# Rapport hebdomadaire de stage

Alexis Blanchet

4 mai 2018

## 1 INTRODUCTION

Cette semaine a été consacrée a une tester deux méthodes de Machine Learning sur le sujet qui nous occupe. A cette fin, la journée de lundi a été consacrée au visionnage de TF1 en direct depuis leur site avec un monitoring des courbes d'audiences en temps réel afin de noter les débuts et fins réel des programmes et publicité sur une durée de temps significative. Cela permet en même temps d'étudier la corrélation entre chute d'audience, publicité et programmes. N'étant pas familier avec les programmes diffusés par TF1 (je n'ai malheureusement jamais eu de télévision et donc doit apprendre, en même temps que la machine,les spécificités des programmes) , j'ai pu ainsi noter les pratiques des débuts et fins de programmes (génériques, annonces des prochains épisodes,pas de coupure de pub,..). En parallèle, d'autres recherches ont été effectués pour ajouter des features au Times series afin de faciliter la détection d'évènements (Kernel For Density Ratio, similarité des pentes, présence de pics). Le reste de la semaine a été consacré a l'implémentation et le test de différents algorithmes de ML pour utiliser l'historique créé lundi. Furent testé LightGBM,CatBoost,XgBoost pour les arbres de décision et CNN,RNN et LSTM pour les réseaux de neurones. On reviendra en détail sur les résultats de chacun.

## 2 PRÉSENTATION DES FEATURES DES TIMES SERIES

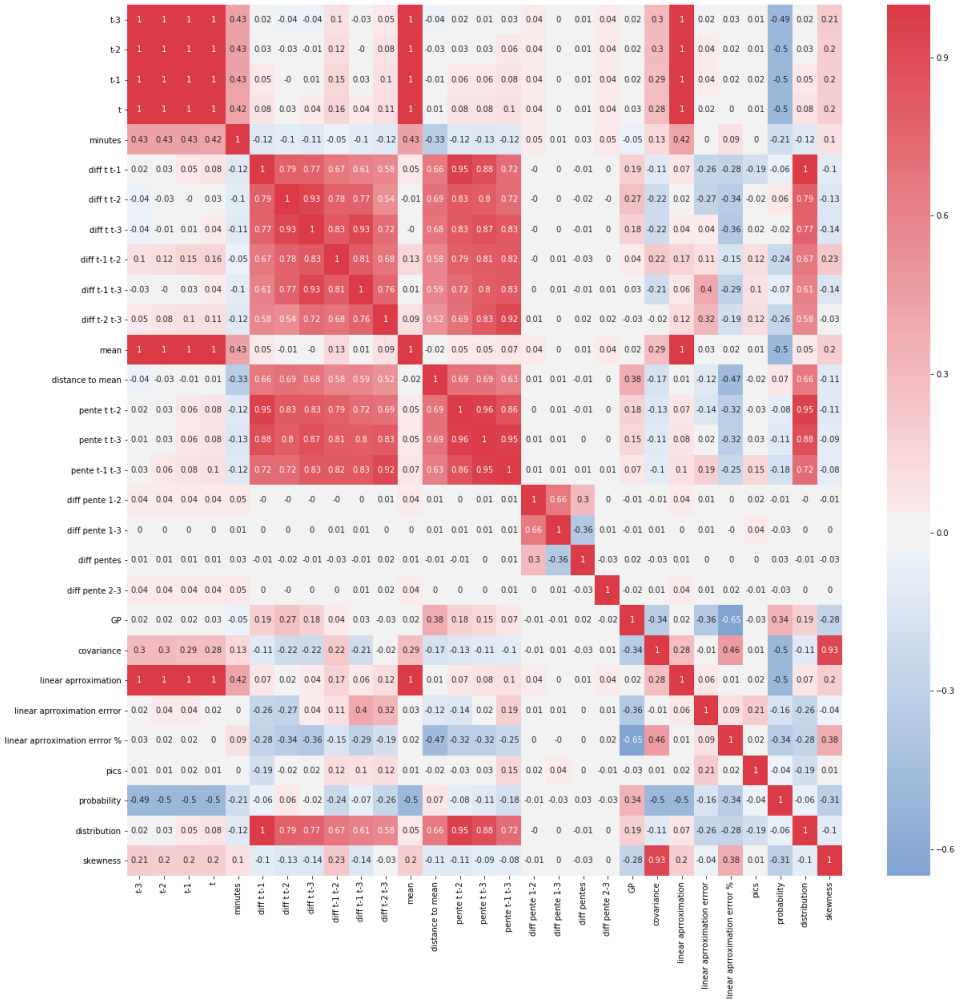
Afin de détecter les événements notables dans la journée, on utilise une fenêtre mouvante de 4 minutes (de t a t-3) afin d'utiliser le passé proche mais pas le futur(le modèle doit être conçu pour de l'utilisation en direct).Le choix de 4 minutes est arbitraire et pourra être sujet a révision. Les features utilisés sont dissociés en deux catégories distinctes :

