

FACULTÉ DES SCIENCES ET INGÉNIERIE
SORBONNE UNIVERSITÉ

AUTO REGRESSIVE MODELS ON TRANSPORT DATA

Rapport de Projet Logiciel DAC

Auteur :
Ahmed Tidiane BALDE

Encadrants :
Vincent GUIGUE
Nicolas BASKIOTIS

3 juin 2019



Table des matières

1	Préambule	3
2	Etat de l'art	4
3	Données	5
3.1	Analyse de données	6
3.1.1	Stations	6
3.1.2	Outliers	6
3.1.3	Hypothèses	8
3.2	Pré-traitement	10
3.2.1	Suppression d'Outliers	10
3.2.2	Normalisation et Split	11
4	Modèles d'auto-régression pour les séries temporelles	12
4.1	Auto Regressive Model (<i>AR</i>)	12
4.1.1	Linear Regression	12
4.1.2	XGBoost Regressor	14
4.2	Auto Regressive Moving Average Model (<i>ARMA</i>)	14
4.3	Auto Regressive Integrated Moving Average Model (<i>ARIMA</i>)	15
4.4	Baselines	16
5	Métriques d'évaluation	17
6	Résultats d'expérience	17
6.1	Baselines Results	17
6.2	Un modèle général	18
6.2.1	Sans Baseline	18
6.2.1.1	AR	18
6.2.1.2	AR pour chaque T	21
6.2.1.3	ARMA	22
6.2.1.4	Comparaison	22
6.2.2	Avec Baseline par station	22
6.2.2.1	AR	22
6.2.2.2	AR pour chaque t	25
6.2.2.3	ARMA	25
6.2.2.4	Comparaison	26
6.2.3	Avec Baseline par station et par jour	26
6.2.3.1	AR	26
6.2.3.2	AR pour chaque t	29
6.2.3.3	ARMA	29
6.2.3.4	Comparaison	30
6.3	Un modèle par jour	30
6.3.1	AR avec <i>Baseline</i> par station et par jour	30
6.4	Un modèle par station	31

6.4.1	AR avec <i>Baseline</i> par station et par jour	31
6.5	Un modèle par station et par jour	31
6.5.1	AR avec <i>Baseline</i> par station et par jour	32
6.5.1.1	Comparaison	34
7	Apprentissage d'ensemble	35
7.1	Averaging Models	35
7.2	Staking Models	35
8	Conclusion	36
	Références	37

Introduction

1 Préambule

Ce projet consiste à appliquer des modèles d'auto-régression sur les données de transport de Paris. Il découle directement d'un stage que j'ai fait il y a de cela quelques mois au *LIP6* avec comme encadrants M. Vincent Guigue et M. Nicolas Baskiotis.

Ledit Stage en quelques mots, était axé sur l'application des modèles de Régression avancés tels que : *XGBoost*, *Random Forest* et bien d'autres. Il fallait donc, en plus des données de transport dont on disposait (informations sur les stations (id, nom, latitude et longitude, zone), sur l'ensemble des validations des usagers (date, temps, station_id, mode)), importer des données complémentaires du site de l'*INSEE* (Institut national de la statistique et des études économiques) concernant les habitants aux alentours de chaque station, ou en des termes concis, l'*IRIS* (Îlots Regroupés pour l'Information Statistique) auquel appartient une station, pour ainsi enrichir les données d'apprentissage.

Quant au projet, il s'agit cette fois de se baser exclusivement sur les données de validations dont nous disposons, les discréteriser en des intervalles définis, apprendre sur un certain nombre de tranches horaires et prédire le nombre de validations durant la tranche horaire suivante ou durant les t tranches horaires suivantes, t étant un paramètre que nous fixerons, d'où l'auto-régression.

Parmi les modèles auto-régressifs, nous pouvons citer, *AR*, *ARMA*, *ARIMA* et bien d'autres. Plusieurs cas sont à étudier pour chacun.

Pour commencer, nous implémenterons les modèles de sorte à pouvoir tester plusieurs autres modèles de Machine Learning en plus de la simple régression linéaire, tels que : *Ridge*, *ElasticNet*, *XGBoost* et *Random Forest*.

Ensuite, nous nous intéresserons à la spécification des modèles. Plutôt que de considérer un seul modèle pour toutes les données, nous pourrons apprendre :

- Un modèle par jour, du Lundi au Dimanche ;
- Un modèle par station de métro et enfin
- Un modèle par station et par jour.

Chaque cas de modèle ci-dessus sera comparé à sa *baseline* respective qui est la moyenne des données agrégées en fonction du modèle, le but étant de pouvoir faire mieux que cette dernière. A cet effet, nous utiliserons plusieurs métriques principalement la *RMSE* (Root Mean Square Error) pour comparer nos différents algorithmes.

Certaines *baselines* étant très fortes, du fait qu'il se passe pratiquement la même chose dans une station de métro un jour de la semaine donné, nous les soustrairons aux données de base pour essayer de mieux prédire les disparités.

Enfin, nous nous attaquerons à une autre alternative qui consiste à apprendre un modèle pour chaque tranche horaire à prédire. Pour aller plus loin, nous évoquerons des modèles d'Ensemble Learning.

Dans les sections qui suivent, nous aborderons le processus de traitement et d'analyse de données plus en détails, ainsi que chaque cas de modèles. Et surtout, nous présenterons les résultats des expériences que nous commenterons avec soin.

2 Etat de l'art

De multiples algorithmes sont en vogue pour l'étude des séries temporelles, parmi lesquelles Auto Regressive (*AR*), (*Wikipedia*, 2019a), Auto Regressive Moving Average (*ARMA*) (*Wikipedia*, 2019b), ainsi que sa variante pour les données non stationnaires Auto Regressive Integrate Moving Average (*ARIMA*), sont ceux que nous abordons dans ce projet de manière assez brève.

Toute fois, au vu de nos données, d'autres modèles pourraient avoir de meilleures performances notamment l'une des variantes du *ARIMA*, Seasonal Auto Regressive Integrated Moving Average (*SARIMA*) (*Milenkoviet al.*, 2018), en supposant peut être que les pics du matin et du soir représentent la saisonnalité du modèle.

Par ailleurs, pour prédire avec assez de précision plusieurs intervalles de temps après, les réseaux de neurones sont très probablement la meilleure option. Voir l'article lu dans le cadre de ce projet (*Guiguetet al.*, 2018), dont nous nous sommes longuement inspirés pour ce travail. Les Recurrents Neural Networks (*RNN*) utilisés dans ce dernier ou encore les (*LSTM*) (Long-short Term Memory) (*Zhaoet al.*, 2017) ont un atout remarquable pour de pareilles tâches car non seulement ils gardent en mémoire les données du passé et les réutilisent pour la prédiction des valeurs futures, mais ils prennent aussi en compte le contexte temporal ou spatial des données.

Analyse

3 Données

La prise en main de données fut facile étant donné que j'eus travaillé là dessus auparavant. Les données sont hébergées sur un des serveurs du *LIP6*, dans une base de données *MySQL*. Cette dernière est composée de plusieurs tables *SQL* dont deux (2) que nous présenterons ici, celles que nous avons principalement utilisées. Ce sont :

station	
	id
	network_code
	transporter_code
	station_code
	name
x	
y	
pricing_zone	
validations_count	

validation	
	user_id
	station_id
	mode
	operation_date
	time
	nature
	trip_portion_id

Un *n-uplet* de chacune des tables ci-dessus ressemble respectivement à :

Station Table

id	nc	tc	sc	name	x	y	pricing_zone	val_count
2	12	112	266	Mairie	634777.498849	6860011.06668	4	554

Validation Table

user_id	station_id	mode	operation_date	time	nature	tp_id
89787145276371350004	53389	3	2015-10-01	19 :46 :12	031	2

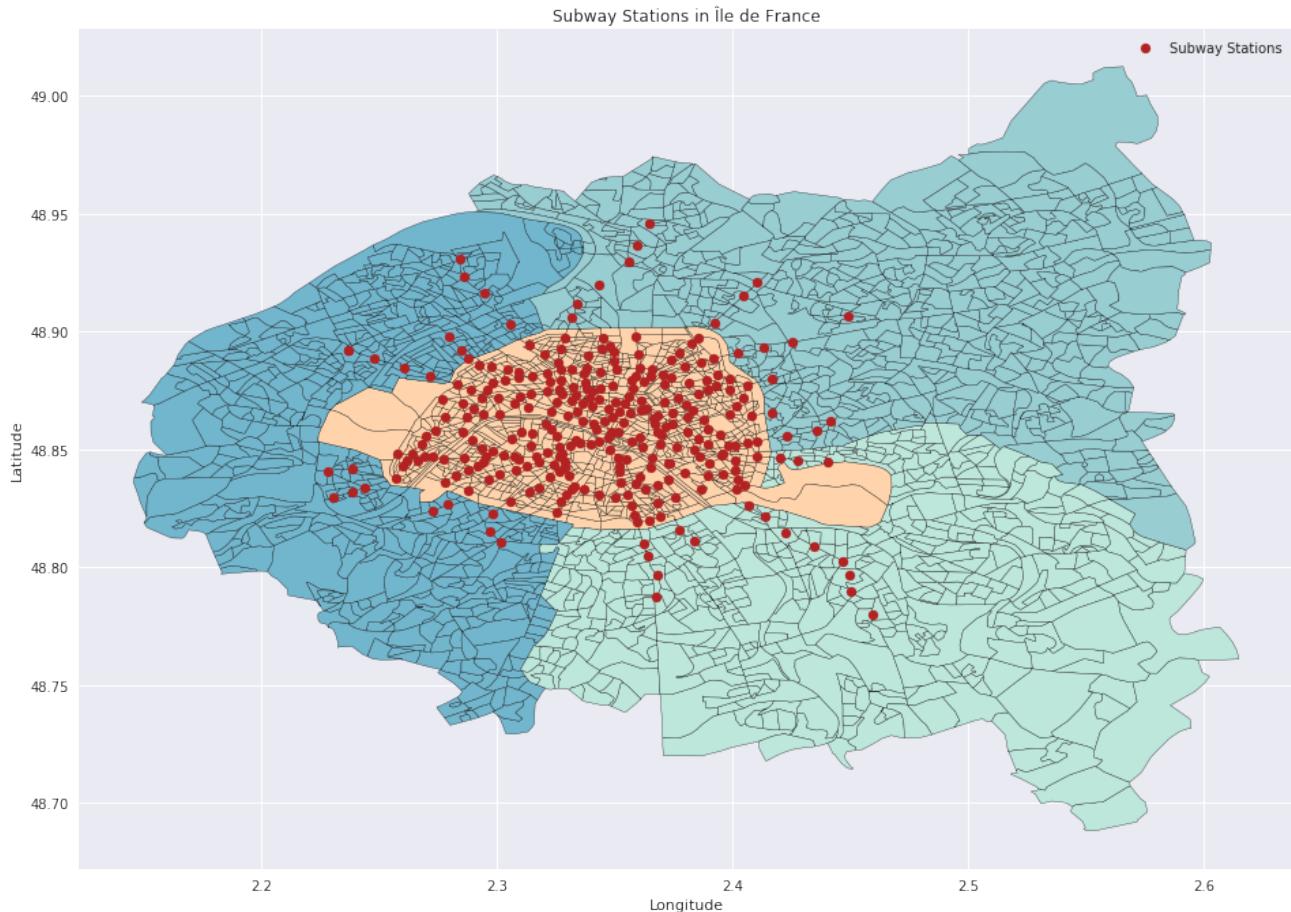
Cette dernière contient l'ensemble des données de validations des usagers sur le réseau de transport parisien pour une période de **3 mois** allant d'**Octobre à Décembre 2015**. Nous avons discréteisé les données en des intervalles de **15 minutes** ce qui nous fait **4** intervalles par heure et **96** par jour et par station, puis compté le nombre de validations. Nous obtenons ainsi des données en 3 dimensions (**Jours x Stations x Intervalles de Temps**) => **(92 x 303 x 96)**.

major	minor	2015-10-01	2015-10-02	2015-10-03	2015-10-04	2015-10-05	2015-10-06	2015-10-07	2015-10-08	2015-10-09	2015-10-10	...	2015-12-22	2015-12-23	2015-12-24	2015-12-25	2015-12-26	2015-12-27	2015-12-28	2015-12-29	2015-12-30	2015-12-31
198	00:00:00	38.0	70.0	57.0	26.0	26.0	39.0	46.0	53.0	46.0	70.0	...	26.0	20.0	21.0	12.0	35.0	11.0	11.0	27.0	29.0	0.0
	00:15:00	20.0	49.0	67.0	13.0	10.0	20.0	31.0	30.0	29.0	84.0	...	14.0	23.0	21.0	19.0	36.0	14.0	17.0	11.0	18.0	0.0
	00:30:00	13.0	39.0	48.0	18.0	4.0	4.0	9.0	26.0	33.0	50.0	...	13.0	5.0	2.0	11.0	22.0	8.0	13.0	36.0	12.0	0.0
	00:45:00	3.0	43.0	61.0	3.0	2.0	6.0	4.0	7.0	33.0	24.0	...	4.0	5.0	9.0	10.0	6.0	1.0	2.0	2.0	3.0	0.0
	01:00:00	1.0	23.0	48.0	0.0	0.0	2.0	3.0	1.0	25.0	25.0	...	5.0	2.0	5.0	1.0	9.0	2.0	1.0	2.0	2.0	0.0

3.1 Analyse de données

3.1.1 Stations

Disposant des coordonnées géographiques pour chacune des **303** stations, affichons les sur une carte.

