# Cours IFT6266,
# Exemple d'application: Data-Mining

Voici un exemple du processus d'application des algorithmes d'apprentissage statistique dans un contexte d'affaire, qu' on appelle aussi "data-mining".

1. **Définition du problème**. On discute avec le client pour identifier les buts recherchés et les convertir en un problème d'apprentissage statistique bien défini. Il faut comprendre le contexte d'affaire, les contraintes d'affaire, les contraintes technologiques, et avoir une idée

   (a) des données historiques disponibles (ce qui a été mesuré dans le passé),

   (b) des données (variables) qui seront disponibles au moment de prendre une décision sur un nouveau cas avec le modèle entraîné (SEULES CES VARIABLES PEUVENT SERVIR COMME INTRANT),

   (c) des variables qui peuvent servir comme variable dépendente (à prévoir), possiblement à posteriori (i.e. avec un décalage temporel),

   (d) des critères par lesquels on pourra évaluer la qualité (au moins de manière comparative) des solutions; idéalement on devrait entraîner les modèles en fonction de ces critères, ou quelque chose d'assez proche),

   (e) des solutions déjà en place ou déjà disponibles (on voudra s'y comparer).

2. **Extraction des données brutes**. Les données sont généralement stockées dans des bases de données relationnelles, avec de multiples tables et des clés (champs) spéciales permettant de les croiser (quelle rangée de la première table correspond à quelle(s) rangée(s) de la deuxième). Il s'agit d'écrire les scripts permettant d'extraire les données voulues. Les clients ont tendance à proposer d'utiliser seulement un petit échantillon (c'est l'habitude, surtout à cause des logiciels inefficaces comme SAS généralement utilisés dans le commerce) mais il vaut mieux essayer d'obtenir le maximum de cas. On va mettre les données dans un format utilisable pour l'apprentissage statistique (fichiers lisibles par vos programmes). On peut définir des interfaces efficaces (classes virtuelles dans un langage orienté objet) pour que les données soit visibles comme une grande table avec une rangée par exemple, même si elles ne sont pas stockées comme ça sur disque. Certaines opérations sur les données (prétraitement, encodage, croisage, etc.) peuvent donc se faire à la volée (au moment où l'exemple est lu), ce qui dans certains cas peut mener à des économies de temps et de mémoire (car des représentations très gourmandes en espace sur disque coûteraient très cher en temps d'accès disque).

3. **Analyse des données**. Il s'agit d'améliorer notre compréhension des données et de leur sémantique, afin de mettre au point un modèle le plus adapté possible. On procède d'abord avec des outils statistiques uni et bi-variés et des outils de visualisation:

   (a) Calculer et afficher les statistiques de base de chaque variable prise séparément. Si c'est une variable discrète quelles sont les valeurs prises? quel est le sens

de ces valeurs? est-ce que les valeurs sont ordonnées (peut-on les coder par un seul nombre ou faut-il un vecteur d'indicatrices binaires?) Est-ce que l'ensemble de toutes les valeurs discrètes possibles de la variable (celles qu'on pourrait retrouver en test) est connu d'avance? Sinon il faudra peut-être prévoir une valeur symbolique spéciale pour les valeurs jamais vues en aprentissage? On pourrait aussi vouloir regrouper les valeurs très rares dans l'ensemble d'apprentissage dans cette même valeur symbolique spéciale. Si c'est une variable continue, a-t-elle des queues épaisses? (on peut mesurer un indice d'épaisseur de queue, mais la visualisation donne déjà une assez bonne indication). Afficher des histogrammes des distributions pour chaque variable. Identifier les patrons de "données manquantes" (si il y a lieu).

(b) Calculer et afficher des statistiques bi-variées entre la ou les variables dépendentes et chaque variable indépendente. Effectuer des tests de dépendence (linéaire, monotone, non-linéaire générale) entre chaque variable indépendente et la variable dépendente. Ceci permet d'identifier des variables suspectes (classées dans les indépendantes mais qui est en fait dérivée de la variable dépendente et ne sera pas disponible au moment des prédictions). On pourrait vouloir éliminer les variables qui ne semblent avoir aucune dépendence avec la variable dépendente (mais il se peut qu'une telle variable devienne utile dans le contexte d'autres variables d'entrée).