1. Les features a l'intérieur de cette fenêtre : on prend par exemple les pentes, la distance, la distance a la moyenne, la pente relative, la distance relative, la probabilité des distances(*), la présence de pics...
2. Les features entre fenêtres : la covariance de chaque fenêtre,la moyenne de chaque fenêtre,la skewness et le kurtosis, la KFDR(Kernel For Density Ratio)

(*) l'ensemble des distances entre points consécutifs d'une journée pour une chaîne permet de trouver statistiquement la distribution gaussienne de ces distances et donc de trouver la probabilité (on troncature les outliers pour trouver cette distribution)

ces features sont sujets a de constantes évolutions en fonction des résultats trouvés et de l'avancée de mes recherches.

ci dessous la matrice de corrélation des features ad hoc.



# 3 VISIONNAGE TF1

La notation des événements de la journée du 30 Avril a débuté a 9h45 pour se finir a 17h45. Sont noté la fin et le début des programmes, le début et la fin des pages de pubs, et la présence ou non des panneaux publicitaires a la fin des programmes. On notera quelques points intéressants qui ressortent de cette étude :

1. Dans les heures relativement creuses, ou alors entre deux épisodes d'une même série, la publicité en fin de programme est inexistante : a 11h10 par exemple la fin du programme est sans pub mais 7 minutes plus tard, une plage de pub de 3 min intervient.Même chose a 17h10, sans doute a cause d'un léger retard dans le planning.
2. le retard dans le programme : demain nous appartient devait débuter a 10h25 et finir a 11h mais a débuter a 1039 et a fini a 11h10, les feux de l'amour qui débute juste après et qui sont programmés de 11h a 12h ont duré de 11h10 a 11h52 avec deux interruptions publicitaires de respectivement 3 et 6 minutes et finit sur une page de pub de 6 minutes.
3. la durée écrans publicitaires varie : 3, 6 minutes le plus souvent, mais aussi 9 minutes entre deux programmes longs (15h31-15h40). il est donc difficile de prédire avec certitude la durée des publicités ou encore la place dans le programme.

ces données seront a traiter pour permettre une mise en contexte des événements. Une prédiction des temps de publicité, de temps restant avant une prochaine publicité ou encore la présence ou non d'écran publicitaire entre deux programmes faciliterait grandement la classification des événements détectés mais la dite détection.

# 4 MACHINE LEARNING

Une fois l'historique constitué je l'ai un peu retravaillé pour ajouter les données manquantes et facilement détectables (publicités après 21h, fin des 12 coups de midi) mais ai évité tout autre ajouts. Le labelling est bruité et cela posera problème mais nous y reviendrons.

