Rapport de Stage

STAGE DE FIN D'ETUDE A SCALED RISK

Ménage Nam | Ingénieur Data Science | 10/09/2017

Remerciements

Je tiens à remercier vivement mes encadrants de stage Bertrand Tillay et Dorin Tulei, pour leur disponibilité, leurs conseils avisés et leur écoute.

Ils étaient assez cléments avec moi, sachant qu'ils étaient conscients que je sortais d'une formation de mathématique et donc pas d'informatique pure.

Mes remerciements vont aussi à toute l'équipe des 15 dev De Scaled Risk pour avoir su m'accueillir dans un environnement de travail très fun, stimulant, et de m'avoir vraiment donné la joie du travail avec leur humour(qui part souvent en cacahuètes, mais c'est comme ça que je l'aime) avec leur honnêteté, leurs repas de quinoa et choux kale en bon parisien.
Plus particulièrement je remercie Roman Wilhem, Nizar Hdadeche, Hadien Belhacene et Guillaume Turchini pour l'installation, la prise en main, et le test du logiciel Scaled Risk.
Je remercie Thierry Duchamps pour son aide dans mes démarches administratives et Axelle Paillet pour ses conseils sur l'utilisation du logiciel R.
Enfin je souhaite remercier Nizar Hdadeche d'avoir pris le temps de me montrer le fonctionnement de ses codes WEKA, pour le Machine Learning.

Plan de la présentation

o/ Désambiguition

OLAP : En informatique, et plus particulièrement dans le domaine des bases de données, le traitement analytique en ligne (anglais OnLine Analytical Processing, OLAP) est un type d'application informatique orienté vers l'analyse en temps réel d'informations dans le but d'obtenir des rapports de synthèse tels que ceux utilisés en analyse financière.

In Memory : Le fait d'effectuer les calculs et de charger les données dans la mémoire vive (RAM), par opposition à la mémoire statique

On read/on write : Se dit de l'agencement des opérations de Lecture et Transformation de la Donnée (voire ETL/ELT), On Read signifie que on lit la donnée qui correspond à un critère, et on n'effectue que des transformations sur cette partie de la donnée. Alors que On Write signifie qu'on effectue une transformation sur l'ensemble de la donnée, puis on extrait des lignes qui correspondent à un certain critère de requête

time stamp: Champ d'horodatage, pour assigner une date précise à une ligne de donnée

Golden copy: Version originale de la donnée, conservée souvent en mémoire comme un back up du modèle, cette version n'est pas censée être changée et est censée être la référence.

ETL/ELT: Extract, Transform, Load et Extract, Load, Transform. Ce sont les Opérations effectués sur la donnée, après le chargement de la donnée(Extract), on peut choisir de lire une partie de la donnée et transformer cette partie, ou transformer tout et extraire une partie de la donnée transformée.

Jexl, lucene : Langages de requêtes, c'est à dire de recherche dans une base de donnée, comparable à SQL

Clé(primaire) : Champ / paramètre d'une table, qui est la référence, et permet de différencier deux lignes de données. C'est la colonne qui permet d'identifier de manière unique un enregistrement dans une table.

Jointures floues : Jointures d'une table à l'autre, qui prend des valeurs de clé non-exactes, approximatives. Voir la partie D.2

I/ Présentation de l'entreprise Scaled Risk

A) L'entreprise Scaled Risk

Depuis 2012, Scaled Risk exploite, contribue et améliore significativement plusieurs composants d'HBase; Pionnier du Big Data et acteur de premier plan sur l'utilisation en entreprise d'Hadoop dans un contexte de besoin d'un système robuste, scalable, et tolérant aux pannes, la société Scaled Risk place le Big Data au cœur de l'innovation et de la transformation digitale de l'industrie financière.

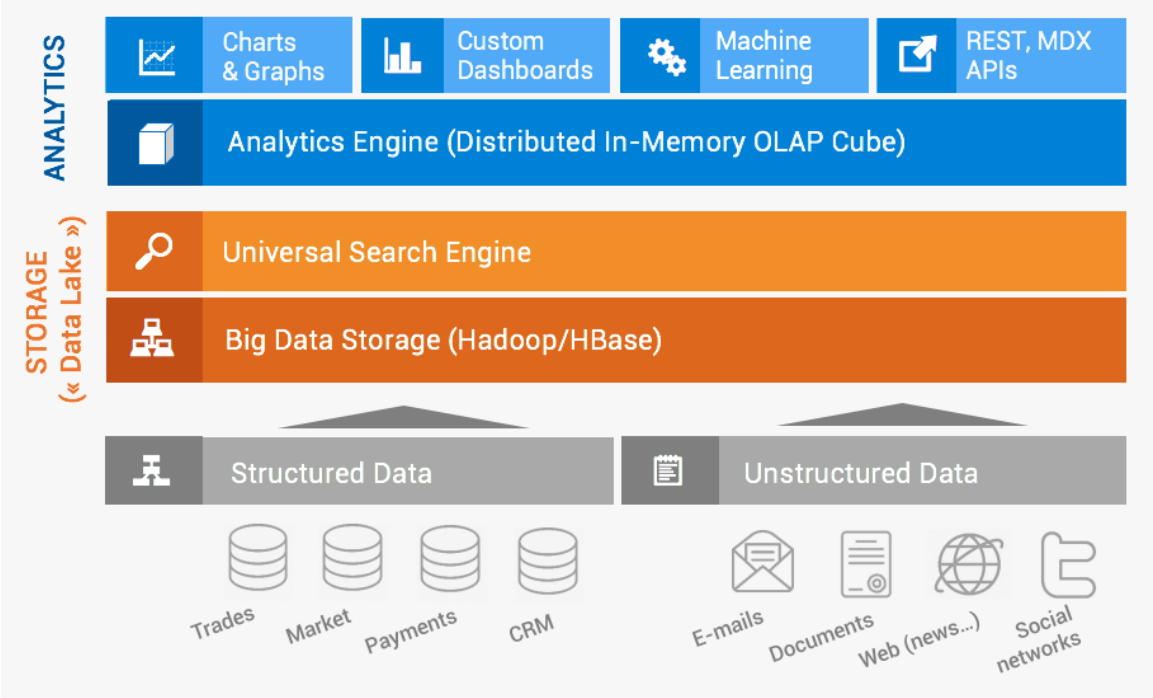
Thierry Duchamp (ESIA), est le fondateur et Directeur Général de Scaled Risk. Il a 15 ans d'expérience sur des positions managériales IT risque chez EFFIX, startup française du logiciel financier, puis chez Thomson Reuters (QA Director sur la suite logicielle Risk Management)