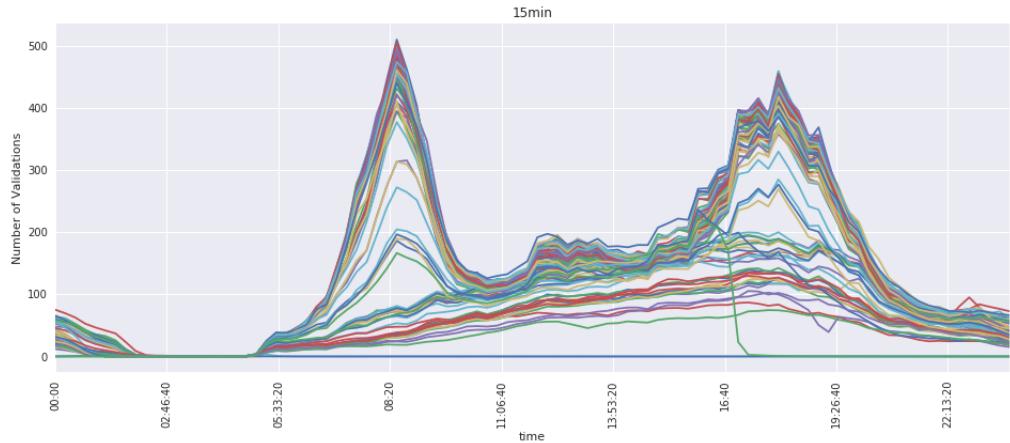


Rappelons que nous ne travaillons que sur le réseau de métro, qui s'étend exclusivement sur Paris et ses alentours, seulement **4** départements sont concernés notamment :

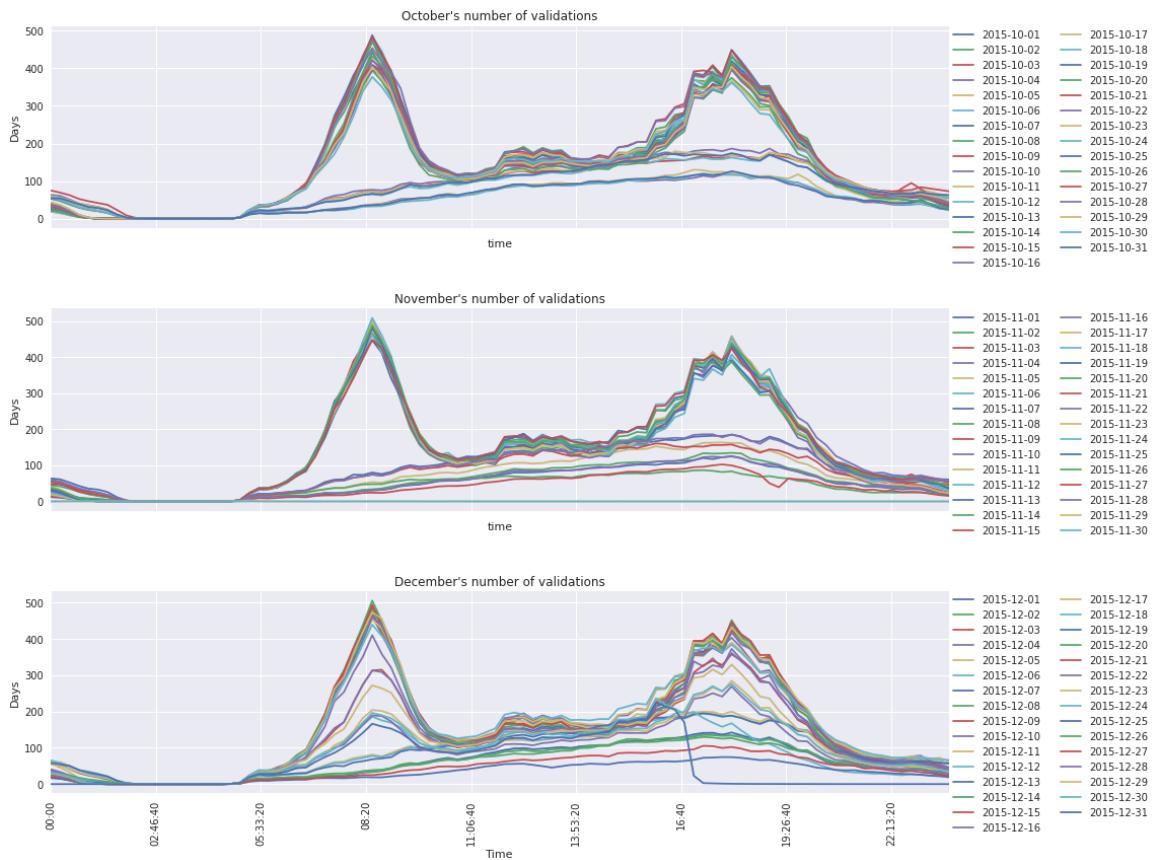
- **75**-Paris
- **92**-Hauts-de-Seine,
- **93**-Seine-Saint-Denis,
- **94**-Val-de-Marne.

3.1.2 Outliers

En moyennant le nombre de validations de toutes les stations pour chaque jour, nous obtenons la figure ci-dessous, les couleurs étant donc les différents jours dont nous disposons dans notre base de données. Nous remarquons tout d'abord qu'entre **01 :00 :00** et **04 :00 :00** du matin, il ne se passe rien, ce qui est normal, les stations sont fermées. Puis les deux pics du matin et du soir, exceptés certains jours qui correspondent naturellement aux *Week-ends*. Nous pouvons aussi distinguer des **outliers**, des jours dont la moyenne du nombre de validations de toutes les stations est à **0**. Ce doivent être des jours fériés où le transport est offert aux usagers. De même, nous pouvons observer des jours qui sont ni des *Week-ends*, ni des jours de semaine ordinaire, avec des mi-pics le matin et le soir.

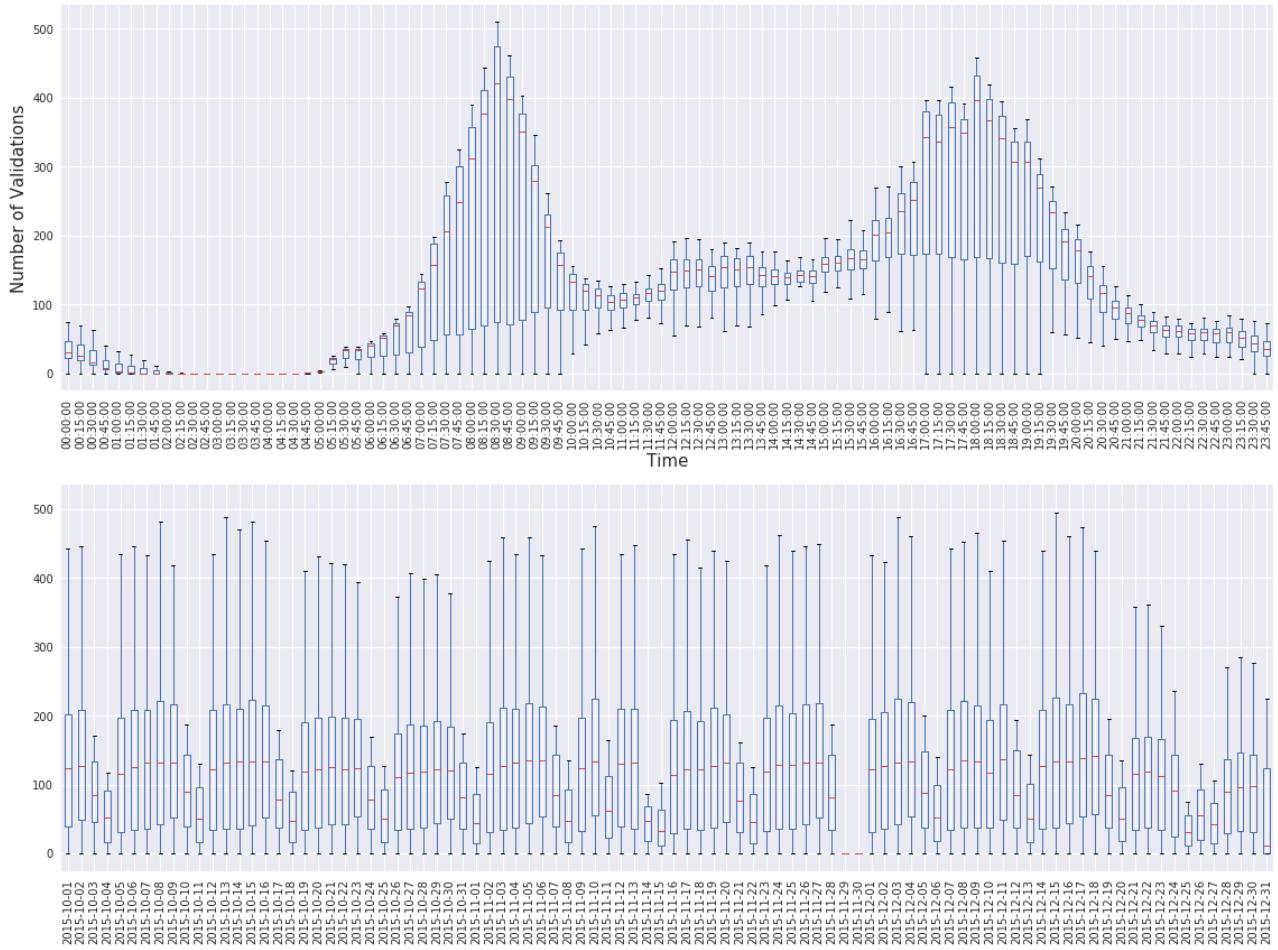


Essayons de voir un peu plus en détails ce qui se passe en réalisant la même étude sur chaque mois.



Le mois d'*Octobre* est plutôt normal, au vu des jours de semaine et des *Week-ends* que nous pouvons distinguer nettement. *Novembre* quant à lui abrite quelques **outliers** comme nous nous y attendions tels que l'**armistice** qui a lieu le **2015-11-11** ainsi que les journées du **2015-11-29** et **2015-11-30**, respectivement la veille et le début de la *COP21*. Quant au mois de *Décembre*, la journée du **2015-11-31** peut être considérée comme un **outlier**, le transport étant gratuit à partir de **16h**, aucune validation n'est répertoriée. De même, nous pouvons discerner les vacances de fin d'année, suite au nombre de validations très hétérogène en fonction des jours.

Sous un angle différent cette fois, voyons les boxplots des données.



