PROJET MACHINE LEARNING

MASTER IEF - 2020

SUJET 2 : IMPORTANCE DES MOTS

Marie-Clémence CIUPEK– Paul GENEVRIER

Caractéristiques de la base :

On dispose d'une base de données comportant les caractéristiques de plusieurs actions entre 2007 et 2017.

Ces caractéristiques sont les suivantes :

- *'OP', 'UP', 'DO', 'CL'* - Prix Open, High, Low et Close
- *'VO'* – Volume
- *'RDMT_x'* - rendement futur à 5 jours ouvrés
- *'HISTO_x'* - rendement historique de la valeur de l'indice à la clôture du marché
- *'VOL_x'* - écart historique des volumes d'échange
- *'UP_x'* - rendement historique de la valeur au plus haut en intraday
- *'DO_x'* - rendement historique de la valeur au plus bas en intraday

x = Suffixes J, S et M indiquant respectivement

- J = une période journalière (entre 2 jours ouvrés consécutifs)
- S = une période hebdomadaire
- M = une période mensuelle.

On y ajoute les caractéristiques suivantes (calculés à partir des données de la base) :

- *'FUTUR_VO_x'* - variation du volume futur pour chaque ticker
- *'MEDIAN_VO'* - médiane du volume écahangé pour chaque ticker
- *'75_CENT_VO'* – quantile à 75% du volume écahangé pour chaque ticker

Nous avons pris soin pour l'ajout de ces caractéristiques, de bien considérer les quantiles du ticker de la ligne sur toute la période, et de même pour les variation des volumes futures, il s'agit bien à chaque ligne de l'écart entre le futur volume et le volume d'aujourd'hui pour le même ticker de la ligne.

Ces nouvelles caractéristiques vont nous permettre de créer des variables à expliquer en les combinant au rendement des actions. Plusieurs variables ont été créées mais nous ne les retiendrons pas toutes.

La base comporte également les mots parus au sein des actualités boursières journalières rattachées au ticker de la même ligne.

La base comporte **106542 lignes** et **280 colonnes**.

Types de données des colonnes de la base brute :

Colonne	Type
TICKER	object
annee	int64
mois	int64
jour	int64
OP	float64
UP	float64
DO	float64
CL	float64
VO	float64
RDMT_x	int64
HISTO_x	float64
VOL_x	float64
UP_x	float64
DO_x	float64
mots	int64

Type de données de la base retraitée :

Colonne	Type
TICKER	object
date	datetime64[ns]
OP	float64
UP	float64
DO	float64
CL	float64
VO	float64
RDMT_x	int64
HISTO_x	float64
VOL_x	float64
UP_x	float64
DO_x	float64
FUTUR_VO_x	float64
MEDIAN_VO	float64
75_CENT_VO	float64
mots	int64

Nous allons zoomer sur les mots les plus fréquents des actualités boursières et créer un modèle de prédiction sur les rendements mensuels.

Quelques statistiques de la base :

Dans un premier temps, nous avons choisi de calculer des statistiques pour chaque ticker : ***stats_by_ticker*** *(DataFrame)*

- Le rendement moyen mensuel de ce ticker : *stats_by_ticker["RDM_MOYEN_M"]*
- Le mot le plus cité pour le ticker : *stats_by_ticker["MOST_FREQ_WORD"]*
- Le nombre d'apparitions de ce mot : *stats_by_ticker["NB_WORD"]*

Puis au vu de l'objectif de ce projet, nous avons préféré calculer des statistiques plus en lien avec l'étude des modèles, pour cela nous avons calculer des statistiques par mot (tous tickers confondus) : ***stats_by_word*** *(DataFrame)*

- Le nombre d'apparition de ce mot : *stats_by_word["APPARITIONS"]*
- Le rendement moyen lorsque ce mot est cité : *stats_by_word["RDM_MOYEN_M"]*
- La fréquence qu'une hausse du rendement (future par rapport à historique) survienne lorsque le mot est cité : *stats_by_word["HAUSSE_RDMT_M"]*
- La fréquence que le volume traité du jour soit supérieur à la médiane du volume (propre au ticker concerné) lorsque le mot est cité : *stats_by_word["VO>MEDIAN"]*
- Idem mais cette fois comparé au quantile 75% : *stats_by_word["VO>QUANTIL_75"]*
- La fréquence qu'une hausse du volume traité par rapport à la veille survienne lorsque le mot est cité : *stats_by_word["V0_HISTO_J>0"]*
- La fréquence qu'une hausse future du volume traité par rapport à demain survienne lorsque le mot est cité : *stats_by_word["V0_FUTUR_J>0"]*

Nous trions ce dataframe par '*APPARITIONS*' puis par '*RDM_MOYEN_M*'.