Hervé Bonazzi (Supélec), est le Président et CFO de l'entreprise. Il a 17 ans d'expérience en création et management de sociétés IT, fondateur d'une startup revendue à Orange, puis co-fondateur et DG de Finaxys (conseil IT-finance, 250 employés, 30M€ CA), élu entrepreneur Fintech de l'année 2016 par KPMG Luxembourg

B) Présentation de l'outil Scaled Risk

Scaled Risk est une plateforme de Data Monitoring & Management scalable, en temps réel. Le Data Lake Scaled Risk est dédié aux ingénieurs orientés business de part de son interface intuitive et sa visualisation claire de la donnée.

En effet, l'assimilation de la donnée se fait par un processus de Drag and drop, dans une GUI (Graphical User Interface) très explicite et bien organisée.



Voici quelques fonctionnalités du programme Scaled Risk, pour effectuer des opérations d'ETL classiques comme enregistrer et rechercher tout type de données, transactions ou documents.

Analyse	Stockage
Créer des rapports multi-source	Vue en temps réel des données
Alerte en temps réel	Tout volume de donnée
Interfaces de drag and drop	Sources internes ou publiques
Tableaux de bord Web personnalisables	S'adapte très facilement aux mises à jour du modèle
Export en PDF, CSV, Excel	Golden Copy

Des grandes quantités de tables peuvent être importées à partir des fichiers bruts, pour après être transformées, par deux principaux vecteurs : les Business Views, la syntaxe de requête Lucene pour le recherche des tables, et la syntaxe Jexl pour l'extraction entre plusieurs tables.

Grâce à ces fonctionnalités, les tâches de préparation et de réconciliation de données nécessaires à la création de modèles métiers (datamarts) sont largement facilitées.

II/ Contexte du stage et Travail effectué

A) Contexte et rappel des objectifs

Le groupe BEL, spécialisé dans la production des fromages, souhaite effectuer des analyses de données en temps-réel sur ses machines de production afin de détecter une probabilité d'un taux de rebut en bout de chaîne, et de mettre en évidence les variables déterminantes dans la génération de ces rebuts.

Les deux difficultés techniques majeures résultent (i) du manque d'une clé de rapprochement commune entre machines qui aurait permis d'identifier de manière unique et certaine un même produit (ou un même lot de produits) entre deux machines, ce qui nécessite de construire un modèle statistique complexe pour reconstituer le « data-lineage » et (ii) de la capacité à industrialiser ce modèle dans une couche logicielle permettant d'effectuer ce type d'analyse statistique en flux, à la sortie des machines.

BEL a sollicité le CEA aux fins de création du modèle statistique et a sollicité Scaled Risk aux fins de démontrer les capacités de la plateforme de Data Management à industrialiser ce modèle dans un contexte de traitement en temps-réel.

B) Approche retenue

Scaled Risk comprend un module d'ETL (« Business Views ») qui permet de créer des « vues matérialisées » à partir de données sources (par exemple des données importées par des fichiers CSV). Ces vues sont créées grâce à des règles de transformation écrites en langage JEXL. Les vues se mettent à jour automatiquement lorsque de nouvelles données sources sont importées.

Transcrire au sein d'une Business View Scaled Risk l'algorithme de jointure R écrit par le CEA, permet d'utiliser l'outil Scaled Risk pour analyser en flux (temps-réel ou non) l'important volume de données générés par les machines.

Il est à noter qu'un certain nombre d'améliorations du fonctionnement des Business Views (développements cœur produit) ont été nécessaires pour permettre la transcription du script R.

C) Configuration de Scaled Risk

Import des données brutes (csv)

Importer des données à partir d'un fichier plat dans Scaled Risk nécessite deux étapes distinctes : (i) la description des colonnes afin de créer le modèle de données et donc la table technique qui viendra accueillir les données (le « type » en langage Scaled Risk) et (ii) l'import des données à proprement parler.

La première étape (création du type) consiste à décrire les champs, leurs qualités ainsi que les clés dans un fichier csv qui contient les noms des champs.