## 4.1 DECISON TREES

La première tentative d'exploitation des données fut de les donner en entrée d'un XgBoost (et de ses variantes LightGBM et CatBoost) afin d'étudier les premiers résultats. J'ai annoté a la main une frise temporelle afin de procéder a une classification binaire : soit il y a un événement (début/fin de pub/programme) soit il ne se passe rien. C'est a dire que le temps de publicité est considéré comme un programme court ne présentant pas d'annotation. Il s'agit la d'un premier jet et différentes méthodes de classification seront testés afin de comparer les résultats. Il s'agit la de créer en premier lieu une baseline pour un futur benchmark. De plus aucun affinage précis des paramètres des algorithmes n'a été fait. J'ai utilisé des templates de mes précédents travaux. Pour la prédiction de contrôle on utilisera la moyenne non pondérée de 2 XgBoost, 2 CatBoost et 2 LightGBM avec a chaque fois des paramètres différents. Aucune comparaison n'a encore été faite entre le classifier de base et et l'aggregat, aucun pondération n'a été testée. De même aucun stacking n'a été encore fait mais cela sera a l'ordre du jour

comme expliqué dans le rapport 1 dans le récapitulatif du modèle final. la mesure utilisée ici est une binary_logloss et les classifiers tournent autour de 0.10 de logloss.

### 4.2 Long short term Memory

Est ensuite utilisé un LSTM avec la même annotation afin de comparer avec les résultats des arbres. De même, aucun travail n'a été fait sur les paramètres (seul un ajustement du batch size et de l'input dimension ont été réalisé) Le LTSM est utilisé avec Keras et Tensorflow en backend. On utilise ici aussi une binary_logloss comme mesure mais on lui ajoute trois mesures que l'on définit/code par nous même : F1-score,Recall,Precision (définition et discussion dans la partie suivante). Les résultats sont ici un peu plus probants que pour les arbres de décision mais reste du même ordre pour un labelling bruité et incomplet (on remarquera que autour de 9h45 la détection est faible). on détécte de plus souvent plusieurs points d'affilé dans la publicité et non pas juste le premier. Cela est en soit peu important car on se trouve encore dans la publicité mais peut poser problème si la chute est dû au début d'un nouveau programme sans pub.

### 4.3 Discussion sur l'évaluation du modèle

Tout d'abord, l'évaluation du modèle dépend entièrement du labelling de départ, et du labelling que l'on souhaite obtenir a la fin : Souhait on détecter le DÉBUT des événements (fin de programme/début de pub) et la FIN (fin de pub et début du nouveau programme) pu alors seulement l'un des deux. De même doit on prendre en compte l'ensemble des points dans la pub ou seulement les extrema. De plus, si nous labelisons au fur et a mesure que l'on avance dans le temps, alors le label des événements précédents va intervenir fortement dans le labelling du point considéré et donc l'algorithme de classification a posteriori et celui en temps réel ne peuvent pas être entrainé de la même façon.

S'en suit une validation/évaluation du modèle : il semble logique d'utiliser les mesures classiques pour évaluer le modèle : binary_logloss (si deux classes sinon multi_logloss) mais cela ne suffit pas car les classes ne sont pas de cardinal semblable (de l'ordre de 1/9) et donc nous devons prendre en considération d'autres paramètres.

Donnons pour commencer quelques définitions :

Soit E la classe des événements, non E et alors un point régulier au quel rien de notable n'est survenu alors on a le tableau suivant :

| | classé E | classé non E |
|---|---|---|
| vrai E | True Positive | False Negative |
| vrai non E | False Positive | True Negative |

De ce tableau nous allons extraire plusieurs mesures qui seront importantes dans la suite des travaux :

Tout d'abord l'Accuracy qui le calcule le ratio de points bien classé sur le nombre de point total. Ici elle sera peu utile du fait de l'importance du déséquilibre entre les deux classes considéré dans notre cas.

$$Accuracy = \frac{TP + TN}{TP + TN + FN + FP} \tag{4.1}$$

La sensibilité est plus intéressante car elle donne le nombre de points importants bien détectés par l'algorithme, mais ne considère pas le nombre de False Negative

$$Recall/sensitivity = \frac{TP}{TP+FN} \tag{4.2}$$

la precision permet de combler la défaillance de la mesure précédente

$$Precision = \frac{TP}{TP+FP} \tag{4.3}$$

la moyenne permet d'avoir une idée générale de la bonne classification pour les deux classes et est utile car pénalise aussi bien la non détection que la détection abusive.