En voici les 20 premières lignes :

	APPARITIONS	RDM_MOYEN_M	HAUSSE_RDMT_M	VO>MEDIAN	VO>QUANTIL_75	VO_HISTO_J>0	VO_FUTUR_J>0
pour	1966	1.36%	48.27%	51.07%	26.81%	54.27%	46.74%
avec	1737	0.94%	47.09%	50.37%	24.99%	52.56%	44.67%
sous	1502	0.74%	52.93%	66.11%	39.55%	67.84%	33.95%
pres	994	0.85%	46.98%	64.79%	39.74%	62.78%	36.42%
dans	981	0.95%	49.44%	46.48%	20.80%	51.48%	46.28%
action	979	0.96%	48.01%	45.45%	21.65%	50.15%	49.03%
d_un	889	0.47%	47.81%	49.83%	25.08%	56.02%	48.37%
vers	807	0.73%	46.34%	72.37%	45.97%	72.00%	31.35%
achat	790	0.38%	48.23%	56.58%	31.01%	55.19%	41.01%
capital	761	1.20%	50.99%	48.75%	24.18%	50.46%	46.78%
capital.	751	1.10%	51.00%	48.20%	23.70%	50.47%	46.87%
part	716	1.19%	48.32%	52.51%	26.68%	56.01%	46.93%
objectif	**636**	**1.47%**	**46.07%**	**73.58%**	**43.08%**	**62.58%**	**33.33%**
actions	607	1.02%	48.76%	44.15%	20.10%	49.92%	47.78%
contrat	540	1.06%	48.70%	43.52%	17.96%	49.44%	51.48%
passe	527	1.38%	49.53%	49.53%	26.00%	57.50%	44.97%
nouveau	511	1.19%	47.16%	46.58%	21.72%	48.34%	48.14%
nouvel	494	1.15%	47.37%	46.15%	22.06%	49.19%	48.18%
group	493	1.29%	47.06%	50.51%	27.79%	51.72%	47.06%

On remarque par exemple pour le mot '*objectif*' qu'il a un rendement moyen plus élevé que les autres, et que le niveau de volume traité est beaucoup plus élevé lors des jours où ce mot est cité. En effet, lorsque 'objectif' est cité, le niveau de volume du jour est au-dessus de son quantile 0.75 lors 43% de ces citations au lieu de 25%. De plus la variation de son volume par rapport à la veille est

positive lors de 63% de ses apparitions. Pour ce mot il semble avoir un effet sur le volume non négligeable.

Sélection des données :

On filtre ensuite la base en ne conservant que les lignes où se trouve au moins un des mots apparait plus de 400 fois (quelque soit le ticker) et dont le rendement mensuel est en moyenne supérieur à 1%.

Ces mots sont les suivants :