Ce fichier csv respecte un format strict tel que représenté dans cet exemple ci-dessous (type Acidification) :

```
#Acidification (#6 pH 2015 revision par produit Janv Mars 2016.xlsx)
dataset;;Acidification;;true

#Date
field;;1;date;false;0;1;date

#Heure pH moulage
field;;2;heure_ph_moulage;false;0;1;localTime

#semaine
field;;3;semaine;false;0;1;integer

#année
field;;4;annee;false;0;1;integer

#Type de produit
field;;5;type_produit;false;0;1;string

#Ligne
field;;6;ligne;false;0;1;string

#pH Mouleuse
field;;7;ph_mouleuse;false;0;1;double

#pH sortie presse
field;;8;ph_sortie_presse;false;0;1;double

#ph 2Heure acidification
field;;9;ph_2h_acid;false;0;1;double

#ph 4 Heure acidification
field;;10;ph_4h_acid;false;0;1;double

#ph 5 h 30 Heure acidification
field;;11;ph_5h30_acid;false;0;1;double

#pH J+1
field;;12;ph_j1_acid;false;0;1;double

#id
field;;39;id;true;1;1;autoInc
```

Le champ #id (de type «autoInc» ou auto-incrémentée) permet de créer une clé technique unique pour chaque ligne.

Dans l'ensemble, le format doit décrire la donnée telle qu'elle est écrite dans les fichiers CSV qui vont venir alimenter le lac.

Figure 1: Tables créées dans Scaled Risk à partir des types CSV

Une fois le type créé, la donnée brute peut être importée. Elle devient dès lors disponible et exploitable dans son format technique d'origine.

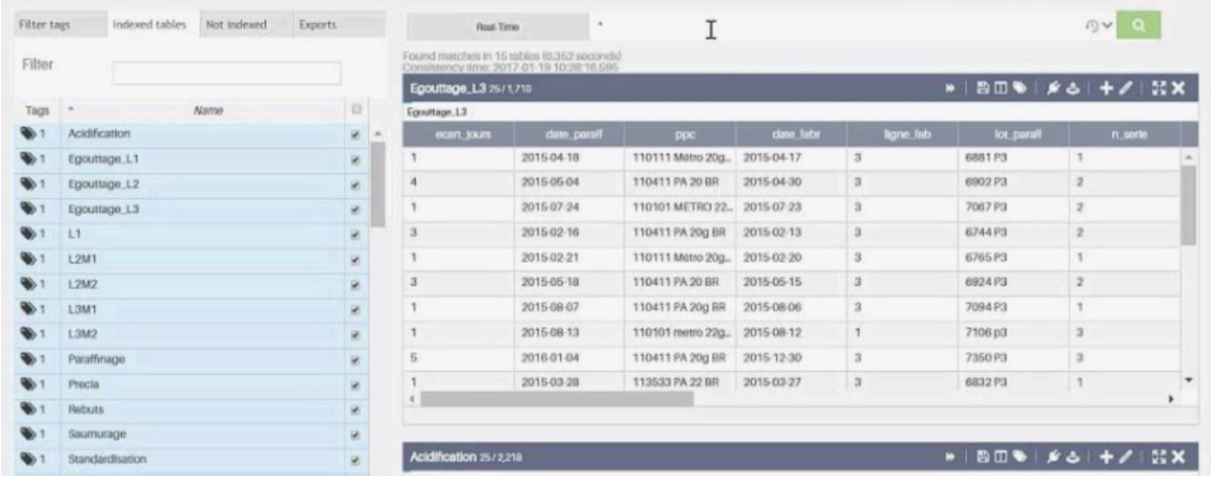


Figure 2: Donnée importée et disponible; ici via le moteur de recherche Scaled Risk

Nous avons effectué cette étape pour tous les fichiers Excel fournis (enregistrés en csv). Il est important de noter qu'ici la donnée brute est intégrée « telle quelle », dans son format technique d'origine, afin de respecter le principe d'ELT. Nous constituons ainsi une golden copy des tables initiales. La première version reste une version de référence de la donnée, qui ne sera pas modifiée.

Toute transformation effectuée ensuite créera une copie de la donnée afin de ne pas altérer cette donnée source.

D) Création de la Business View avec jointures floues

1) Principe

Comme expliqué ci-dessus, le principe ici est de « traduire » le script R en JEXL, de façon à persister la transformation dans un mode industriel. Contrairement à un moteur R qui doit réeffectuer toutes les opérations (transformations, calculs) en mémoire à chaque nouvelle donnée importée, le système des Business Views nous permet de gérer uniquement l'incrément.

2) Reconstitution des jointures floues

Pour constituer la base de données et retracer le parcours d'un lot de produits de l'étape Paraffinage, nous utilisons les informations disponibles dans la table (nom de produit, numéro de ligne, numéro de lot, et champ d'horodatage spécifique à l'étape) pour constituer une clé partielle.

En partant de la table Paraffinage, nous faisons appel à une fonction qui prend en argument cette clé partielle, et qui renvoie une ligne dans la table/étape suivante, dont les champs correspondent le plus à cette clé.

Comme la clé est composée de plusieurs champs, il se peut des fois que l'algorithme favorise un champ ou un autre, et qu'il y ait des correspondances pour un champ et pas un autre, c'est en cela que la méthode de requêtage est appelée floue.

Voici un exemple pour illustrer :

Precia_Select 3,985 / 3,985

Precia_Select

produit_precia	d_h_fin_precia	id	d_h_deb_precia	code_produit
METRO 20G BR	2016-02-19 04:1…	1015955	2016-02-19 03:40:…	113532
METRO 22G BR	2016-02-19 05:2…	1015967	2016-02-19 04:24:…	113501
METRO 20G BR	2016-02-19 05:1…	1015961	2016-02-19 04:25:…	113532
METRO 20G BR	2016-02-19 06:1…	1015973	2016-02-19 05:31:…	113532

Pour la ligne dont l'heure d'entrée dans l'étape Précia est à 4h24, le 19/02/2016, on exécute un code Jexl suivant pour récupérer la ligne la plus proche.