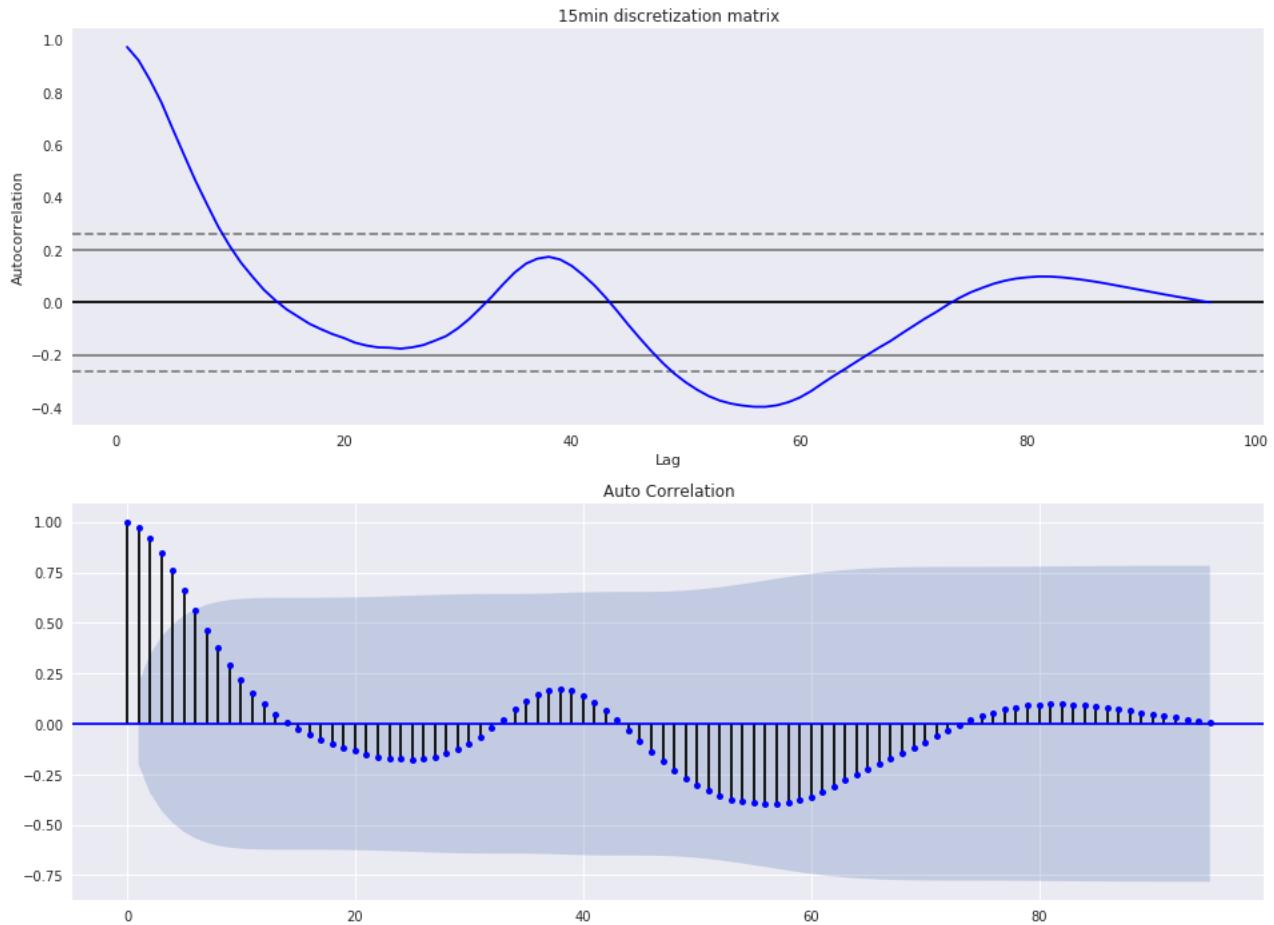
Les **boxplots** sont très étirés, les seconds **quartiles** très larges alors que la médiane est très haute, dû pas seulement aux **outliers** mais aussi aux *week-ends*. En plus des *whiskers* qui s'étendent jusqu'à 0, nous avons là des signes ostensibles d'un besoin de pré-traitement avant l'application d'un quelconque algorithme. La dernière figure nous permet quant à elle de voir l'alternance entre les jours de la semaine et le *Week-end*, et confirme en même temps nos dires en ce qui concerne les dates du 29 et 30 Novembre.

Dans la section dédiée au pré-traitement, nous allons donc retirer des données toutes les dates que nous considérons comme **outliers** dans le but d'avoir de meilleurs résultats, non influencés par des évènements quelconques.

3.1.3 Hypothèses

En attendant, il convient de souligner que nous avons fait des hypothèses selon lesquelles, les jours sont indépendants entre eux, c'est à dire que ce qui se passe un *Lundi* est indépendant de ce qui se passe un *Mardi* ou un quelconque autre jour de la semaine. À l'opposé, nous supposons que les évènements qui se passent dans une même journée sont dépendants. Par exemple, le nombre de validations qu'il pourrait y avoir dans une station et pour un jour donné à **01 :00 :00** du matin, ce qui correspond à l'intervalle **[00 :45 :00 - 01 :00 :00]**, dépend de ce qui s'est passé un quart d'heure avant à **00 :45 :00**, qui est corrélé à son tour à ce qui s'est passé le quart d'heure précédent et ainsi de suite.

Ci-dessous nous essayerons de vérifier ces hypothèses en affichant l'auto-corrélation entre les différents intervalles de temps.



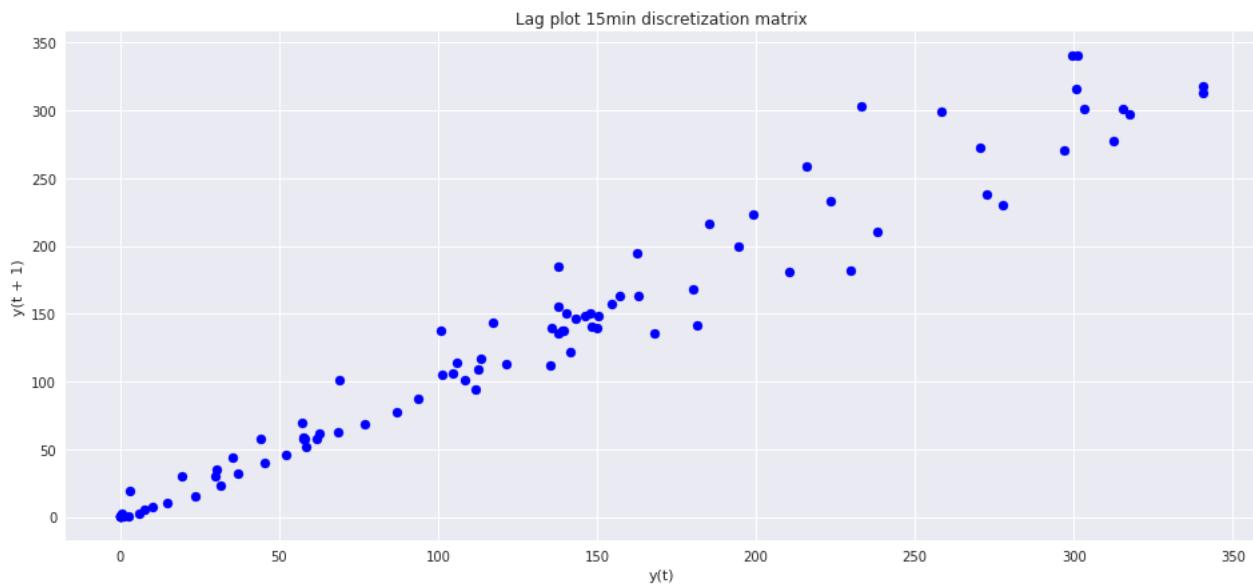
Les deux figures représentent toutes l'auto-corrélation entre les intervalles de temps. Le cône en bleu est la limite de l'intervalle de confiance fixé par défaut à 95%. Lorsque les barres s'élèvent en dehors, elles sont considérées comme des preuves d'auto-corrélation, et celles intégralement couvertes par le cône comme des preuves contre l'auto-corrélation.

L'axe x représente les **96** intervalles de temps et y le coefficient d'auto-corrélation toujours compris entre -1 et 1 .

La première barre à 0 aura toujours un coefficient d'auto-corrélation de 1 car elle exprime la corrélation de chaque intervalle de temps à lui-même. La seconde, est la corrélation entre chaque intervalle de temps et celui qui le précède. La troisième évoque la corrélation entre chaque intervalle de temps et le second qui le précède, ainsi de suite. La cinquième *5ime* variable de temps étant la dernière à se retrouver au dessus du cône, nous pouvons ainsi conclure que chaque intervalle de temps est majoritairement corrélé à celui qui le précède et que cette corrélation diminue au fur à mesure et voir même disparaît complètement au bout du cinquième *5ime* intervalle de temps précédent.

Rappelons que, dans la plupart des expériences menées, les figures ayant pour axe x les intervalles de temps correspondent à la moyenne de toutes les stations et également de tous les jours dont nous disposons quand rien n'est précisé.

Nous avons mené une dernière étude en affichant chaque intervalle de temps contre celui qui le précède.



Une forme linéaire comme nous pouvons l'observer nous suggère tout simplement qu'un modèle auto-régressif est probablement le meilleur choix et que nos données ne suivent pas une distribution aléatoire.

3.2 Pré-traitement

3.2.1 Suppression d'Outliers

Suite aux différentes expériences ci-dessus, nous avons éliminé les **outliers** les jours fériés comme l'aristice, ceux correspondants au début de la COP21 ainsi que toutes les vacances de décembre. Nous nous retrouvons avec une matrice 3D de dimensions **(78 x 303 x 96)**. Au total, 14 jours ont été retirés.

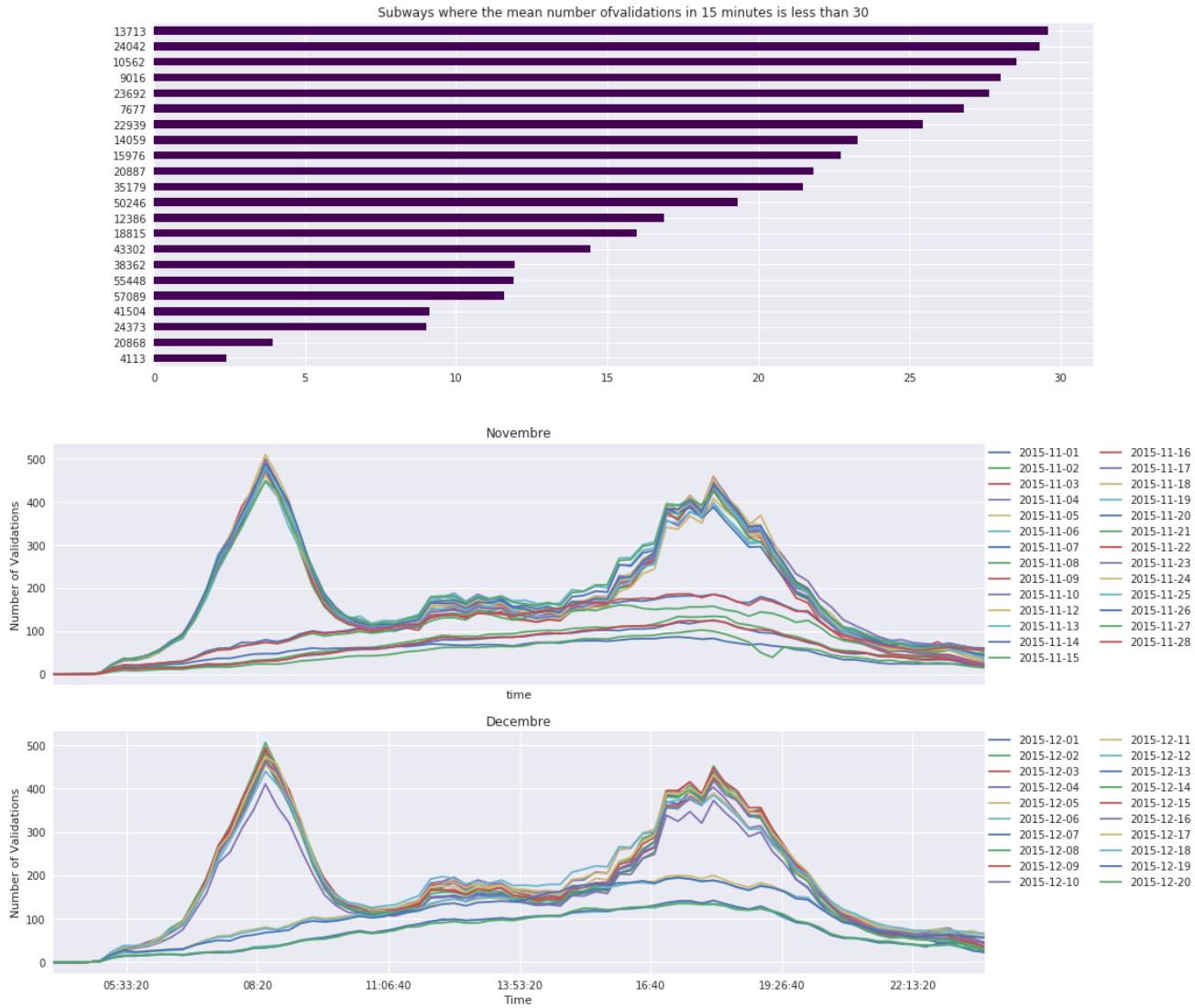
Les données entre **00 :00 :00** et **04 :00 :00** ont également été supprimées, au vu des figures que nous avons étudiées. Et comme nous l'avons évoqué ci-haut, il ne se passe rien de particulier durant cette tranche horaire. Au risque d'avoir un modèle qui diverge lors de la prédiction (multiplication des *weights* par 0), nous choisissons donc de les supprimer, ce qui nous laisse avec des données de dimensions **(78 x 303 x 80)**.

