en plusieurs itérations, où la fonction f est un modèle d'entrainement ou une application disposant d'une matrice de passage qui met à jour ses paramètres ou ses coefficients à chaque itération d'entrainement dans le but d'adapter son calcul d'output label en fonction des nouveaux inputs du training set qui lui sont présentés.

Le **validation set** est utilisé pendant l'étape de validation, cette partition est aussi utilisée pendant le training, cela consiste à ajuster les paramètres du modèle en cours d'entrainement en testant le modèle sur les échantillons du validation set. Notons que la notion de validation set n'est pas nécessaire dans les tâches de prédictions peu complexes, il est généralement utilisé quand il s'agit d'effectuer des tâches d'entrainements complexes sur un modèle à base de réseaux de neurones. Ce validation set est aussi utilisé pour stopper le processus d'itérations d'entrainement quand on se rend compte que les performances d'entrainement sur le validation set commencent à se dégrader.

Enfin le **test set** est une partition qui consiste à confronter une portion du jeu de données initial non utilisé pendant le training/validation au modèle entrainé, l'idée est de voir comment le modèle entrainé réagi en termes de calcul d'output et de performance face à de nouveaux inputs. Cette partition de données est utilisée pour calculer les métriques d'évaluation du modèle entrainé.

Comment diviser mes données afin de pouvoir tester correctement mon IA ?

Dans le cadre de l'apprentissage supervisé (voir tips of the week n°2), il est utile de pouvoir tester la performance de son modèle IA. Pour ce faire, il ne faut pas réutiliser les données qui ont déjà servi à entrainer le modèle car les résultats de notre test seraient trop optimistes et donc biaisés. Ainsi, il est recommandé de couper notre data set en deux : un training set et un test set (ou en trois avec le validation set). En général, on dit que le training set représente 70% et le test set 30% du data set total (ou 80/20% si on a beaucoup de données) mais ces chiffres restent à titre indicatif (70:20:10% pour des training:validation:test sets). L'idée est d'avoir un maximum de données à entrainer tout en gardant un nombre suffisant de données pour pouvoir tester. Il est aussi important de **stratifier** les données au moment du découpage, c'est-à-dire de conserver la même proportion de chacune des classes du data set total.

Il est également possible d'entrainer un même modèle plusieurs fois en choisissant un couple training/test sets (ou training/validation sets) différent à chaque fois afin d'éviter un biais lié au choix découpage des données, cela s'appelle la cross-validation. Différentes techniques existent, une des plus commune est le **k-fold cross validation**. Cela consiste à découper le data set en deux (training et test sets ou training et validation sets) de k manière différentes en veillant à ce que chaque donnée ne soit qu'une fois et au moins une fois dans le test set. Ici aussi, il faut penser à stratifier le découpage. Pour avoir le taux de réussite par exemple (ou n'importe quelle autre métrique), il suffit de faire la moyenne des k modèles. Vous trouverez plus d'informations dans ce lien :
https://machinelearningmastery.com/k-fold-cross-validation