User Jexl

```
66
67      //on extrait les lignes de l'etape egouttage
68      var e = DB:before("Egouttage_Select_split","L" + ligne, lot_p, produit, this.d_h_deb_precia);
69      if(!empty(e)){
70          this.ligne_egouttage = e.get("ligne_egouttage").split("")[2];
71          this.lot_p_egouttage = e.get("lot_p_egouttage");
72          this.dh_sortie_egou = e.get("dh_sortie");
73          this.dh_entree_egou = e.get("dh_entree");
74          this.n_serie = e.get("n_serie");
75          this.h_trd = e.get("h_trd");
76          this.h_trm = e.get("h_trm");
77          this.h_trf = e.get("h_trf");
78          this.produit_egouttage = e.get("produit_egouttage");
79          this.tr_moyen = e.get("tr_moyen");
80          this.dmf=e.get("dmf");
81
82
```

Les autres champs de la clé partielle (ligne, lot_p, produit) sont ceux de la table Paraffinage, car les valeurs par défaut n'existent pas dans la table Précia.

L'algorithme va donc chercher la ligne dont le champ d'horodatage correspond au 19/02/2016, à 4h24, dans la table Égouttage, mais il n'y a pas une correspondance exacte :

Egouttage_Select_split 16,572 / 16,572

Egouttage_Select_split

ligne_egouttage	lot_p_egouttage	produit_egouttage	dh_sortie	dmf
L2	5394	PA 20G BR	2016-02-19 00:10:...	D
L2	5394	PA 20G BR	2016-02-19 00:45:...	M
L2	5394	PA 20G BR	2016-02-19 01:15:...	F
L2	5393	METRO 20G BR	2016-02-19 03:26:...	D
L1	6180	METRO 22G BR	2016-02-19 04:20:...	D
L2	5393	METRO 20G BR	2016-02-19 07:10:...	M
L1	6180	METRO 22G BR	2016-02-19 07:50:...	M
L2	5393	METRO 20G BR	2016-02-19 10:41:...	F
L2	5393	METRO 20G BR	2016-02-19 10:55:...	D
L1	6180	METRO 22G BR	2016-02-19 11:40:...	F

Il retourne donc la ligne avec le champ d'horodatage le plus proche, c'est-à-dire la ligne dont l'heure de sortie d'égouttage est à 4h20. C'est en cela que la requête est floue, elle ne renvoie pas la valeur exacte, car celle-ci n'existe pas, mais se contente de la valeur qui est la plus proche.

Après avoir sélectionné d'une étape à la suivante les bonnes lignes correspondant au bon numéro de lot, nous pouvons tout regrouper dans une table finale avec les données consolidées. Pour s'assurer de la consistance de la donnée, nous filtrons sur les lignes telles que les valeurs de la clé (ligne, nom de produit, et numéro de lot) soient les mêmes d'une étape à l'autre.

Les champs des étapes de Vision et de Rebuts sont aussi ajoutés.

Le pourcentage de rebuts est calculé comme le rapport de la quantité de produits déclassés sur le nombre total de fromages.

Ainsi, une table finale est créée avec les données consolidées (nous vous avons exporté et transmis son contenu sous format csv). Elle contient toutes les variables de chaque étape et c'est sur cette table que nous allons exécuter les algorithmes de détection des rebus.

E) Analyse des rebuts

Après avoir retracé le cheminement des produits, nous passons à l'étape d'analyse de la donnée. Avec la donnée numérique et catégorique (dont la valeur n'est pas numérique), on effectue 10 validations croisées de la donnée, pour lancer et benchmarker une série de 3 algorithmes d'apprentissage et de régression (un réseau de neurones multicouche, un algorithme de Random Forest, et un algorithme de Bagging), afin de prédire le pourcentage de rebuts.

Rappelons les principes des 3 algorithmes employés :

Bagging :

Le premier, l'algorithme de Bagging (Bootstrap aggregating) est un algorithme proposé par Leo Breiman en 1994 pour améliorer la classification en combinant et moyennant les sous-échantillons générés par tirage avec remise

Etant donné un ensemble de formation standard **D** de taille **N**, l'algorithme de Bagging génère **m** nouveaux ensembles d'apprentissage $\mathbf{D_i}$ chacun de taille **n**, en échantillonnant D uniformément et avec remise. En échantillonnant avec remise, certaines observations peuvent être répétées dans chaque $\mathbf{D_i}$. Si **n** = **N**, alors pour un grand **N**, l'ensemble $\mathbf{D_i}$ devrait contenir **(1 - 1 / e)** **(≈63.2%)** des observations de la donnée initiale.

Cela se démontre avec un calcul assez simple, en admettant que la probabilité qu'une observation de l'échantillon bootstrap (c-à-d l'échantillon des observations tirées) ne soit pas dans l'échantillon original/initial est de $1 - \frac{1}{N}$.
Ainsi la probabilité qu'une observation n'est **pas** dans l'échantillon de bootstrap est de $(1 - \frac{1}{N})^N$.

Donc la limite quand n est grand est de e^{-1}, et donc la probabilité que l'observation soit dans l'échantillon de bootstrap est de $1 - e^{-1}$

Des exemples uniques de **D**, le reste étant dupliqué. Ce type d'échantillon est connu sous le nom d'un échantillon de bootstrap. Les **m** échantillons bootstrap ci-dessus sont combinés en faisant la moyenne de la sortie (pour la régression) ou du vote (pour la classification) pour obtenir un estimateur de la variable d'intérêt.