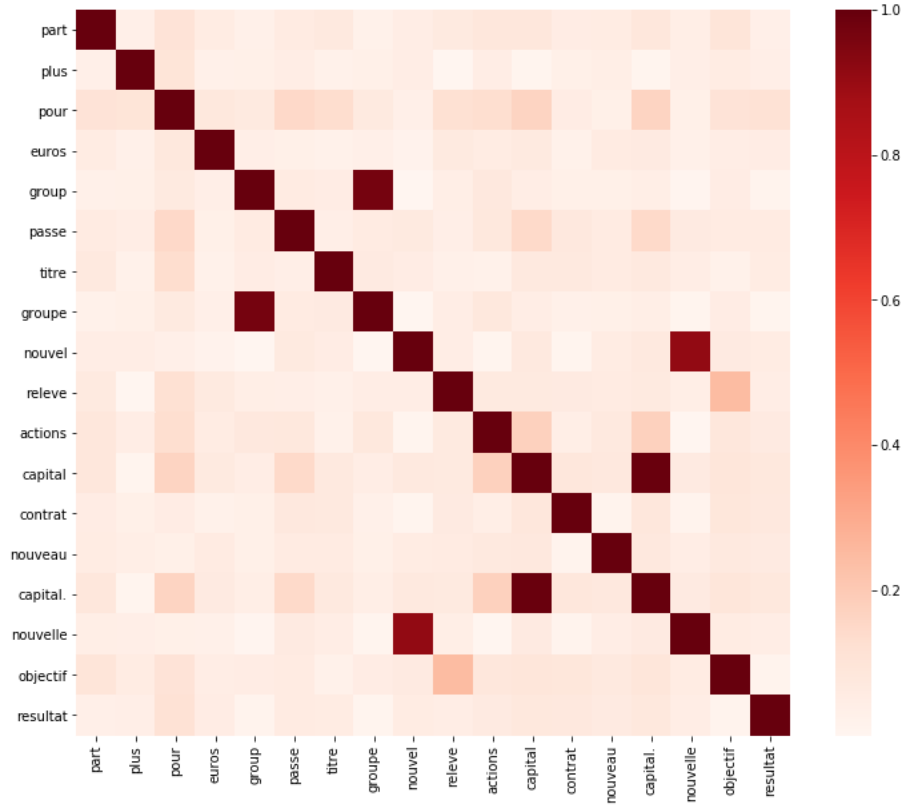
Mot	Nombre d'apparitions	Rendement mensuel moyen
part	716	0.0119
plus	418	0.0101
pour	1966	0.0136
euros	446	0.0104
group	493	0.0129
passe	527	0.0138
titre	467	0.0122
groupe	467	0.0146
nouvel	494	0.0115
releve	413	0.0103
actions	607	0.0102
capital	761	0.012
contrat	540	0.0106
nouveau	511	0.0119
capital.	751	0.011
nouvelle	415	0.012
objectif	636	0.0147
resultat	458	0.0112

On obtient une base filtrée de 7129 lignes.

Cette filtration va nous permettre de faire notre apprentissage sur une base plus significative dans le sens où les mots sur lesquels le travail va être fait ont une fréquence d'apparition élevée et semblent avoir un impact non négligeable sur le rendement de l'actif.

Retrait des variables trop corrélées :

Matrice de corrélation des variables :



On retire les variables trop corrélées positivement ou négativement i.e. les variables dont la valeur absolue du coefficient de corrélation est supérieure à 75%. Cette filtration basée sur la corrélation permet de retirer les mots en doublons ou étant très proches (nouveau/nouvel, capital/capital., group/groupe, …). Ces doublons risquent de fausser l'apprentissage.

A la recherche d'AUC :

Afin de trouver le meilleur modèle nous avons focalisé notre recherche sur l'AUC en premier lieu. Cependant nous nous sommes vite aperçus des limites de ce raisonnement. Nous allons expliquer les différentes étapes de notre démarche finalement utilisée pour ce projet.

Il a fallu continuellement reconsidérer les métriques, les hyperparamètres et évidemment le choix de la variable à expliquer afin d'obtenir un modèle satisfaisant.

Un AUC, oui, mais à quel prix ?

Le choix de la variable à expliquer doit prendre en compte, le fait que la variable doit avoir un nombre suffisant de valeurs positive (y=1). Sinon, lorsque la majorité de la variable à expliquer est trop faible, l'optimisation sur l'AUC sera biaisée. En effet, dans le cas où seulement 1% des y sont positives, alors si le modèle prédit uniquement des valeurs négatives (y_pred=0), il aura une précision et un recall nuls, pourtant ne se trompe rarement car la grande majorité des variables à prédire sont nulles.

- ⇨ Lors de notre projet cela nous est arrivé lors plusieurs essais de variables à expliquer, et nous obtenions des AUC entre 0.75 et 0.85, nous pensions avoir trouver le bon modèle en se référant au score d'AUC, mais c'était tout simplement la variable à expliquer qui n'était pas adaptée

Le choix des métriques :

Dans un premier temps, nous nous sommes focalisés sur l'AUC, cependant comme expliqué au-dessus nous nous sommes rendu compte de l'importance de considérer le recall et la précision pour juger de l'efficacité du modèle.

Puisque le but du projet est de prédire un signal d'achat, nous avons souhaité favoriser la précision plutôt que le recall. Quitte à manquer des opportunités, nous préférons investir dans moins de de prévisions positives mais plus souvent bonnes.