$$G-mean = \sqrt{\frac{TP}{TP+FN} * \frac{TN}{FP+TN}} \tag{4.4}$$

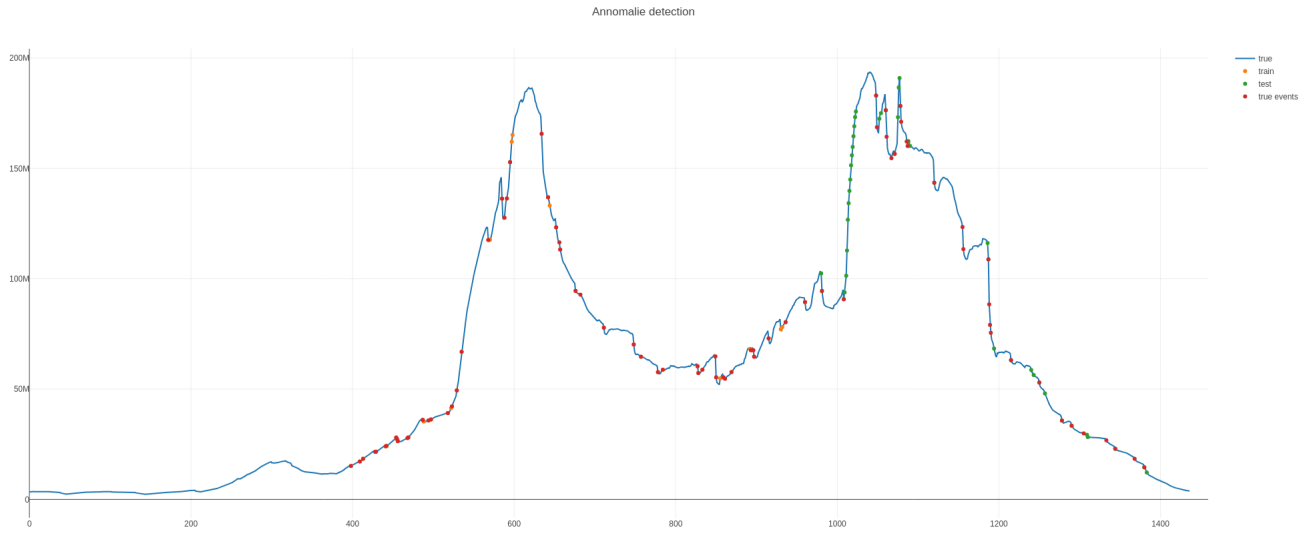
Enfin la plus classique des mesures est la F-beta-measure qui est aussi appelée f-score est un moyen de combiner precision et sensibilité de manière a pénaliser tout comportement abusif, et ce en se centrant sur la classe que l'on veut détecter, au contraire de la G-mean. C'est cette mesure que l'on utilisera dans la suite.