Random Forest :

Supposons qu'on a $\mathbf{z} = \{(x_1, y_1) \dots (x_n, y_n)\}$ ensembles d'apprentissage avec x décrit par **p** attributs (donc x est un vecteur avec **p** coefficients)

Pour i = 1… Nb_Arbre

- Tirer un échantillon aléatoire $\mathbf{z_b}$ avec remise parmi **z**
- Estimer un arbre sur $\mathbf{z_b}$ avec randomisation des variables
- Pour la construction de chaque nœud de chaque arbre, on tire uniformément **q** variables parmi **p** pour former la décision associée au nœud

Après cette étape on possède B arbres que l'on moyenne (régression) ou qu'on vote le meilleur candidat (classification)

Le choix optimal pour **q** est souvent $\sqrt{\mathbf{p}}$

Quelques remarques :

Dans le cas de Random Forests, le tirage aléatoire des variables explicatives à chaque nœud aboutit à des arbres non corrélés

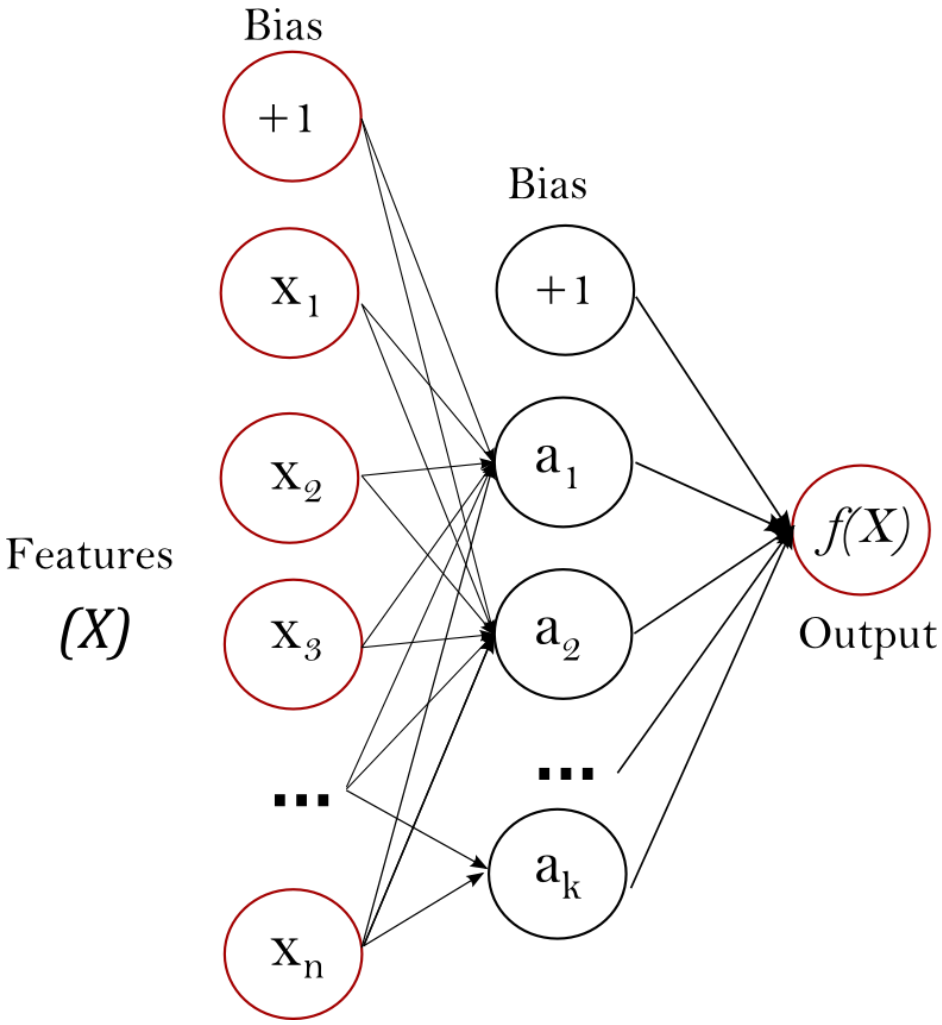
Chacun des petits arbres est moins performant mais l'union fait la force : la stratégie d'agrégation est très performante puisque ces arbres sont non corrélés.

On estime l'erreur via la méthode out-of-bag pour prévenir le over-fitting (sur-apprentissage).

On peut interpréter les résultats de RF en représentant le nombre de fois qu'une variable a été utilisée dans les B arbres construits

Multilayer Perceptron :

La couche d'entrée (1ère couche de la gauche), se compose d'un ensemble de neurones $\{x_i \mid x_1, x_2, \ldots, x_m\}$ représentant les fonctions d'entrée.



Chaque neurone dans la ou les couches cachées (les couches après la couche d'entrée) transforme les valeurs de la couche précédente par une sommation linéaire pondérée $w_1x_1 + w_2x_2 + \ldots + w_mx_m$, suivie d'une fonction d'activation non linéaire, comme la fonction Tangente hyperbolique, ou la fonction sigmoïde

$$S(x) = (1 + e^{-x})^{-1}$$

La couche de sortie reçoit les valeurs de la dernière couche cachée et les transforme en valeurs de sortie.