(c) Calculer et afficher des statistiques jointes entre chaque variable et la variable "temps" pour voir si les distributions de ces variables changent de manière significative avec le temps, la saison, le jour de la semaine, etc. Visualiser l'évolution des espérances, variances, et histogramme en fonction du temps pour chaque variable. Essayer de comprendre la raison de ces non-stationarités quand elles semblent vraiment évidentes (ça peut être des bogues dans la base de données, par exemple, ou des changements d'interprétations des variables).

(d) Identifier chaque variable indépendente (intrant) comme était soit "ordonnée" ou "symbolique". Les variables continues sont naturellement ordonnées. Les variables discrètes mais dont les valeurs sont ordonées (par exemple si la valeur est 6 on s'attend à ce que l'effet soit proche de si la valeur est 7, et pour la plupart des valeurs $i$, on s'attend à ce la variation d'effet ait le même signe quand on passe de $i$ à $i+1$ ou de $i+1$ à $i+2$ (monotonicité locale). Les autres variables discrètes sont "symboliques". Cette catégorisation des variables sera importante pour l'encodage des intrants de l'algorithme d'apprentissage.

À cette étape il sera aussi bon de décider d'une méthode de partition des données pour la selection d'hyper-paramètres et la comparaison de modèles. Si le nombre d'exemples est suffisant la méthode train-valid-test est grandement préférable, à cause des étapes manuelles qui peuvent intervenir dans les prochaines étapes.

4. **Prétraitement des données**. Il s'agit d'effectuer des transformations des données pour les rendre plus utilisables par les algorithmes d'apprentissage statistique.

   (a) Éliminer les variables qui ne contiennent pas assez d'information (trop rarement

non-manquantes, ou sans variation suffisante, ou sans vraiment aucune dépendence apparente ou raison de croire qu'elle est prédictive de la variable à prédire).

(b) Transformer la variable à prédire (selon les contraintes de la tâche et du type de modèle visé, on est parfois limité à des transformations affines, e.g. si on veut calculer l'espérance) afin de l'amener à une distribution "raisonnable". Dans certains cas on pourrait vouloir discrétiser la variable dépendente afin de prédire dans quelle intervalle elle va se réaliser.

(c) Compresser les variables discrètes prenant un trop grand nombre de valeurs différentes, en assignant plusieurs valeurs à une catégorie. Par exemple on pourrait garder seulement les 3 premières lettres du code postal, ou réduire la date au mois de l'année, ou bien l'on pourrait utiliser la distribution jointe empirique de cette variable avec la variable dépendente pour regrouper les valeurs qui donnent des effets semblables sur la variable dépendente.

(d) Transformer les variables indépendentes continues de manière à éliminer les effets dûs simplement à leur distribution marginale. La technique la moins sophistiquée consiste à standardiser (mais attention à ce qu'il n'y ait pas de queue épaisse), mais on peut faire mieux en uniformisant (en faisant une transformation non-linéaire monotone bijective et non-paramétrique qui mène à une variable de loi approximativement uniforme (entre 0 et 1). Cela se fait simplement en remplaçant la valeur par son rang centile (normalisé entre 0 et 1), qui s'obtient en faisant un histogramme avec intervalles de masse égale entre eux.

(e) Désaisonaliser les variables qui présentent des effets saisonniers. Le principe est de soustraire (si l'effet est additif) la composante explicable par la saison (heure du jour, jour de la semaine, jour de congé particulier, mois de l'année, etc...). Si au moins une variable comporte un effet saisonnier, on ajoutera en variable explicative les mêmes indicateurs saisonniers (e.g., $1_{t=mardi}$).

(f) Soustraire (ou autrement enlever) la tendance non-stationnaire qui aurait été detectée à l'étape précédente dans certaines variables (par exemple à cause de l'augmentation générale des prix, etc...).

(g) Optionnellement appliquer un algorithme de réduction de dimensionalité linéaire ou non-linéaire sur les variables indépendentes transformées. Il faudra générer des données en dimension réduite pour une gamme de dimensions (e.g., 5, 15, 50, 150) car on ne sait pas d'avance ce qui fonctionnera le mieux dans le cadre de l'apprentissage supervisé. Si on essaie une méthode non-linéaire on incluera aussi une version linéaire. Faire attention à utiliser seulement les données d'apprentissage pour estimer la transformation de réduction de dimensionalité.