Là encore nous avons remis en question ce choix, plusieurs modèles nous permettaient d'obtenir un AUC proche de 0.7 et une précision entre 0.65 et 0.75. Mais ces modèles donnaient un recall quasi nul (par exemple nous prédisions 2 valeurs « Vrai Positif » et 1 valeur « Faux Positif »).
Nous ne jugeons pas optimal de laisser passer de nombreuses opportunités pour en jouer seulement 2 avec une précision de 0.67

Ainsi nos métriques d'optimisation sont l'AUC (de la courbe ROC) et le recall.

Pour l'hyperparamétrage, il faut choisir une parmi ces 2 métriques pour le *refit*. Nous avons testé les 2. Pour certains cas de notre étude prendre le recall comme ***refit*** entraînait une très faible amélioration du recall et une baisse de l'AUC, alors que si l'AUC est utilisé comme ***refit*** nous améliorons notre AUC, en perdant relativement peu en recall, et même parfois en gagnant en précision. On a donc privilégié l'**AUC comme métrique de *refit***.

Le choix de la variable à expliquer

1) Nous avons premièrement regardé la variable binaire conditionnée par un rendement futur moyen supérieur à 2%. Script *: y=filtered_data.RDMT_M.apply(lambda x : 1 if x>= 0.02 else 0)* Comme dit ci-dessus, nous nous focalisions seulement sur l'AUC au début, avec de nombreux hyperparamétrages sur différents hyperparamètres, l'AUC semblait avoir un maximum de 0.55.
2) Nous avons donc choisi de construire d'autres variables à expliquer et de les tester dans notre modèle à travers différents d'hyperparamétrages.
   Voici 2 autres variables binaires que nous avons regardées et analysées :
   - **Rendement futur supérieur à 2% et (Volume du jour > Quantile_Volume(0.75) ou Hausse future du volume journalier d'au moins 75%)**
     ➔ ici nous rajoutons une condition sur le volume, nous expliquons ce choix par le fait que des actualités positives sur une entreprise provoque généralement un « rush » des investisseurs sur l'action de l'entreprise, entraînant généralement une hausse du volume traité.
     Si la nouvelle apparaît au cours de la journée alors ce sera le volume de ce jour qui sera élevé (d'où la condition > au quantile(75%)), si elle apparaît après la fermeture du marché , ce sera le volume du lendemain qui sera impacté que celui de la veille (d'où hausse > 0.75).
     Pour cela nous avons dû calculer pour chaque ticker son quantile (75%) de volume quotidien, et à chaque ligne du ticker calculer la variation future du volume à 1 jour.

     Script : *y2=filtered_data.RDMT_M.apply(lambda x : 1 if x>= 0.02 else 0)*\
     ((filtered_data.FUTUR_VO_J).apply(lambda x : 1 if x>0.75 else 0)+
     (filtered_data.VO-filtered_data['75_CENT_VO']).apply(lambda x : 1 if x> 0 else 0)-
     *((filtered_data.FUTUR_VO_J).apply(lambda x : 1 if x>0.75 else 0)**
     (filtered_data.VO-filtered_data['75_CENT_VO']).apply(lambda x : 1 if x> 0 else 0)))

   - **Rendement futur supérieur à 2% et (Rendement futur mensuel > Rendement historique mensuel)**
     ➔ Ici nous rajoutons une condition sur la hausse du rendement, lorsque le cycle économique est propice aux actions, le rendement futur est plus une généralité sur de la dynamique de marché qu'une réaction à des actualités/nouvelles positives, ainsi on cherche les rendements positifs en hausse par rapport au mois dernier.

     Script : *y3=(filtered_data.RDMT_M-filtered_data.HISTO_M).apply(lambda x : 1 if x>= 0 else 0)*(filtered_data.RDMT_M.apply(lambda x : 1 if x>0.02 else 0))*

   Nous avons regardé d'autres variables binaires selon les niveaux et variations du volume (Journalier, Hebdo, Mensuel), mais les niveaux d'AUC n'étaient pas satisfaisants donc nous les avons exclues pour la suite de notre étude. En voici quelques exemples :

   *#y=filtered_data.RDMT_M.apply(lambda x : 1 if x>= 0.02 else 0)**
   # (filtered_data.VO-filtered_data['MEDIAN']).apply(lambda x : 1 if x> 0 else 0)
   *#y=filtered_data.RDMT_M.apply(lambda x : 1 if x>= 0.02 else 0)**
   #(filtered_data.FUTUR_VO_M).apply(lambda x : 1 if x> 0.25 else 0)
   *#y=filtered_data.RDMT_M.apply(lambda x : 1 if x>= 0.02 else 0)**
   #(filtered_data.FUTUR_VO_S).apply(lambda x : 1 if x> 0.5 else 0)

3) Finalement après avoir effectué des hyperparamétrages pour chaque variable binaire (voir partie suivante), la plus consistante est la 3ème variable (***y3***): **rendements futurs mensuels > 2% et rendements en hausse**.