En nous intéressant aux stations, nous avons effectué une analyse sur celles avec un nombre de validations négligeable.

Il y a deux (2) stations dont la moyenne du nombre de validations pour un intervalle de temps est inférieur à 5, d'**id** respectifs 4113 et 20868. Elles ont été aussitôt retirées. Nous resterons alors sur des données de dimensions **(78, 301, 80)**.

Ci-dessus, nos nouvelles données des mois de *Novembre* et *Décembre*. Elles se confondent beaucoup plus au mois d'*Octobre* qui, quant à lui, ne contenait aucun jour qui sorte de l'ordinaire.

Et pour finir, nous avons affiché l'ensemble des données pré-traitées, composées de 78 jours, 301 stations et 80 intervalles de temps.

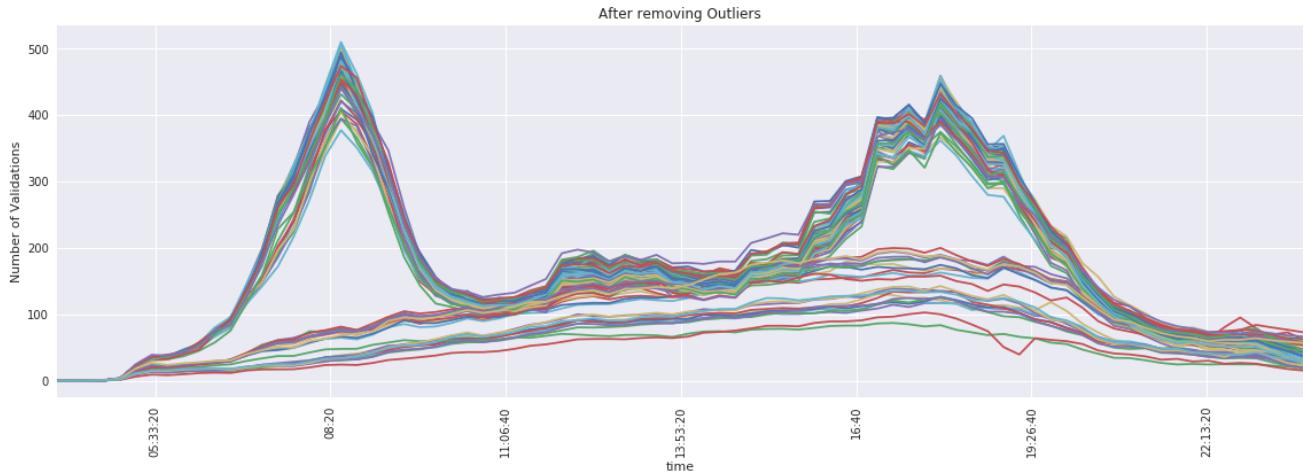


3.2.2 Normalisation et Split

Au vu des algorithmes que nous appliquerons tels que la régression linéaire, il est important de normaliser les données pour éviter d'avoir des *weights* biaisés par les stations en plein *Paris* avec un nombre de validations assez élevé, ou par les pics que nous observons à certaines heures de la journée. C'est ainsi que nous avons choisi d'appliquer une normalisation **MinMax** entre -1 et 1 , en agrégeant les données par station.

Par ailleurs, nous avons partitionné les données en **Train** et en **Test** en prenant soin de créer un échantillonnage stratifié, c'est à dire assez équilibré par rapport aux jours de la semaine.

Le jeu de données d'apprentissage *Train* est composé d'environ 8 semaines, soit 57 jours exactement et celui de *Test* de 3 semaines, soit 21 jours avec autant de *Lundi* que de *Mardi*, etc...



4 Modèles d'auto-régression pour les séries temporelles

4.1 Auto Regressive Model (AR)

4.1.1 Linear Regression

Un modèle AR prédit les tendances actuelles ou futures en fonction de celles du passé. La notation $AR(p)$ où p correspond à l'ordre est définie comme suit :

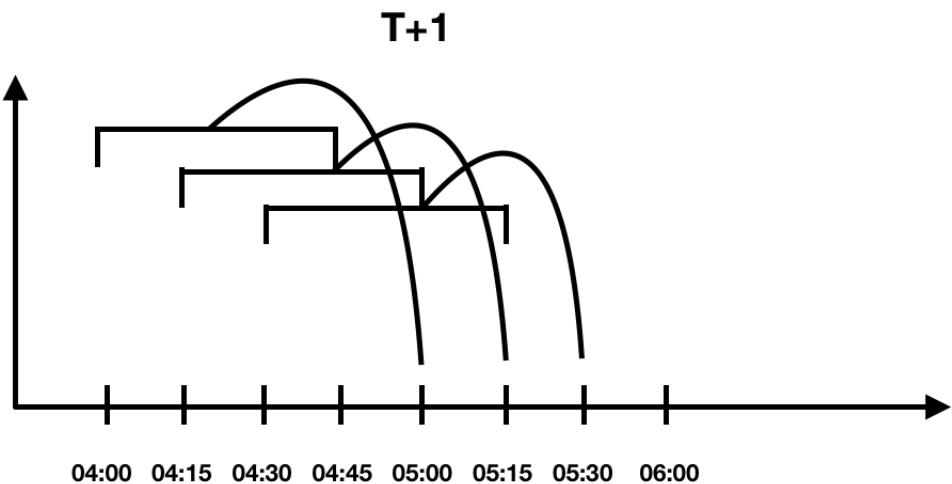
$$X_t = c + \sum_{i=1}^p \theta_i X_{t-i} + \varepsilon_t$$

où $\theta_1, \dots, \theta_p$ sont les paramètres du modèle, c une constante et ε_t le bruit blanc,

(Wikipedia, 2019a)

L'ordre p est le nombre de valeurs du passé à prendre en compte, en l'occurrence, pour p égal à 4, nous commencerons par prédire le 5^{ime} intervalle de temps. À partir de là, pour chaque intervalle de temps à prédire nous prendrons les 4 intervalles de temps précédents. La figure suivante nous sert d'illustration. Elle représente la prédiction à T+1. Pour prédire le nombre de validations à 05 :00 :00, nous utilisons les 4 intervalles de temps précédents et ainsi de suite, en supposant que p est fixé à 4.

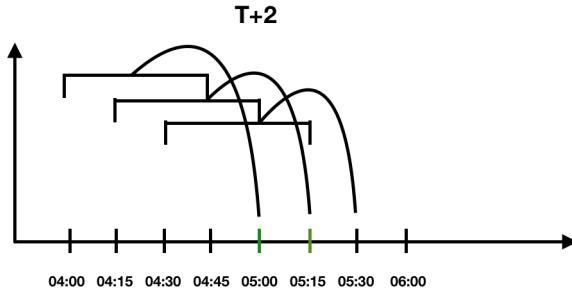
Sous un autre angle, notre jeu de données d'apprentissage ressemblerait à la figure suivante :



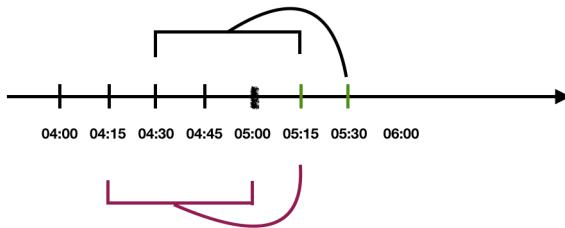
X	Y
04:00 04:15 04:30 04:45	05:00
.	.
04:15 04:30 04:45 05:00	05:15
.	.
04:30 04:45 05:00 05:15	05:30
.	.
04:45 05:00 05:15 05:30	05:45
.	.
05:00 05:15 05:30 05:45	06:00
.	.
.	.

Maintenant, au delà de **T+1**, il serait bien de pouvoir prédire à **T+n**, **n** étant le $n - i$ me intervalle après la tranche horaire composée des p intervalles utilisés.

Pour **T+2** par exemple, il s'agit d'utiliser certaines des valeurs prédites à $T+1$, puis pour **T+3**, il s'agira donc d'utiliser non seulement certaines valeurs prédites à $T+1$, mais aussi à $T+2$, ainsi de suite. La représentation graphique de cette procédure est la suivante :



Pour prédire à 05:30, on réinsère la valeur réelle de 05:00.



Après avoir réinsérer la valeur réelle de 05:00, on re-prédict l'affluence à 05:15, puis nous utilisons cette dernière pour la prédiction à 05:30.

Sous forme matricielle :

La partie à réaliser avec parcimonie est certes le remplacement des valeurs réelles au bon endroit ainsi que la prédiction qui intervient avant celle de **T+n** voulu.

Notons que pour éviter de recalculer certaines prédictions, nous utilisons une des techniques d'optimisation de code : la mémorisation.

Une autre alternative consiste à ne pas utiliser les valeurs prédites, et, pour chaque intervalle de temps à prédire, apprendre un modèle spécifique, comme ci-dessous :

4.1.2 XGBoost Regressor

Comme nous l'avons vu ci-haut, la simple théorie derrière les modèles d'auto-régression que nous avons étudiée et que nous allons étudier par la suite est la *régression linéaire*.

Tous les algorithmes de régression pourraient donc s'y adapter sans pour autant changer notre implémentation. C'est pourquoi nous avons pris la liberté de tester certains algorithmes comme le XGBoostRegressor ou encore le RandomForestRegressor.

4.2 Auto Regressive Moving Average Model (ARMA)

ARMA(p, q) où p est le paramètre auto-régressif et q celui du *Moving Average(MA)*, est défini par :

$$X_t = c + \sum_{i=1}^p \theta_i X_{t-i} + \varepsilon_t + \sum_{i=1}^q \beta_i \varepsilon_{t-i}$$

X	Y
04:00 04:15 04:30 04:45	05:00
.	.
04:15 04:30 04:45 05:00	05:15
.	.
04:30 04:45 05:00 05:15	05:30
.	.
04:45 05:00 05:15 05:30	05:45
.	.
05:00 05:15 05:30 05:45	06:00
.	.

↑

Pour prédire à T+2, prédire d'abord à T+1, reshape la matrice sous la forme de X et remplacer cette colonne par

avec $\epsilon_1, \dots, \epsilon_q$, les erreurs,

(Wikipedia, 2019b)

Sous forme matricielle, en prenant p=4 et q=1, nos jeux de données en apprentissage seront similaires à la représentation ci-dessous.

Quant aux données de *Test*, pour que nous puissions appliquer l'algorithme en inférence, il faut supposer que nous ne disposons pas des données réelles. Par conséquent nous ne pouvons pas calculer l'erreur. Nous avons ajouté des 0 comme ci-dessous de façon à ce que nos données en Test soient conformes à celles en Train pour ainsi pouvoir appliquer les algorithmes.

4.3 Auto Regressive Integrated Moving Average Model (ARIMA)

Le modèle ARIMA est une variante de ARMA qui s'applique à des données non stationnaires. Il consiste à intégrer les données un nombre d de fois de façon à supprimer la non-stationnarité pour ainsi revenir au cadre de ARMA. Il se note $ARIMA(p, d, q)$, avec p et q les mêmes paramètres que le modèle précédent, respectivement le paramètre auto-régressif et celui du *Moving Average(MA)*. Et pour finir, d le degré de différenciation. L'équation correspond exactement à celle de ARMA vu ci-haut :

$$X_t = c + \sum_{i=1}^p \theta_i X_{t-i} + \epsilon_t + \sum_{i=1}^q \beta_i \epsilon_{t-i}$$

Nos données étant déjà stationnaires, nous allons principalement nous concentrer sur les deux premiers modèles décrits, AR et ARMA.