Les avantages du MLP sont :

- La possibilité d'apprendre des modèles non linéaires.
- La possibilité d'apprendre des modèles en temps réel

Les inconvénients du Perceptron Multicouche (MLP) incluent :

- MLP avec couches cachées possède une fonction de LogLoss non convexe où il existe plus d'un minimum local. Par conséquence, différentes initialisations de poids aléatoire peuvent conduire à une précision de validation différente.
- MLP nécessite l'ajustement d'un certain nombre de paramètres intermédiaires tels que le nombre de neurones cachés, le nombre de couches et le nombre d'itérations

La fonction LogLoss est souvent du type :

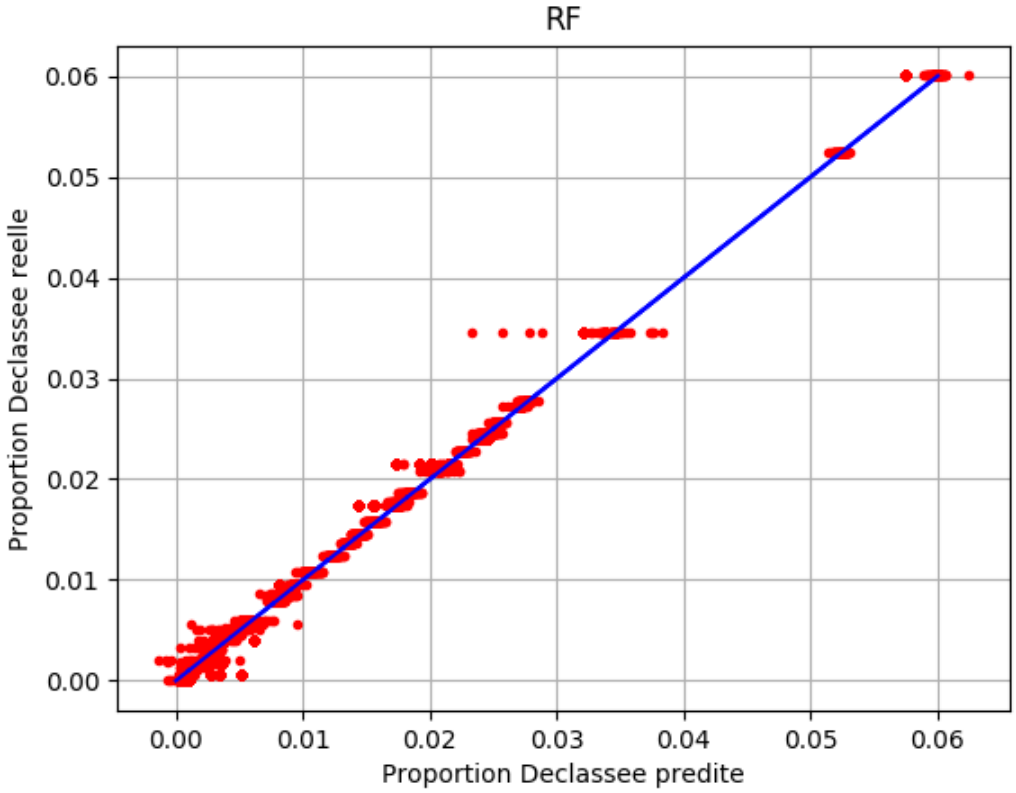
$$\mathrm{L(x)} = \sum_{\substack{k=0 \\ l=0}}^{N,M} y_{kl} \log(p_{kl})$$

C'est la fonction à minimiser pour montrer que l'algorithme de classification ou de régression est efficace. Où **N** est le nombre d'observations, **M** est le nombre de valeurs possible pour la variable à prédire, $\boldsymbol{y_{kl}}$ est un booléen qui indique si oui ou non le label/valeur **l** est la valeur correcte pour l'observation **k**, et $\boldsymbol{p_{kl}}$ est la probabilité que l'observation **k** ait la valeur prédite **l**.

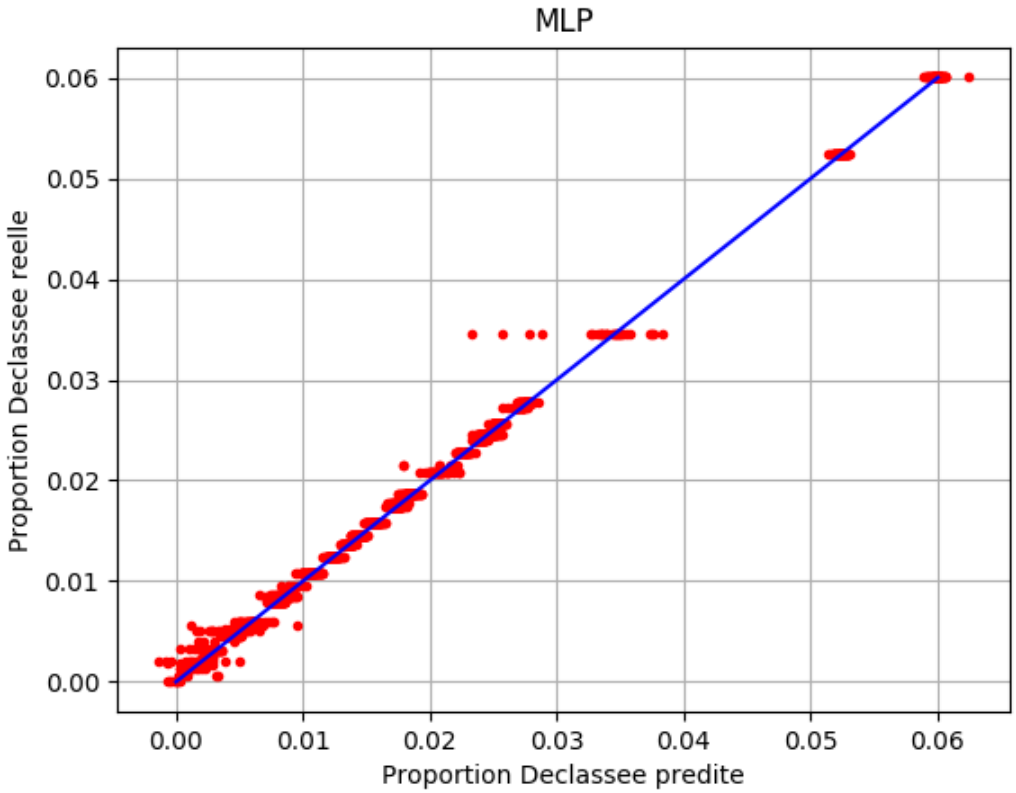
Finalement nous avons réussi à obtenir exactement les mêmes pourcentages de déclassés que ceux extraits par le CEA. Ce pourcentage varie entre 0 et 6.005%.