Variable	AUC (ROC)	Recall	Precision
y2	0.72	0.11	0.52
y3	**0.69**	**0.33**	**0.49**

En choisissant la variable y3 on sacrifie 0.03 d'*AUC* et 0.03 de *Precision* pour multiplier par 3 notre *Recall*.

Représentation graphique de la variable choisie :

On utilise le package *plotly* pour obtenir le graphique en barres ci-dessous qui associe à chaque ligne de la base le rendement mensuel correspondant (variable à expliquer). La couleur de la barre dépend de la valeur de notre variable explicatrice y3. Une barre jaune indique y3 = 1 et une barre bleue y3 = 0.

A noter qu'il s'agit d'un zoom (seulement 3000 rendements ici), car pour plus de 7000 le graphique devenait illisible.

La méthode d'hyperparamétrage : Un hyperparamétrage en 2 temps

Ce qui suit a été fait pour chacune des 2 variables (*y2* et *y3*)

1) Un modèle de départ :
   - Fonction objectif : « binary : logistic »
   - n_estimators : 200
   - learning_rate : 0.1

   ➔ on cherche à avoir une idée des paramètres, donc dans un premier temps on souhaite de la rapidité

2) Un 1er hyperparamétrage rapide : Optimisations successives sur les hyperparamètres

a) Optimisation sur *max_depth* et *min_child_weigth*
Si un des hyperparamètres et à la limite du range, on déplacer le range et recommencer en conservant le meilleur autre hyperparamètre
➔ On conserve le meilleur modèle

b) Optimisation sur *gamma* sur le meilleur modèle obtenu en a)
➔ On conserve le meilleur modèle (Attention précision et/ou recall peuvent être nul, si refit='AUC')

c) Optimisation sur *sub_sample* et *colsample_bytree* sur le meilleur modèle obtenu en b)
➔ On conserve le meilleur modèle

d) Optimisation sur *sub_sample* et *colsample_bytree* sur le meilleur modèle obtenu en c)
➔ On conserve le meilleur modèle

Dans le fichier *pre_tunning.py*, il y a le script à titre d'information que nous avons exécuté pas à pas afin d'obtenir un premier aperçu d'un hyperparamétrage. A partir du modèle obtenu on élargit ses hyperparamètres à un grid d'hyperparamètres.
Attention ce grid doit tout de même contenir les valeurs par défault des hyperparamètres, car parfois ces valeurs par défaut sont optimales.

Voici nos grids choisis pour *y2* et *y3* :

```
parameter_space_y2={'max_depth':[6,9,10], 'min_child_weight':[1,2,3], 'gamma':[0,0.4,1],
                    'subsample':[0.9,1], 'colsample_bytree':[0.9,1], 'reg_alpha':[0,1],
                     'learning_rate':[0.02], 'n_estimators':[500]}
parameter_space_y3={'max_depth':[6,9,10],  'min_child_weight':[0,1,1.5], 'gamma':[0,1],
                    'subsample':[0.7,0.95,1], 'colsample_bytree':[0.7,0.95,1],
                    'reg_alpha':[0,1], 'learning_rate':[0.02],  'n_estimators':[500]}
```

3) Un 2ème hyperparamétrage plus précis : Optimisation sur les hyperparamètres en simultanée
A partir du modèle obtenu en d) on définit un grid élargi depuis les valeurs des hyperparamètres du modèle (d).
On fixe :
- n_estimators : 500
- learning_rate : 0.02

➔ on cherche à être plus précis maintenant.
<u>On obtient ces modèles :</u>

```
tuned_model_y2=xgb.XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
        colsample_bynode=1, colsample_bytree=0.9, gamma=1,
        learning_rate=0.02, max_delta_step=0, max_depth=6,
        min_child_weight=2, missing=None, n_estimators=500, n_jobs=1,
        nthread=4, objective='binary:logistic', random_state=1,
        reg_alpha=1, reg_lambda=1, scale_pos_weight=1, seed=1,
        silent=False, subsample=1, verbosity=1)

tuned_model_y3=xgb.XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
        colsample_bynode=1, colsample_bytree=1, gamma=1,
        learning_rate=0.02, max_delta_step=0, max_depth=10,
        min_child_weight=0, missing=None, n_estimators=500, n_jobs=1,
        nthread=4, objective='binary:logistic', random_state=1,
        reg_alpha=0, reg_lambda=1, scale_pos_weight=1, seed=1,
```

silent=False, subsample=1, verbosity=1)