T+1					T+2					T+3								
X		Y			X		Y			X		Y						
04:00	04:15	04:30	04:45		05:00		04:00	04:15	04:30	04:45		05:15		04:00	04:15	04:30	04:45	05:30
.
04:15	04:30	04:45	05:00		05:15		04:15	04:30	04:45	05:00		05:30		04:15	04:30	04:45	05:00	05:45
.
04:30	04:45	05:00	05:15		05:30		04:30	04:45	05:00	05:15		05:45		04:30	04:45	05:00	05:15	06:00
.
04:45	05:00	05:15	05:30		05:45		04:45	05:00	05:15	05:30		06:00		04:45	05:00	05:15	05:30	06:15
.
05:00	05:15	05:30	05:45		06:00		05:00	05:15	05:30	05:45		06:15		05:00	05:15	05:30	05:45	06:30
.
X_train					Y_train					X_test								
X1	X2	X3	X4	eps1	Y_train					X1	X2	X3	X4	eps1				
04:00	04:15	04:30	04:45	f(X) - Y	05:00					04:00	04:15	04:30	04:45	0				
.	0				
04:15	04:30	04:45	05:00		05:15					04:15	04:30	04:45	05:00	0				
.	0				
04:30	04:45	05:00	05:15		05:30					04:30	04:45	05:00	05:15	0				
.	0				
04:45	05:00	05:15	05:30		05:45					04:45	05:00	05:15	05:30	0				
.	0				
05:00	05:15	05:30	05:45		06:00					05:00	05:15	05:30	05:45	0				
.	0				

4.4 Baselines

Les *baselines* sont les moyennes des données agrégées selon différents axes choisis. Nous pouvons en énumérer 4 comme indiqué dans l'introduction.

- Baseline Générale : La moyenne de toutes les données sur l'axe des intervalles de temps.
- Baseline par jour : Pour chaque jour de la semaine, la moyenne des données correspondantes sur l'axe des intervalles de temps.
- Baseline par station : Pour chaque station, la moyenne des données correspondantes sur l'axe des intervalles de temps.
- Baseline par station et par jour : Pour chaque station, pour chaque jour de la semaine, la moyenne des données correspondantes sur l'axe des intervalles de temps.

Nous rappelons que certaines de ces *baselines* sont assez fortes notamment les deux dernières, l'idée étant donc de faire mieux mais aussi de voir à partir de quelle prédiction $T+$ la courbe d'erreur rencontre la *baseline*. Dans la section *résultats d'expérience*, nous prendrons soin d'afficher les scores de chacune d'entre elles et générerons par la même occasion le résultat graphique.

5 Métriques d'évaluation

Plusieurs métriques furent utilisées, chacune nous apportant une information différente sur nos prédictions et sur leurs principales différences avec nos données. Les unes plus robustes aux outliers que d'autres, soit :

- **RMSE** : est celle que nous avons principalement utilisée pour comparer nos différents modèles.
- **MAPE** : est en pourcentage, donc naturellement plus parlante. Nous l'avons utilisée pour créer des clusters de stations sur la carte ; nous aborderons ce sujet plus en détails dans la section des expériences.
- **MPE** : nous permet d'avoir une idée assez claire de si nos modèles sur-estiment ou sous-estiment les prédictions par rapport aux valeurs réelles.

6 Résultats d'expérience

Comment choisir le paramètre auto-régressif p ?

Nous l'avons vu dans la sous-section *Analyse de données*, en affichant le graphe d'*Auto-corrélation (ACF)* avec un intervalle de confiance de 95%, le 5^{ème} intervalle de temps est englouti par le cône, une preuve *contre* l'auto-corrélation. Un ordre p fixé à 4 serait donc théoriquement bien.

Notons que plus l'ordre p est grand, moins nous avons de données et plus l'erreur sera élevée en théorie. Cependant, cette hypothèse n'est pas totalement justifiée, car si p est suffisamment grand de façon à ce que le pic du matin soit compris dans les données en entrée, nous aurons dans ce cas une erreur plus faible puisque les pics font l'objet d'une erreur de prédiction plus importante. Dans les expériences qui suivent, nous fixerons **p** à 4 et **tplus** à 8. Donc, à partir de chaque tranche horaire d'une (1) heure sur l'ensemble des intervalles de temps, nous essayerons de prédire l'affluence pour les (2) heures qui vont suivre.

6.1 Baselines Results

	None	s	j	sj
R2	0.073173	0.813549	0.104357	0.979260
RMSE	331.811381	148.824483	326.181518	49.636328
MSE	110098.792486	22148.726859	106394.382613	2463.765091
MAE	2.285234	0.581370	1.656471	0.205513
MAPE	228.523427	58.137036	165.647083	20.551330
MPE	-207.777042	-36.689913	-145.424580	-6.026211

FIGURE 1 – Scores for each Baseline

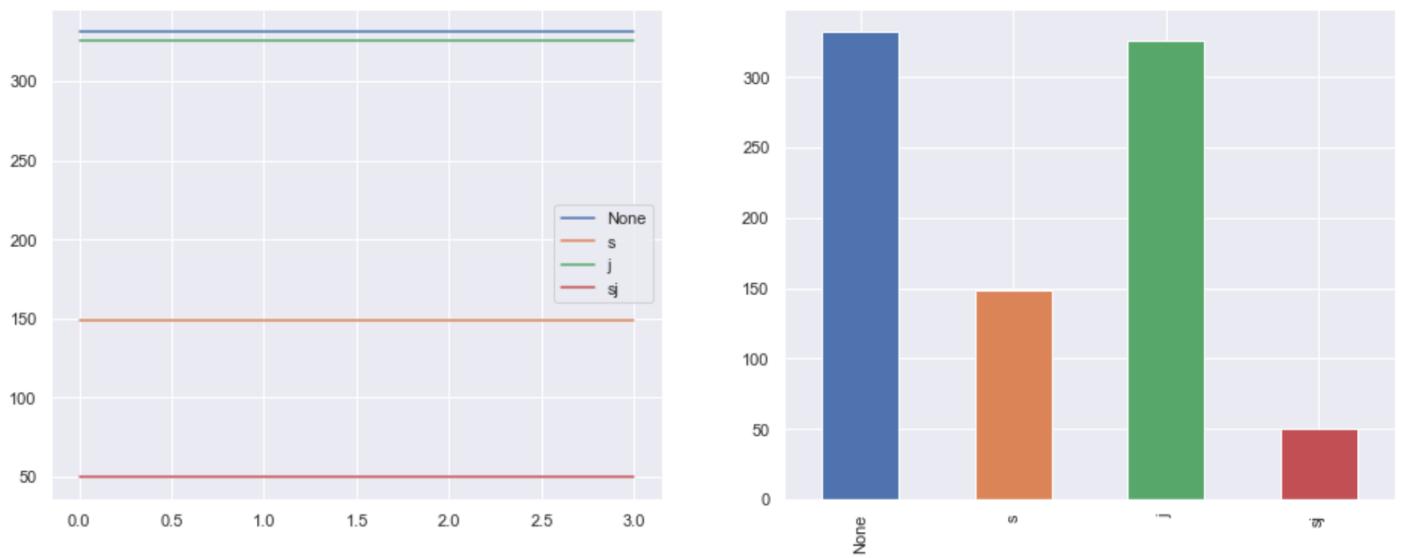


FIGURE 2 – Plot of Baselines : *None* is the General Model

6.2 Un modèle général

Ordre $p = 4$

Tplus = 8

6.2.1 Sans Baseline

Dans ces premières expériences, nous utiliserons de temps à autre le modèle *XGBoost* car il est plus performant que la *régression linéaire* sur les modèles assez génériques comme celui-ci.

6.2.1.1 AR

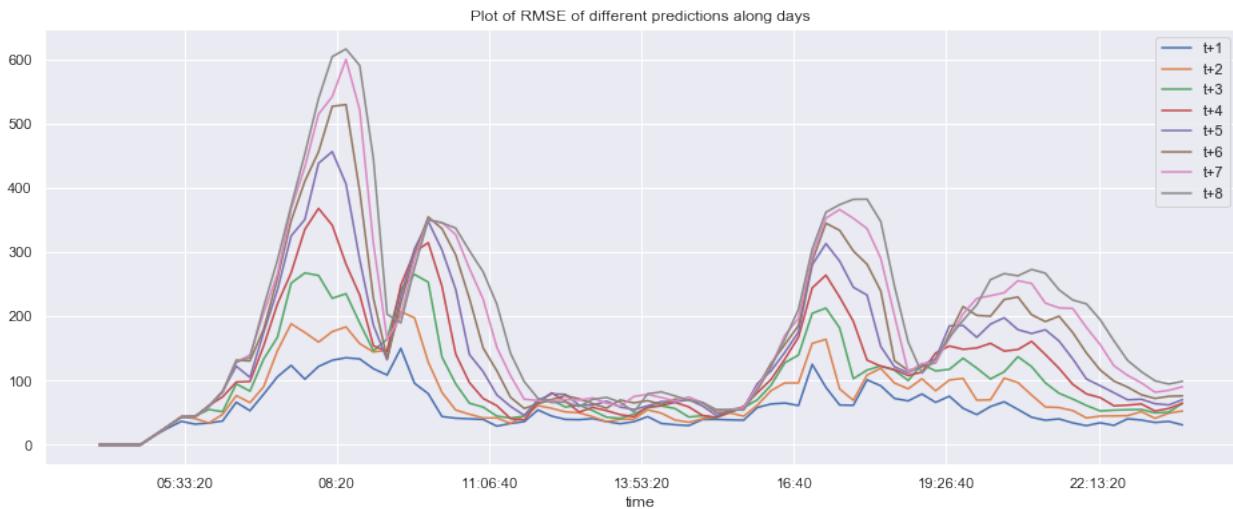
	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.957393	0.912012	0.840469	0.763957	0.680103	0.609441	0.551512	0.508770
RMSE	72.639629	104.386118	140.557523	170.972983	199.038227	219.925016	235.671372	246.645843
MSE	5276.515635	10896.461680	19756.417302	29231.760810	39616.215643	48367.012702	55540.995802	60834.171852
MAE	0.373669	0.537392	0.704030	0.849625	0.987059	1.098535	1.192554	1.268383
MAPE	37.366882	53.739249	70.403020	84.962474	98.705902	109.853491	119.255375	126.838301
MPE	-22.974751	-37.104618	-51.895249	-64.672558	-76.466775	-86.100682	-94.168351	-100.652354

Scores for General Model without Baseline : Linear Regression

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.963014	0.929514	0.876291	0.814749	0.736219	0.656289	0.576771	0.503640
RMSE	67.678236	93.429182	123.774733	151.464694	180.739719	206.313772	228.938627	247.930422
MSE	4580.343675	8729.012111	15320.184427	22941.553384	32666.846081	42565.372355	52412.895058	61469.494058
MAE	0.306233	0.407537	0.499870	0.579448	0.672298	0.752247	0.833474	0.907843
MAPE	30.623298	40.753735	49.986971	57.944809	67.229761	75.224706	83.347413	90.784310
MPE	-15.320157	-23.178730	-29.805488	-35.414554	-42.287482	-48.331630	-54.784127	-60.898352