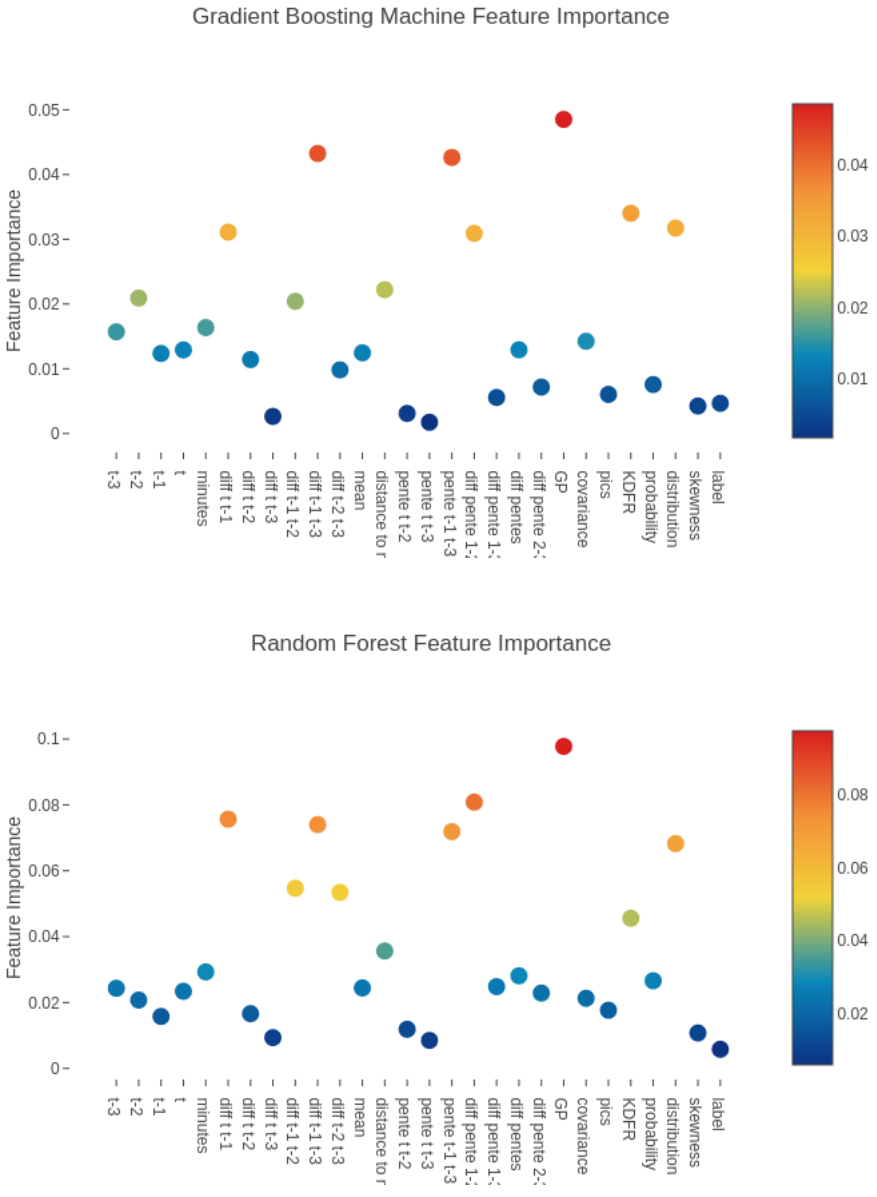
$$F-beta-measure = \frac{(1+\beta^2) * Recall * Precision}{\beta^2 * Recall + Precision} \tag{4.5}$$

Il ne faut cependant pas oublier que la série est temporelle. Une mauvaise classification peut ne pas être pénalisée si le point que l'on souhaitait détecter est son voisin. On pourra ainsi calculer le Mean Square Error (MSE) des minutes des évènements détectés. Cette prise en compte temporelle n'est pas encore implémenté et devra être plus réfléchie et définie avant d'être implémentée.