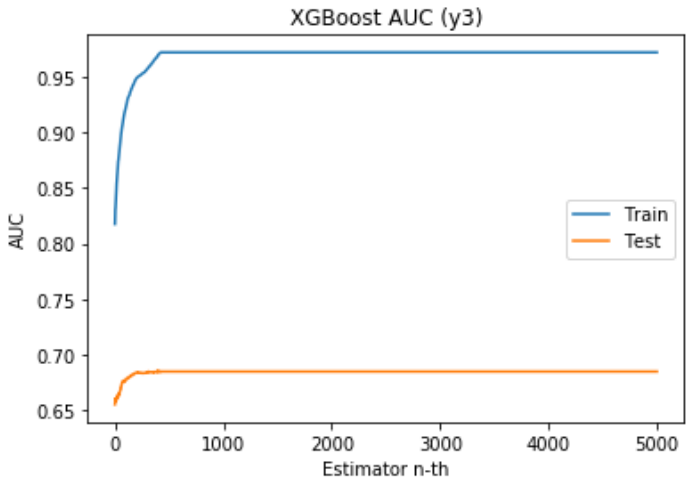
On trouve les scores présentés dans la partie « Choix de la variable à expliquer ».

La prévision finale :

Nous avons appliqué cette méthode à nos 2 variables à expliquer.

Nous conservons celle pour laquelle les métriques sont les plus hautes (rappel on vise AUC = 0.75 et Precision = 0.4). Comme énoncé dans la partie « Choix de la variable à expliquer», nous conservons la variable qui est aussi conditionnée par l'augmentation du rendement par rapport à la dernière période (*y3*).

Maintenant essayons d'affiner une dernière fois notre modèle sélectionné. On augmente le nombresd'estimations (*n_estimators)* à 10 000. On perd de la rapidité du modèle, mais comme on le voit graphiquement l'AUC n'augmente plus à partir d'un certain nombre d'estimations, donc en ajoutant un principe d'arrêt anticipé (*early_stopping_rounds*) sur la métrique AUC nous pouvons gagnons en rapidité.



On fixe donc *early_stopping_rounds=500* pour compenser cette hausse de temps d'exécution.

Pour espérer un meilleur AUC, nous avons fait varier fixer l'hyperparamètre *learning_rate* à un niveau plus faible que 0.02 sur le modèle final, plus cet hyperparamètre est faible plus le modèle est coûteux en temps. En ayant essayé 0.01, 0.001 et 0.0001, on ne gagne pas en AUC, ni en précision et ni en recall. Donc nous laissons le taux d'apprentissage à 0.02

Finalement d'après notre étude, ce modèle capte plus de 1/3 de nos signaux d'achats. Lorsqu'il prédit une hausse de rendement par rapport au mois précédent et un rendement supérieur à 2%, il y a 50% de chance que ce soit vrai.

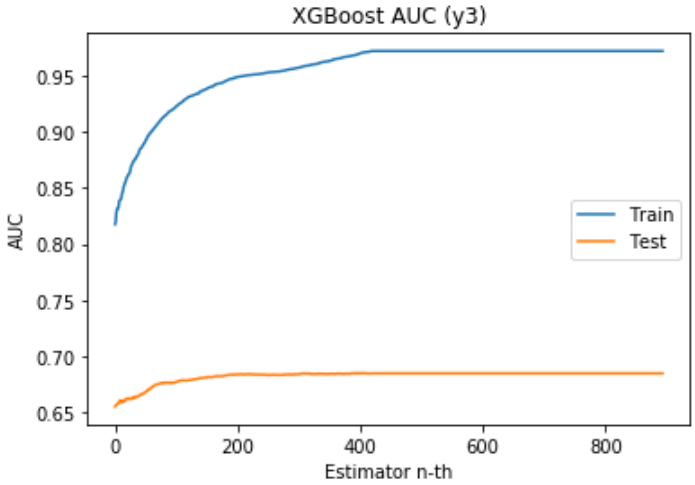
Matrice de confusion :