Scores for General Model without Baseline : XGBoost

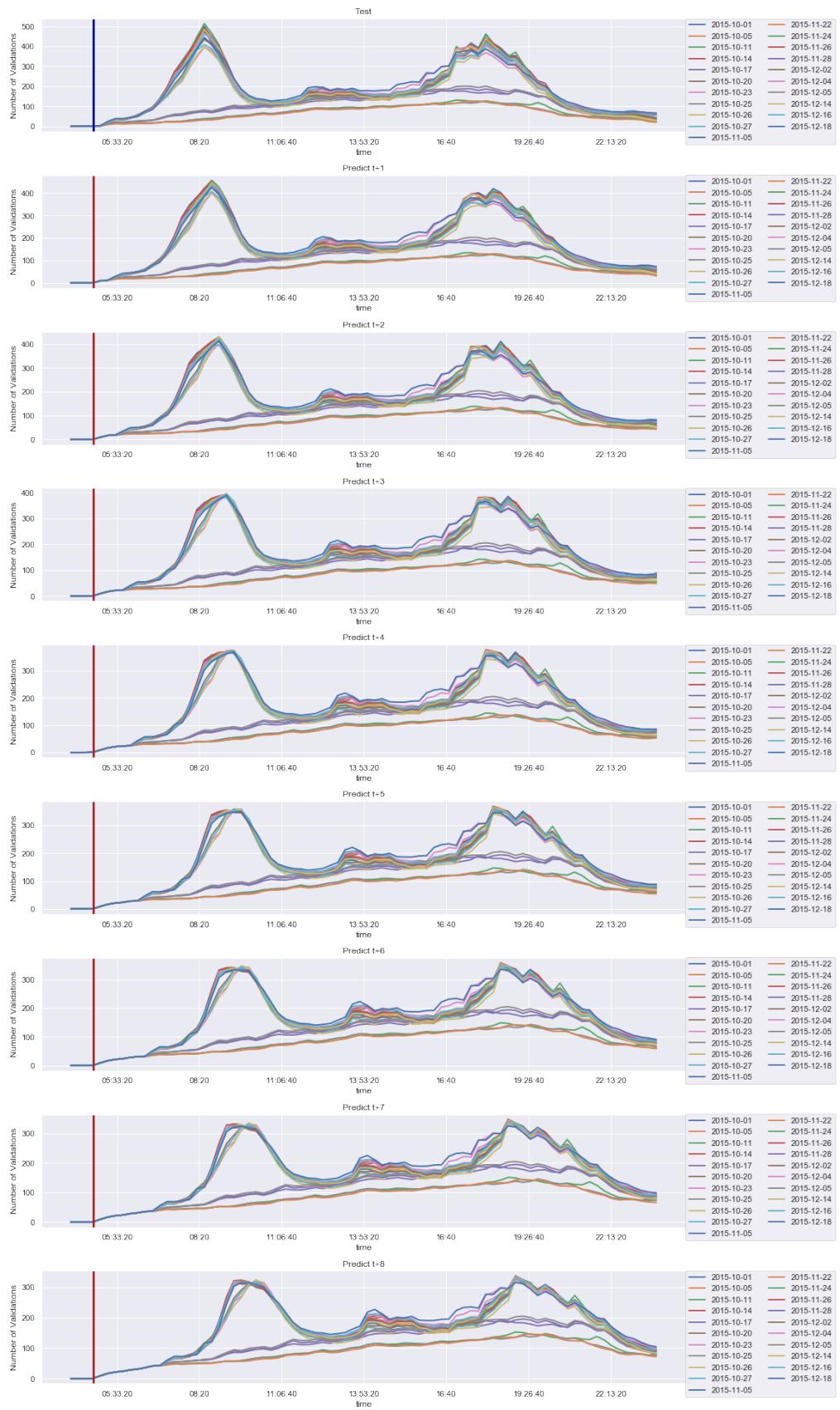
Le *XGBoost* est largement meilleur, excepté à $T + 8$ où les deux modèles sont presque égaux.



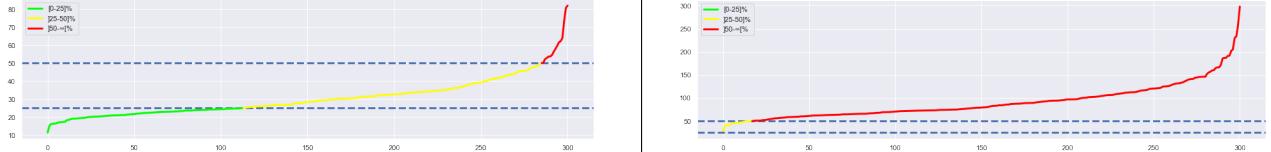
RMSE Score error for General Model without Baseline : XGBoost

Les couleurs représentent les résiduels pour chaque T . Naturellement, à $T+1$ nous avons moins d'erreurs qu'à $T+2$, et ainsi de suite. Toutefois, notons que les erreurs sont corrélées au pic de la journée. Nous observons à cet effet plus d'erreurs vers 8h20 et 18h.

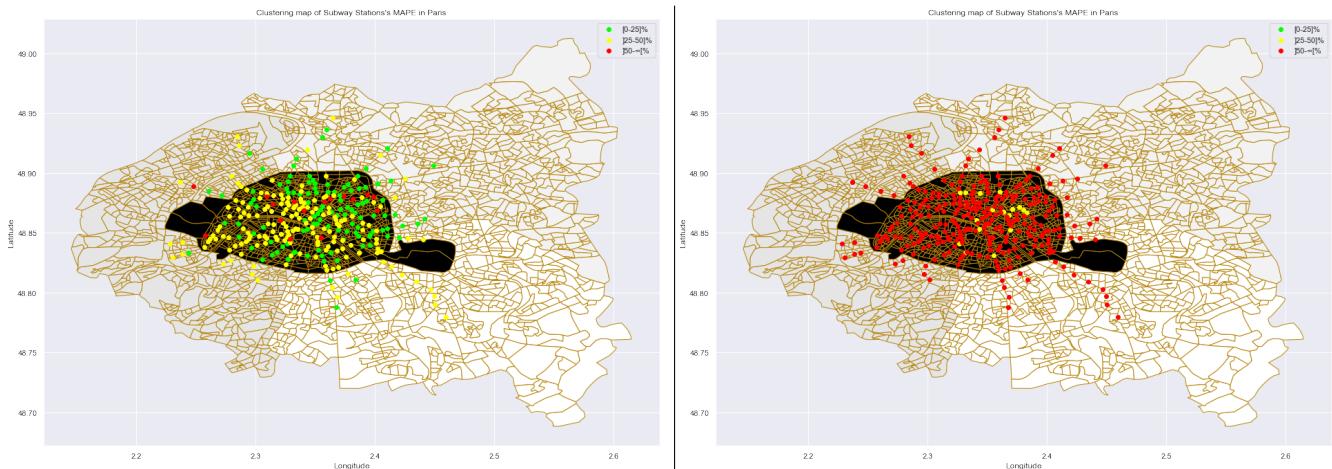
Ci-dessous, la moyenne des prédictions projetées sur l'axe des intervalles de temps, nous discernons de manière assez nette le décalage qui s'accentue au fur et à mesure que nous augmentons en T .



Prédictions des différents T



MAPE curves of stations



MAPE clusters map of stations

Ci-dessus nous avons créé différents intervalles pour pouvoir faire des *clusters* des stations que nous arrivons à mieux prédire et, à contrario, celles où nous avons du mal :

- En vert : Les stations avec une *MAPE* entre $]0-25\%]$;
- En jaune : celles dont la *MAPE* est comprise entre $]25-50\%]$;
- Et en rouge : *MAPE* comprise entre $]50-\infty]$ %.

Les premières figures à gauche correspondent au résultat à $T+1$, donc théoriquement les meilleurs résultats pour ce cas-ci, et, à droite, les résultats à $T+8$, où nous avons beaucoup plus d'erreurs.

6.2.1.2 AR pour chaque T

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.957393	0.916229	0.854215	0.789638	0.708007	0.643045	0.589737	0.553534
RMSE	72.639629	102.401901	135.848539	164.116513	194.459707	216.230404	233.108853	244.401499
MSE	5276.515635	10486.149289	18454.825598	26934.229820	37814.577462	46755.587814	54339.737142	59732.092738
MAE	0.373669	0.543406	0.703511	0.851749	1.003051	1.112539	1.201803	1.260886
MAPE	37.366882	54.340645	70.351115	85.174913	100.305122	111.253919	120.180259	126.088637
MPE	-22.974751	-37.568456	-51.261019	-64.115341	-77.775969	-87.358835	-95.136449	-100.142482

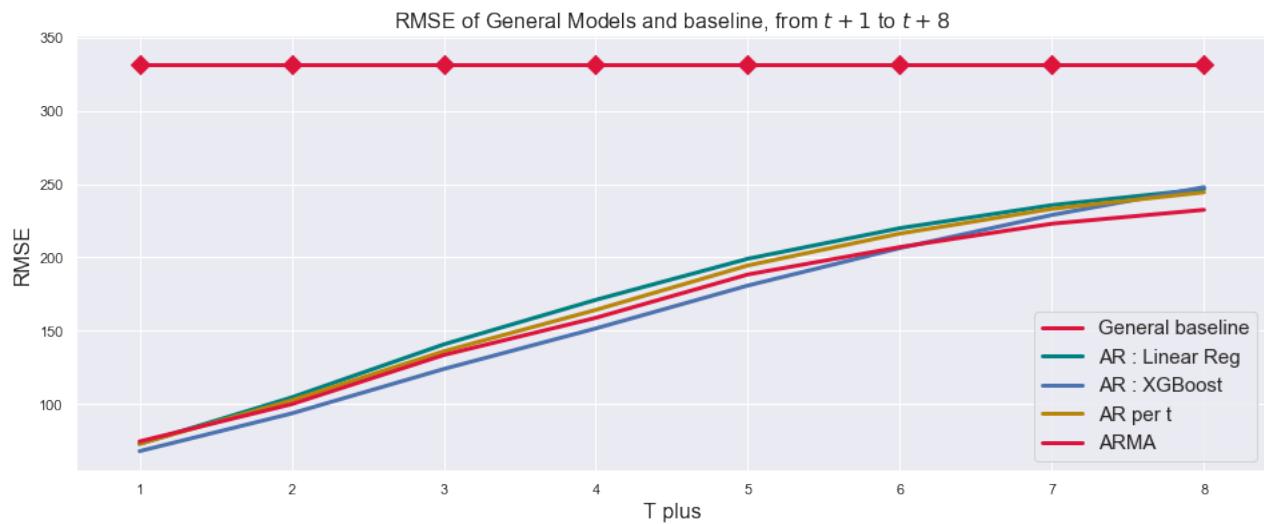
Scores for General Model AR for each T without Baseline : Linear Regression

6.2.1.3 ARMA

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.955394	0.919684	0.856527	0.796593	0.713698	0.653945	0.598542	0.563708
RMSE	74.323906	99.731486	133.295688	158.713523	188.297222	207.016159	222.972642	232.445086
MSE	5524.042938	9946.369276	17767.740385	25189.982510	35455.843913	42855.690170	49716.798910	54030.718025
MAE	0.411724	0.572886	0.789080	0.942650	1.124558	1.242049	1.357351	1.429804
MAPE	41.172376	57.288627	78.907967	94.265048	112.455847	124.204907	135.735055	142.980404
MPE	-23.388812	-37.442148	-57.166433	-71.046445	-87.575854	-98.303595	-108.883177	-115.457229

Scores for General Model ARMA without Baseline : Linear Regression

6.2.1.4 Comparaison



Le modèle *XGBoost* est meilleur que les autres sur ce cas-ci, ce qui est normal. Mais si nous comparons les modèles de *régressions linéaires*, le modèle AR pour chaque intervalle de temps à l'aire de se démarquer légèrement.

6.2.2 Avec Baseline par station

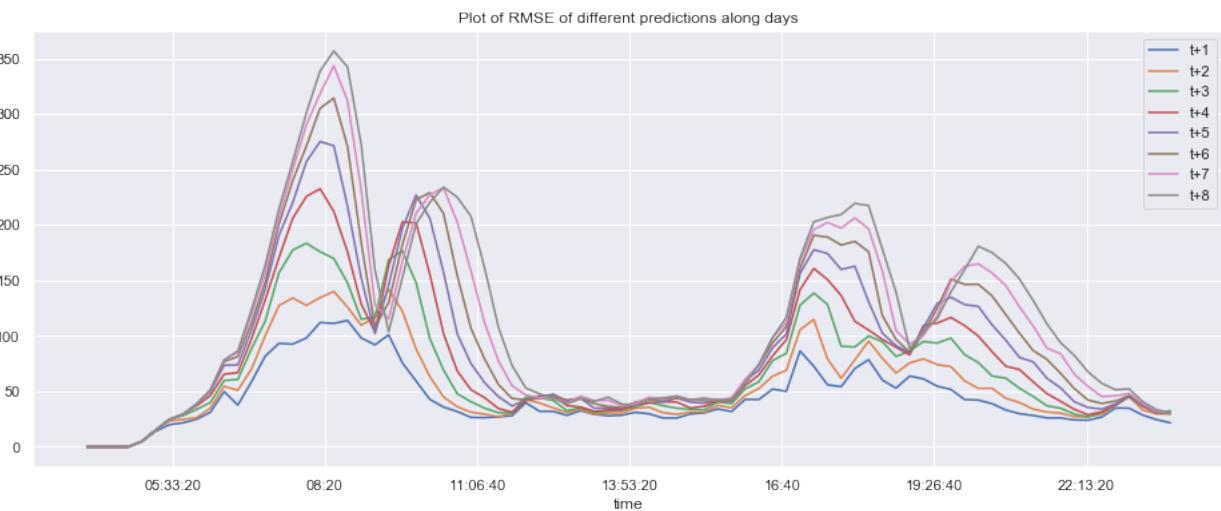
6.2.2.1 AR

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.973638	0.955606	0.930002	0.903435	0.875810	0.851603	0.830712	0.813808
RMSE	57.137753	74.146821	93.105080	109.355870	124.015189	135.563722	144.791823	151.849168
MSE	3264.722817	5497.751038	8668.555947	11958.706265	15379.767181	18377.522699	20964.672132	23058.169737
MAE	0.301511	0.386594	0.466494	0.534107	0.596870	0.647288	0.691188	0.727459
MAPE	30.151093	38.659416	46.649411	53.410738	59.687038	64.728789	69.118756	72.745856
MPE	-10.292482	-14.271728	-18.125159	-21.429857	-24.281573	-26.645624	-28.643325	-30.288128