Nous traçons ensuite les valeurs prédites du pourcentage de rebuts par rapport aux valeurs réelles, pour les 3 algorithmes, et nous synthétisons avec un tableau regroupant les indicateurs de performance de chaque algorithme.

Random Forest



Réseau de Neurones Multicouche (MultiLayer Perceptron)



Bagging

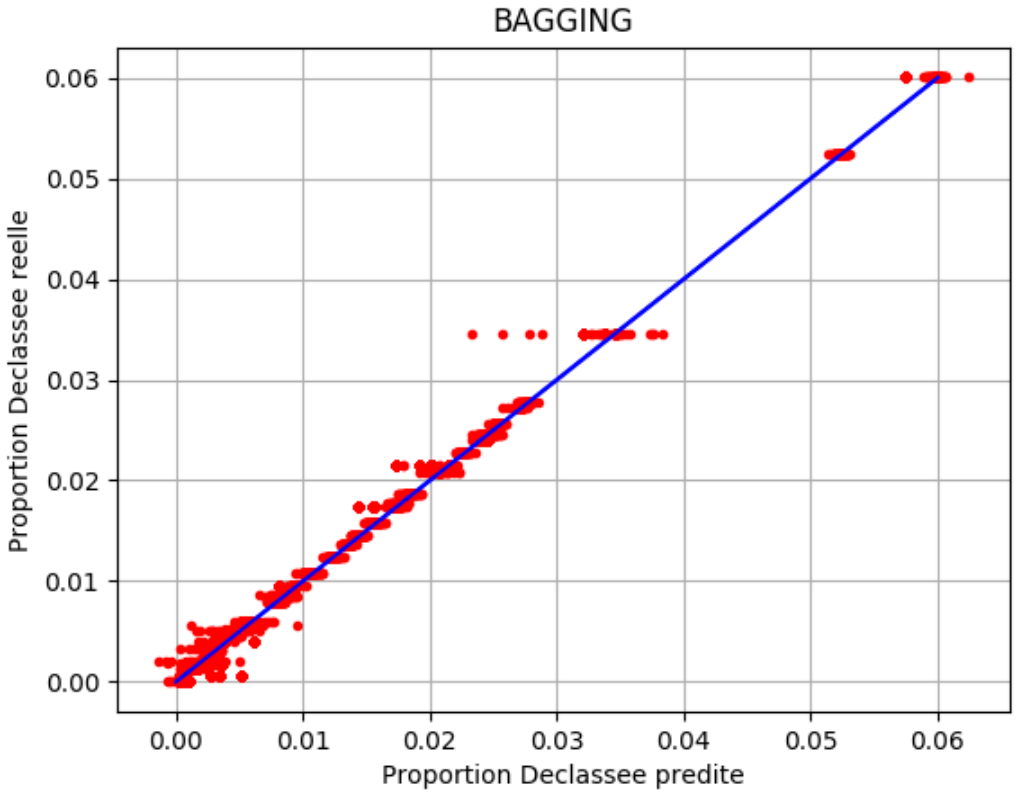


Tableau récapitulatif des performances

	Corrélation	RMSE	RAE (%)	RRSE (%)
MLP	0.9513	0.0529	27.4774 %	31.3654 %
Random Forest	0.9255	0.0645	16.3064 %	38.2243 %
Bagging	0.78	0.11	61.0532 %	68.5671 %

Quelques rappels sur les formules de calculs de ces indicateurs :

$$\text{Corr} = \frac{\sum_j^N (P_{ij} - P_{i,moy})(T_j - T_{moy})}{(\sum_j^N (P_{ij} - P_{i,moy})^2 (T_j - T_{moy})^2)^{1/2}}$$

$$\text{RMSE} = \sqrt{\frac{1}{n} \sum_{j=0}^{n} \frac{(P_{ij} - T_j)^2}{(T_j)^2}}$$

$$\text{RRSE} = \sqrt{\sum_{j=0}^{n} \frac{(P_{ij} - T_j)^2}{(T_j - T_{moy})^2}}$$

$$\text{MAE} = \sum_{j=0}^{n} |P_{ij} - T_j|$$

Où Pij correspond à la valeur prédite par l'algorithme i pour la j-ème ligne de donnée, Tj

Correspond à la valeur exacte pour la j-ème ligne de donnée, et T_{moy} correspond à la moyenne des Tj

F) Analyse des attributs significatifs

Pour cette analyse, nous avons d'abord isolé les attributs qui correspondent à chaque étape, lancé les algorithmes d'apprentissage et déterminé les étapes ayant le plus d'influence en calculant une corrélation.