156 (*TP*)	314 (*FN*)
159 (*FP*)	797 (*TN*)

Score :

AUC (ROC)	Recall	Precision
0.69	0.33	0.50

Courbe d'AUC en fonction du nombre d'estimation (après ajout de *early_stopping_rounds*) :

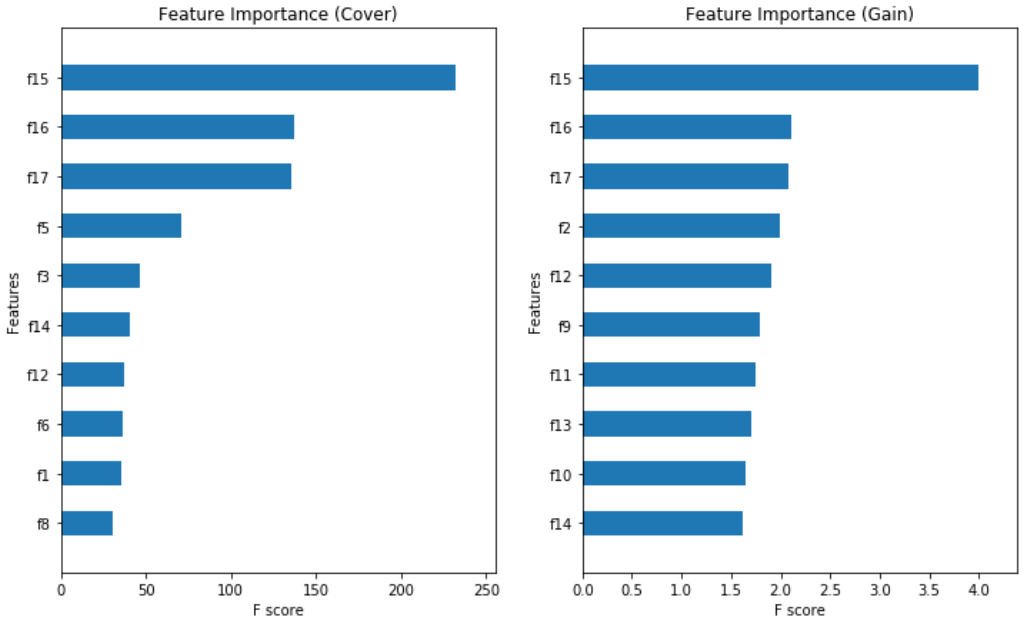


Importance des variables explicatives :

Parmi les variables explicatives, on y compte l'apparition des mots de la liste filtrée, le rendement historique mensuel, le volume du jour, la variation historique du volume journalier.

Script : *list_input=words.columns.tolist()*
list_input+=['HISTO_M','VO','VOL_J']
X=filtered_data[list_input]

Pour sélectionner les variables les plus importantes nous regardons *Cover* et *Gain* des features du modèle. Puisque les *features* « mots » sont binaires, le *weight* n'est pas adapté pour interpréter l'ordre d'importance des *features*.



Evidemment, le niveau de rendement du mois précédent (*HIST_M : f15*) est largement classé 1er pour le *gain* et le *cover*, c'est logique puisqu'il fait partie de la condition de la variable à expliquer. De ces graphiques, on remarque aussi que les données sur le volume (*VO : f16 ; VOL_J : f17*) jouent un rôle non négligeable dans le modèle (2e et 3e pour le *Cover* et pour le *Gain*), donc il est rationnel de laisser ces 2 données de volume dans nos inputs.

Intéressons-nous aux mots en particulier :
Seul 1 mot est présent dans le top 10 du *Gain* et du *Cover*, c'est *f12 :* ***'nouveau'***.
En terme de *Cover* les données sur le volume ont bien plus d'importance, cependant pour le *Gain*, l'ordre de grandeur entre les mots les plus importants et le même que les *features* Volume.

Donc les mots ont bien « leur mot à dire » dans la prévision du modèle, car ils ont un impact équivalent aux données de volume sur l'*Accuracy* du modèle
Les 4 premiers mots avec le plus d'impact sur le *Gain* du modèle sont :

- ***f2 : pour***
- ***f12 : nouveau***
- ***f9 : actions***
- ***f11 : contrat***

Les mots *pour*, *nouveau* et *contrat* ont d'ailleurs un sens « positif » lors d'une actualité d'entreprise