Scores for General Model with Baseline per station : Linear Regression

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.977059	0.963088	0.942438	0.919898	0.894072	0.869247	0.845490	0.823792
RMSE	53.301446	67.611037	84.430357	99.598507	114.534416	127.249756	138.327950	147.721507
MSE	2841.044094	4571.252262	7128.485159	9919.862662	13118.132553	16192.500294	19134.621644	21821.643486
MAE	0.281557	0.355188	0.424903	0.485079	0.544698	0.592932	0.636739	0.673271
MAPE	28.155708	35.518811	42.490303	48.507864	54.469790	59.293214	63.673875	67.327146
MPE	-10.999592	-15.378526	-19.343655	-22.493803	-25.158362	-27.066982	-28.440084	-29.337709

Scores for General Model with Baseline per station : XGBoost



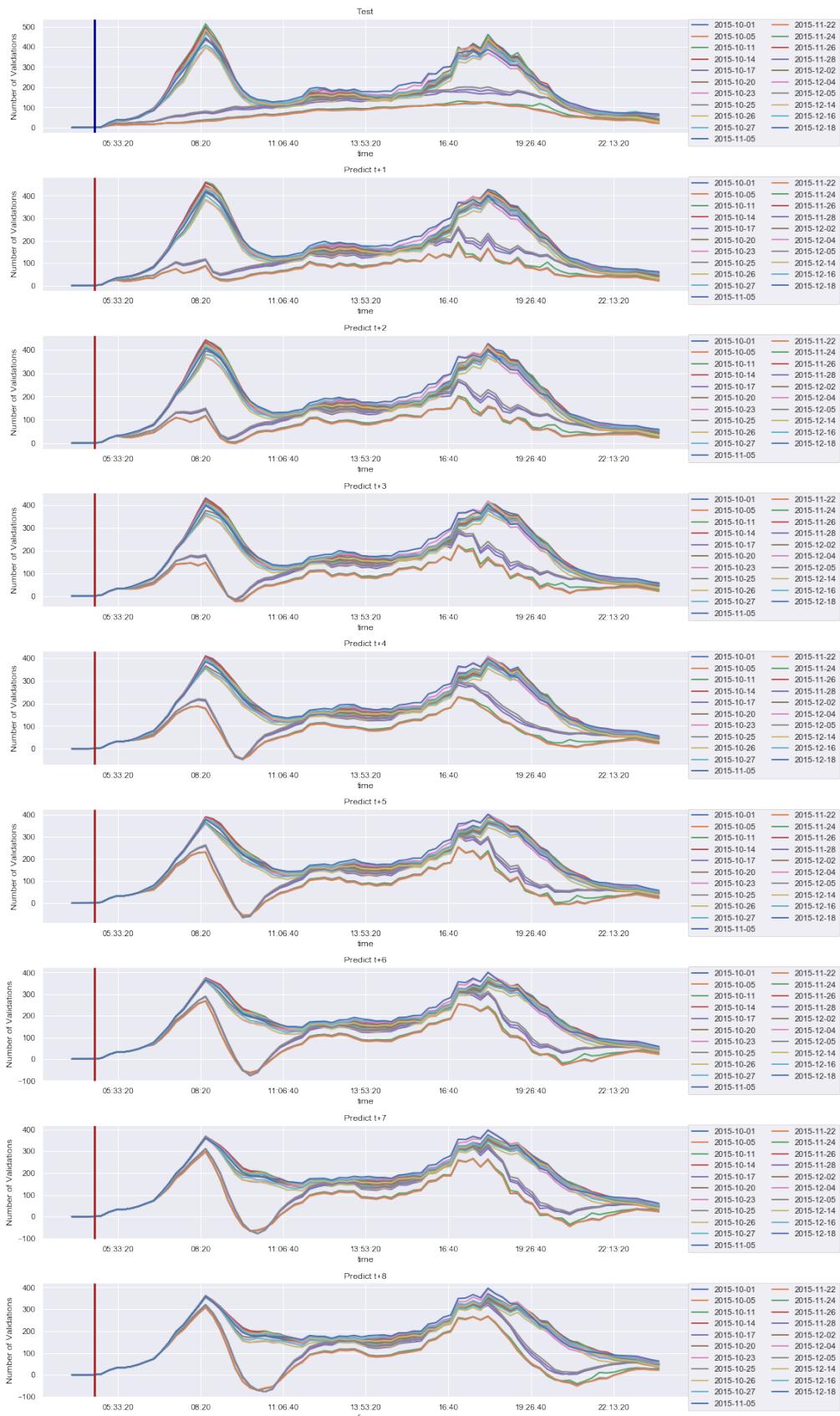
RMSE Score error for General Model with Baseline per station : XGBoost

Comparée à l'échelle de la même figure sans avoir utilisé la *Baseline*, nous remarquons qu'elle est passée de 600 à 300, donc diminuée de moitié.

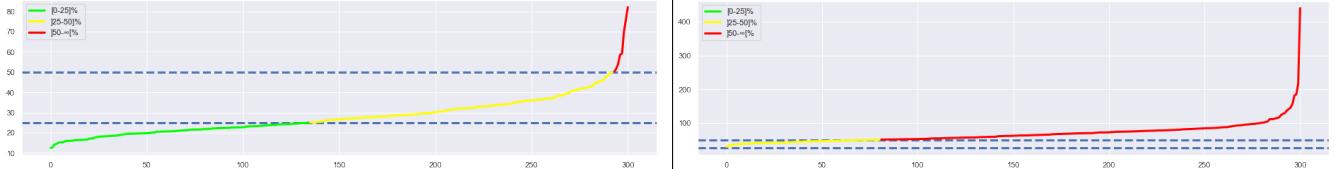
Soulignons que le décalage qui était présent sans la *baseline* a disparu, dû au fait que nous avons soustrait les données par la *baseline per station*, ce qui nous permet d'apprendre le contraste entre cette dernière et nos données.

Néanmoins, la prédiction des *Week-ends* que nous obtenons est assez étrange, elle s'explique par le simple fait que la *baseline per station* est complètement indépendante des jours de la semaine. Disposant de moins de *week – ends* que de jours ouvrés, elle suit donc naturellement la tendance la plus récurrente.

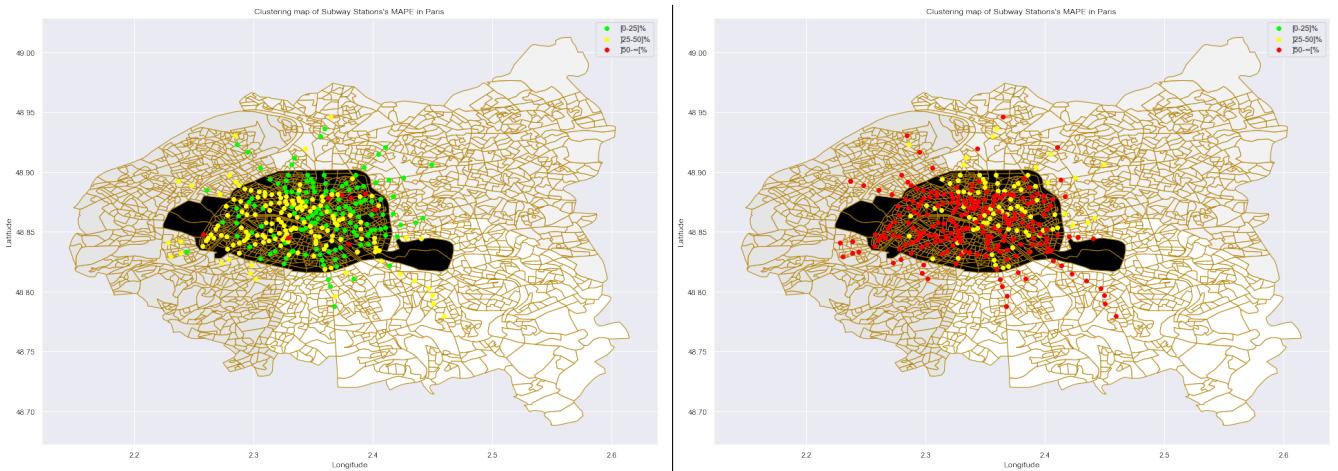
Plus loin, nous verrons qu'appliquer la *baseline par station et par jour* résout ce problème car cette dernière tient compte des jours de la semaine.



Prédictions pour les différents T



MAPE curve of stations



MAPE clusters map of stations

6.2.2.2 AR pour chaque t

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.973638	0.956850	0.933206	0.908190	0.879162	0.854129	0.832161	0.814928
RMSE	57.137753	73.494214	91.953749	108.421155	125.096547	138.227347	149.098917	157.354849
MSE	3264.722817	5401.399491	8455.491931	11755.146908	15649.146079	19106.799447	22230.486949	24760.548437
MAE	0.301511	0.382881	0.458758	0.525475	0.590395	0.638958	0.680575	0.712183
MAPE	30.151093	38.288124	45.875847	52.547480	59.039548	63.895837	68.057503	71.218255
MPE	-10.292482	-14.397518	-18.103360	-21.452191	-24.858667	-27.472098	-29.557365	-30.845729

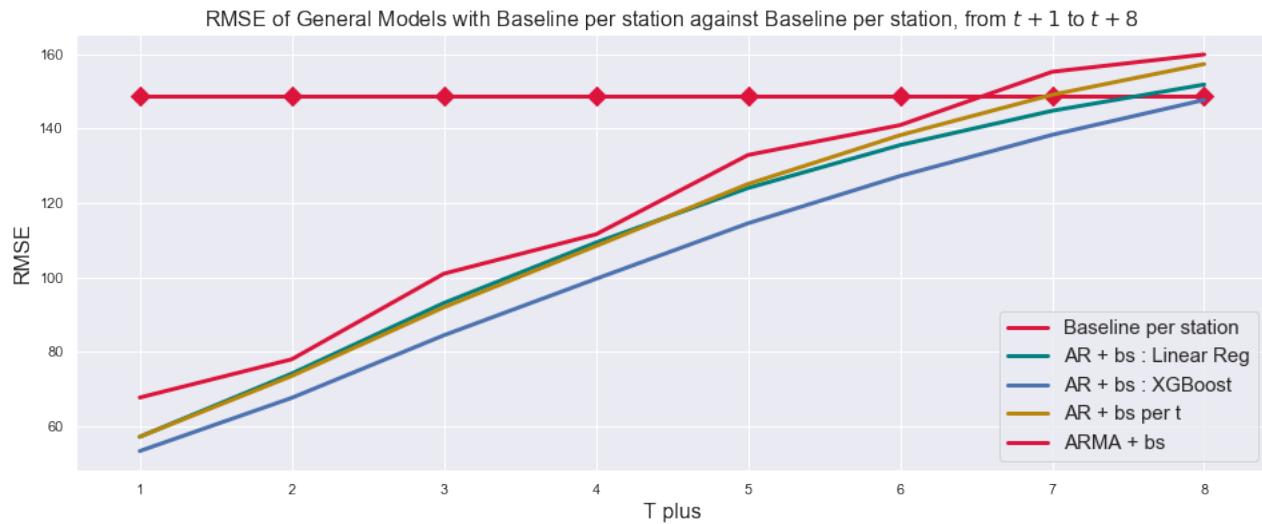
Scores for General Model AR for each T with Baseline per station : Linear Regression

6.2.2.3 ARMA

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.963007	0.950894	0.917643	0.899500	0.857372	0.839611	0.805315	0.793451
RMSE	67.684858	77.983175	100.990864	111.561332	132.902772	140.934907	155.273622	159.934908
MSE	4581.239982	6081.375659	10199.154533	12445.930855	17663.146788	19862.648132	24109.897753	25579.174913
MAE	0.353464	0.413435	0.532680	0.580273	0.690591	0.723910	0.807076	0.822531
MAPE	35.346433	41.343522	53.267990	58.027347	69.059060	72.391009	80.707571	82.253145
MPE	-7.961618	-10.505841	-15.351202	-18.256573	-23.097341	-25.550929	-29.544326	-31.348622