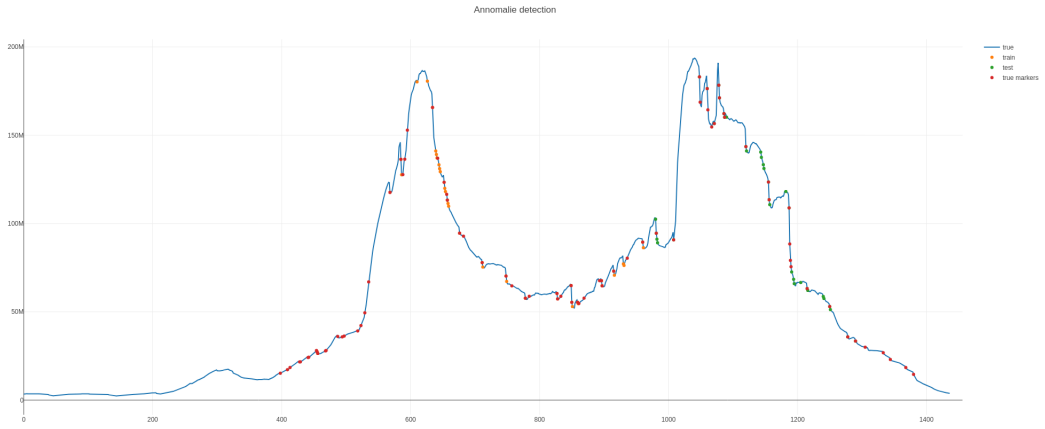
## 5 VISUALISATION DE DONNÉES

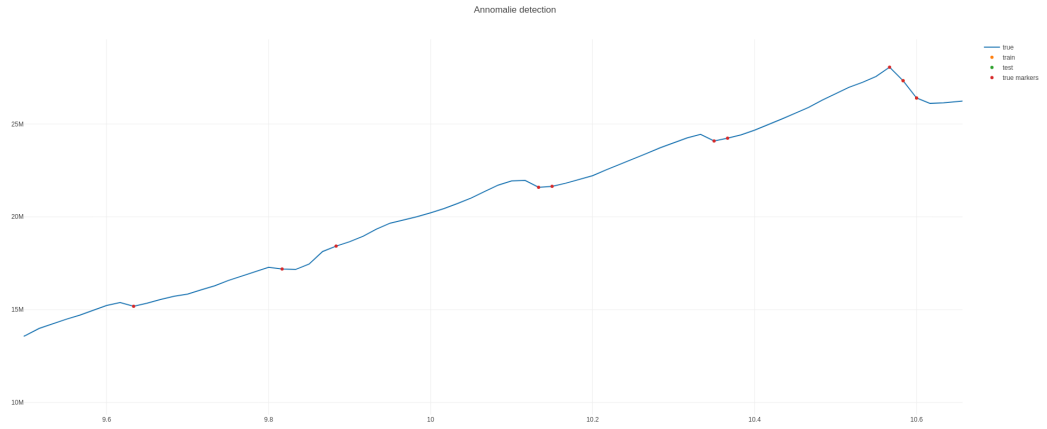
La visualisation de cette semaine est centrée sur les résultats des différents algorithmes de Machine learning testé cette semaine. On a entrainé les différents algorithmes sur une partie de l'historique puis on prédit l'ensemble de la journée afin de comparer les résultats. On affiche de plus les véritables événements tels qu'annotés par moi sur la journée du 30 avril et on affiche de plus ces points sur la courbe d'audience de TF1 pour le 30 avril afin de relier ces événements aux événements sur les courbes d'audience. D'autre journées ont aussi été soumises a l'algorithme afin d'essayer de voir s'il y avait des aberrations de détection. le dernier graphe est un zoom qui permet de voir qu'autant la détection du début des pubs/Fin des programmes est possible juste avec les courbes d'audience, autant la détection de la fin des pubs requiert plus de données afin d'identifier précisément les marqueurs de ces fins.


Annomalie detection
200M
150M
100M
50M
0
0
200
400
600
800
1000
1200
1400
true
train
test
true events


Gradient Boosting Machine Feature Importance
Feature Importance
0.05
0.04
0.03
0.02
0.01
0
t-3
t-2
t-1
t
minutes
diff t t-1
diff t t-2
diff t t-3
diff t-1 t-2
diff t-1 t-3
diff t-2 t-3
mean
distance to r
pente t-2
pente t-3
pente t-1 t-3
diff pente 1-
diff pentes
diff pente 2-
GP
covariance
pics
KDFR
probability
distribution
skewness
label
0.04
0.03
0.02
0.01


Random Forest Feature Importance
Feature Importance
0.1
0.08
0.06
0.04
0.02
0
t-3
t-2
t-1
t
minutes
diff t t-1
diff t t-2
diff t t-3
diff t-1 t-2
diff t-1 t-3
diff t-2 t-3
mean
distance to r
pente t-2
pente t-3
pente t-1 t-3
diff pente 1-
diff pentes
diff pente 2-
GP
covariance
pics
KDFR
probability
distribution
skewness
label
0.08
0.06
0.04
0.02


Annomalie detection
200M
150M
100M
50M
0
0
200
400
600
800
1000
1200
1400
true
train
test
true markers


Annomalie detection
true
train
test
true markers
25M
20M
15M
10M
9.6
9.8
10
10.2
10.4
10.6