(h) Optionnellement on pourrait construire un **processus générateur** de variations sur les véritables données, qui nous permet de transformer de manière aléatoire un exemple original en un nouvel exemple tout aussi plausible. Par exemple, si notre entrée est une image, on peut faire des transformations géométriques (par des petites quantités aléatoires) telles que translation,

rotation, changement d'intensité, etc., qui ne sont pas supposées changer la valeur de la variable dépendente (e.g. la classe de l'objet qui est dans l'image). Dans certains cas on sait qu'il y a une forme de bruit sur certaines variables, et on peut s'en servir pour simuler de telles variations. Tout ceci nous donne accès à une quantité virtuellement infinie d'exemples, que l'on pourra idéalement générer à la volée (pour l'apprentissage on-line ou gradient stochastique), ou bien l'on fera un gros ensemble d'apprentissage (si on utilise une méthode d'optimsation "batch", qui a besoin de voir tous les mêmes exemples d'apprentissage ensemble et plusieurs fois).

5. **Apprentissage supervisé**. À cette étape on appliquera une gamme de différents algorithmes d'apprentissage statistique sur les données transformées. Il faut s'assurer d'inclure des algorithmes très simples qui serviront de point de comparaison, par exemple des prédicteurs paramétriques linéaires (régression logistique ou régression linéaire ou régression linéaire Bayesienne), et si le nombre de données le permet, des estimateurs non-paramétriques simples. La forme particulière de l'algorithme pourra aussi être ajustée en fonction des particularités du problème à résoudre. On utilise l'ensemble de validation pour comparer les valeurs d'hyper-paramètres, incluant si il y a lieu les hyper-paramètres utilisés dans le prétraitement des variables.

6. **Analyse qualitative des résultats**. On va d'abord vérifier que l'on obtient des résultats raisonnables avec tous les algorithmes. Si les résultats semblent aberrants on pourra ici détecter des problèmes avec le prétraitement. On va ensuite faire une analyse qualitative des erreurs. Est-ce que les cas de grande erreur ont des points en commun? on pourra parfois essayer de visualiser les cas donnant le plus d'erreur. Tout élément qui permettrait de prédire l'erreur indique qu'il manque quelque chose dans les intrants. Par exemple si l'erreur est prévisible en fonction du temps $t$ cela peut vouloir dire que l'étape de stationarisation n'a pas été faite correctement, et l'on pourrait vouloir ajouer $t$ comme intrant.

7. **Analyse quantitative des résultats**. On va effectuer des tests statistiques pour estimer la performance de généralisation de chaque algorithme et leur performance relative (deux à deux), afin de vérifier si certains algorithmes sont vraiment supérieurs à d'autres.

8. **Analyse économique des résultats et rapport**. On essaie d'extrapoler à partir de ces mesures ce que le meilleur modèle permettrait d'atteindre comme performance économique (par exemple les économies qu'il permettrait de réaliser), dans le but de pouvoir évaluer l'intérêt économique d'implanter une telle solution sur le terrain. On écrit un rapport qui documente la procédure expérimentale, les resultats des comparaisons statistique, et l'analyse économique.

9. **Implantation sur le terrain**. Il s'agit maintenant de monter le système informatique qui utilisera les algorithmes d'apprentissage sur le terrain. En général cela implique de faire une interface avec le système de bases de données de l'organisation utilisatrice, et de considérer des optimisations de l'algorithme pour faire face à des

contraintes possibles de temps de calcul (par exemple il faudra peut-être paralléliser l'algorithme). Il faut aussi mettre en place des mécanismes de vérification pour s'assurer que le prédicteur est utilisé dans des conditions qui restent valides (par exemple que toutes les variables continuent à suivre une loi marginale similaire à celle identifiée dans la phase de modélisation). On veut donc implanter un système de monitoring qui traque les utilisations du système, les valeurs des variables, et les erreurs commises.

10. **Évaluation en continu**. Une fois que le système est implanté sur le terrain on va vérifier que les performances promises sont approximativement réalisées, et on va déclencher des alertes quand cela ne se produit pas comme prévu. On pourra aussi vouloir effectuer un ré-entraînement sur une base régulière, et un re-examen des distributions des variables à une fréquence moins élevée. Le ré-entrainement devrait idéalement aussi inclure aussi l'optimisation des hyper-paramètres, qui devrait donc un processus automatisé. On peut choisir des plages de valeurs pour chaque hyper-paramètre, ainsi que des vérification que ce n'est pas une valeur extrême de la plage qui est sélectionnée (dans ce cas il faut agrandir ou déplacer cett plage).