Nous avons choisi d'utiliser les algorithmes Bagging, MLP et Random Forest, pour prédire le taux de rebuts.

Malgré leur bonne performance dans la prédiction des rebuts, les algorithmes Bagging et MultiLayer Perceptron se portent moins bien quand il y a moins de variables et donnent des résultats inexploitables. Nous obtenons en revanche des résultats intéressants avec l'algorithme Random Forest.

	Corrélation	RMSE	RAE (%)	RRSE (%)
Mouleuse	0.812	0.0986	37.3676 %	58.4467 %
Acidification	0.9188	0.0678	22.7055 %	40.218 %
Saumurage	0.4825	0.1478	77.8115 %	87.631 %
Egouttage	0.9712	0.0402	7.0 %	23 %
Precia	-0.0438	0.196	110.644 %	116.2178 %
Paraffinage	0.4253	0.1527	89.7869 %	90.556 %

Nous avons ensuite fait une analyse plus fine sur chacune des variables, pour déterminer celles qui sont les plus influentes.

La première approche à laquelle nous avons souscrit consiste à, pour chaque variable prise isolément, appliquer une méthode de régression avec la colonne des proportions.

Cependant, même avec les 1700 lignes de la table finale, il n'y a que 42 valeurs différentes/uniques de la proportion déclassée. Cela conduit à des corrélation assez faibles (entre 0.1 et 0.5).

Cette démarche de faire une régression sur chaque variable fournirait des résultats cohérents avec plus de données initiales.

attributs évalués	coefficients de corrélation
Nb.Position.Ganse.C2	0.14
Tr.D	0.16
pH.Sortie.Presse	0.17
pH.Mouleuse	0.20
Debit.Caille.TM1	0.29
pH.H.2	0.32
Nb.Long.Ganse.C2	0.39
Tr.Moyen	0.42
Tr.F	0.44
Nb.Diametre.C2	0.44
Tr.M	0.48

Ce premier résultat reste pertinent pour constituer une liste d'attributs qui pourront être déterminant dans la prédiction des rebuts, moyennant d'accepter de combiner avec d'autres attributs, pour qu'il y ait plus de valeurs et de lignes de données (uniques).

Voici un exemple pour illustrer :

Nous avons une table avec deux champs A et B, comme ceci :

Champ A	Champ B	Colonne C
1	2	0.01
1	3	0.01
1	4	0.03

Si nous isolons les colonnes A et C pour essayer de faire une régression, c'est à dire prédire la valeur de C en fonction de A, nous n'avons que deux lignes uniques (les lignes qui dupliquent une ligne existante n'apportent aucune information supplémentaire à l'algorithme d'apprentissage):

Champ A	Champ C
1	0.01
1	0.03

Mais en introduisant un modèle qui prend en compte les 3 champs pour faire la régression, nous avons une ligne de donnée en plus. Concrètement, c'est ce même principe qui permet de passer d'une donnée initiale de 42 lignes à un table de donnée initiale de 188 lignes, en prenant en compte plusieurs attributs qui ont un bon coefficient de corrélation.

Nous ne regardons pas les attributs qui correspondent à l'étape Vision et l'étape Rebuts, car l'intérêt est de déterminer si les attributs mesurés en amont de la chaîne de production sont déterminant. En effet, plus l'attribut significatif (pour l'apprentissage) appartient aux étapes qui sont tôt dans la production, plus le lot sera rapidement déclassé.

En ajoutant des variables au fur et à mesure, on voit que la corrélation s'améliore. Alors que la corrélation est de 0.52, pour 42 lignes uniques dans le csv et 5 variables:

- Debit.Caille.M1
- pH.H.2
- pH.Mouleuse
- pH.Sortie.Presse
- TU1
- TU2
- To

En ajoutant la variable Tr.D, nous commençons à avoir un bon coefficient de corrélation avec 0.61, puis 0.64 avec les variables Tr.F, Tr.M, et Tr.Moyen, nous arrivons à 0.75, et avec toutes les variables, nous obtenons un indice de corrélation de 0.8. Le nombre de lignes dans le csv monte à 188 lignes de données. Ces résultats s'amélioreront sans doute avec plus de donnée.

A titre de comparaison, sur un exemple de prédiction de cancer du sein par rapport à l'âge, pour un échantillon d'apprentissage de 284 lignes uniques(pour 1 paramètre),l'algorithme Random Forest donne une corrélation de 75% .

III/Conclusion

L'approche suivie permettra des ingestions de volumes extrêmement importants dans un contexte temps-réel ou quasi-temps-réel.

La capacité de traitement incrémentale de Scaled Risk permet de resserrer drastiquement le « time-to-action », car des alertes peuvent être levées extrêmement rapidement. L'utilisation de R est limitée à un contexte de R&D dans la mesure où le moteur R doit réexécuter le script à chaque modification du fichier de données (R doit charger toute la donnée en mémoire).

L'analyse des rebuts nous permet de constater que le résultat de notre transformation reste assez proche du code R original dès lors que nous arrivons à un résultat très proche dans la détermination de la proportion de déclassés obtenus par le script du CEA.

Nous constatons qu'un certain nombre de variables des étapes Mouleuse, Acidification, et Saumurage ont une forte corrélation avec les résultats obtenus de la table finale. La deuxième démarche consistant à appliquer une régression sur chacun des attributs donne à voir, malgré un manque de profondeur de données, un potentiel intéressant de certains attributs.

On note également une cohérence avec le premier résultat obtenu, car les attributs retenus appartiennent aux étapes de production qui ont obtenu un bon score de corrélation Random Forest.