Scores for General Model ARMA with Baseline per station : Linear Regression

6.2.2.4 Comparaison



6.2.3 Avec Baseline par station et par jour

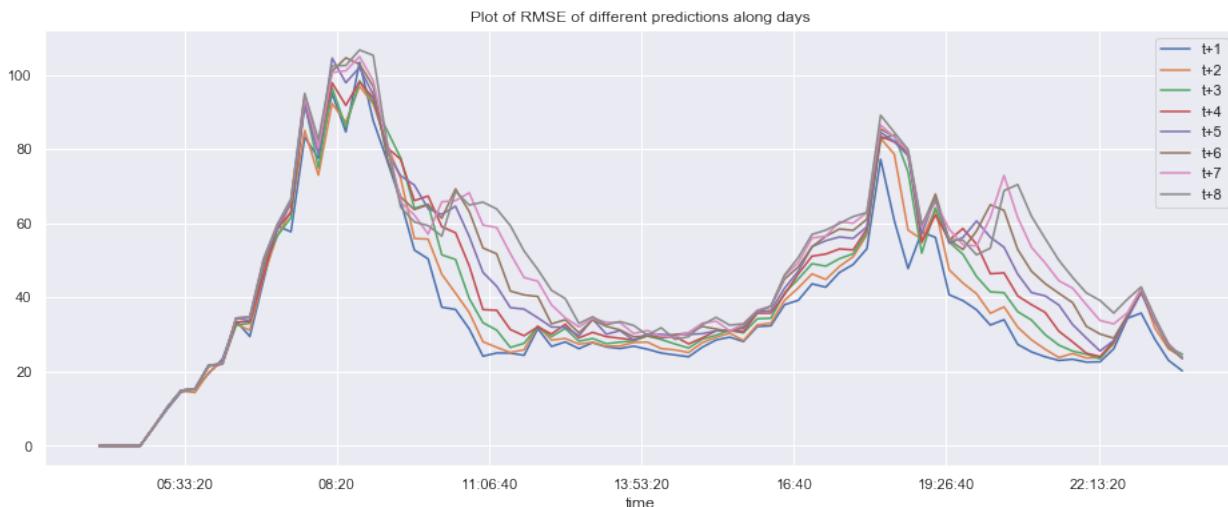
6.2.3.1 AR

♦	t+1 ♦	t+2 ♦	t+3 ♦	t+4 ♦	t+5 ♦	t+6 ♦	t+7 ♦	t+8 ♦
R2	0.984350	0.982526	0.980245	0.978512	0.976534	0.974910	0.973309	0.971643
RMSE	44.024430	46.518363	49.461988	51.585127	53.908164	55.741878	57.492650	59.259480
MSE	1938.150476	2163.958056	2446.488237	2661.025299	2906.090124	3107.156936	3305.404811	3511.685958
MAE	0.205134	0.219452	0.232309	0.244346	0.256218	0.266827	0.276997	0.286321
MAPE	20.513398	21.945203	23.230949	24.434605	25.621805	26.682707	27.699737	28.632095
MPE	-5.430427	-5.887639	-6.238330	-6.563671	-6.898288	-7.056705	-7.175181	-7.260120

Scores for General Model with Baseline per station per day : Linear Regression

♦	t+1 ♦	t+2 ♦	t+3 ♦	t+4 ♦	t+5 ♦	t+6 ♦	t+7 ♦	t+8 ♦
R2	0.984725	0.983123	0.981234	0.979927	0.978316	0.977100	0.976015	0.974930
RMSE	43.493683	45.717105	48.207938	49.858689	51.821041	53.253680	54.501047	55.719173
MSE	1891.700503	2090.053652	2324.005303	2485.888841	2685.420263	2835.954388	2970.364114	3104.626193
MAE	0.201640	0.213471	0.222447	0.230115	0.236500	0.243351	0.249464	0.254818
MAPE	20.164047	21.347131	22.244665	23.011483	23.650001	24.335071	24.946390	25.481773
MPE	-5.925449	-6.552508	-6.926481	-7.192379	-7.493972	-7.679699	-7.761119	-7.812018

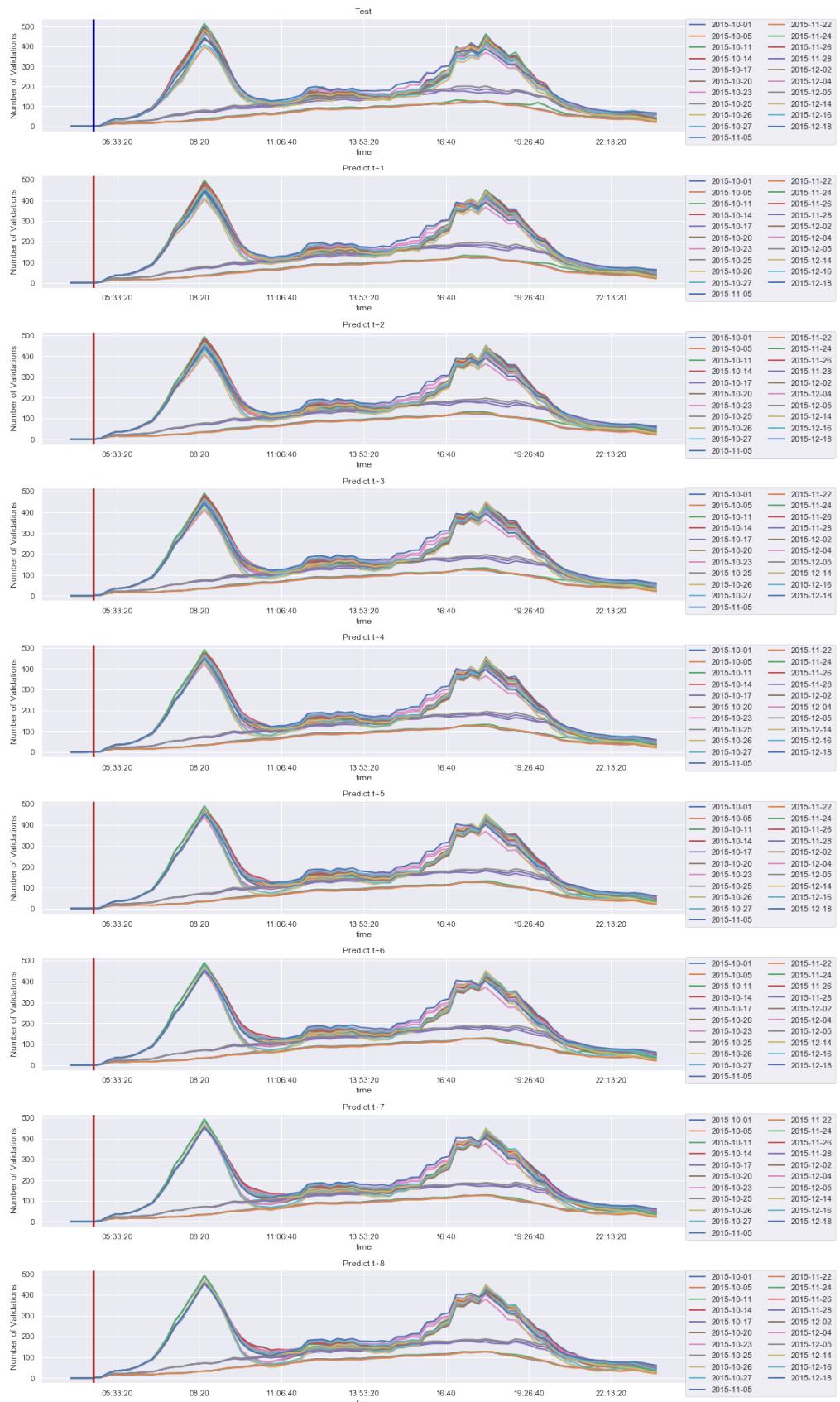
Scores for General Model with Baseline per station per day : XGBoost



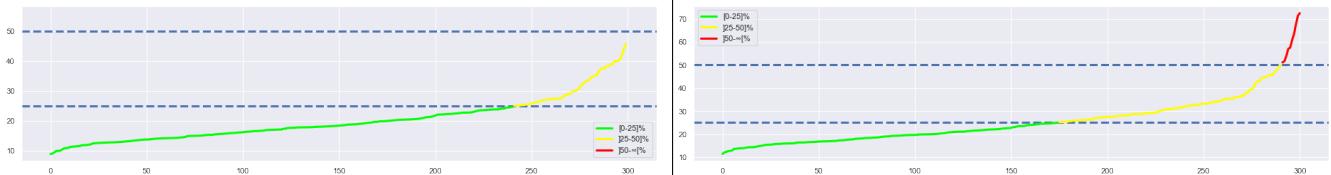
RMSE Score error for General Model with Baseline per station : XGBoost

Une fois de plus, nous avons des résultats encore meilleurs, l'échelle de l'erreur est redescendue à 100 et en observant les prédictions ci-dessous, elles se confondent plutôt bien à la toute première figure qui correspond aux valeurs réelles que nous voulons prédire.

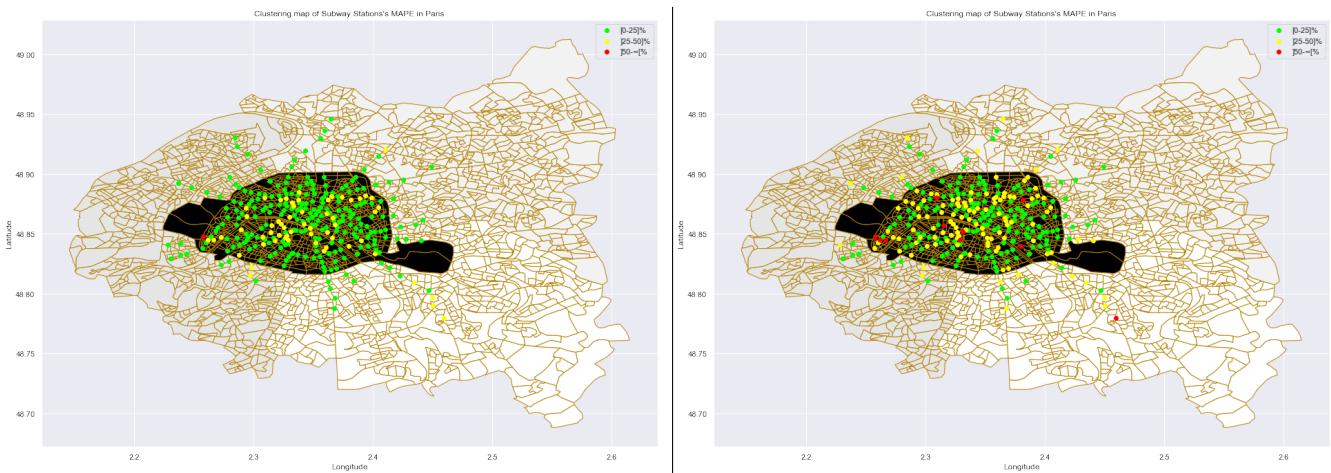
De surcroît, en analysant les scores *MAPE*, à $T+1$, environ 250 stations de métro ont moins de 25% d'erreurs, 50 autres ont entre 25% et 50% d'erreurs et seulement une station a plus de 100% d'erreurs. À $T+8$, la performance du modèle se fait ressentir par l'accentuation des stations avec un score *MAPE* inférieur à 50% globalement.



Prédictions des différents T



MAPE curves of stations



MAPE clusters map of stations

6.2.3.2 AR pour chaque t

\diamond	$t+1 \diamond$	$t+2 \diamond$	$t+3 \diamond$	$t+4 \diamond$	$t+5 \diamond$	$t+6 \diamond$	$t+7 \diamond$	$t+8 \diamond$
R2	0.984350	0.982631	0.980582	0.979147	0.977436	0.976209	0.975105	0.974116
RMSE	44.024430	46.628097	49.579886	51.671486	54.057385	55.822985	57.422222	58.847428
MSE	1938.150476	2174.179465	2458.165108	2669.942482	2922.200837	3116.205616	3297.311564	3463.019759
MAE	0.205134	0.212297	0.217083	0.222681	0.228082	0.232161	0.236384	0.240347
MAPE	20.513398	21.229731	21.708290	22.268127	22.808152	23.216134	23.638356	24.034692
MPE	-5.430427	-6.005963	-6.077314	-6.107455	-6.093925	-5.935145	-5.748865	-5.545445

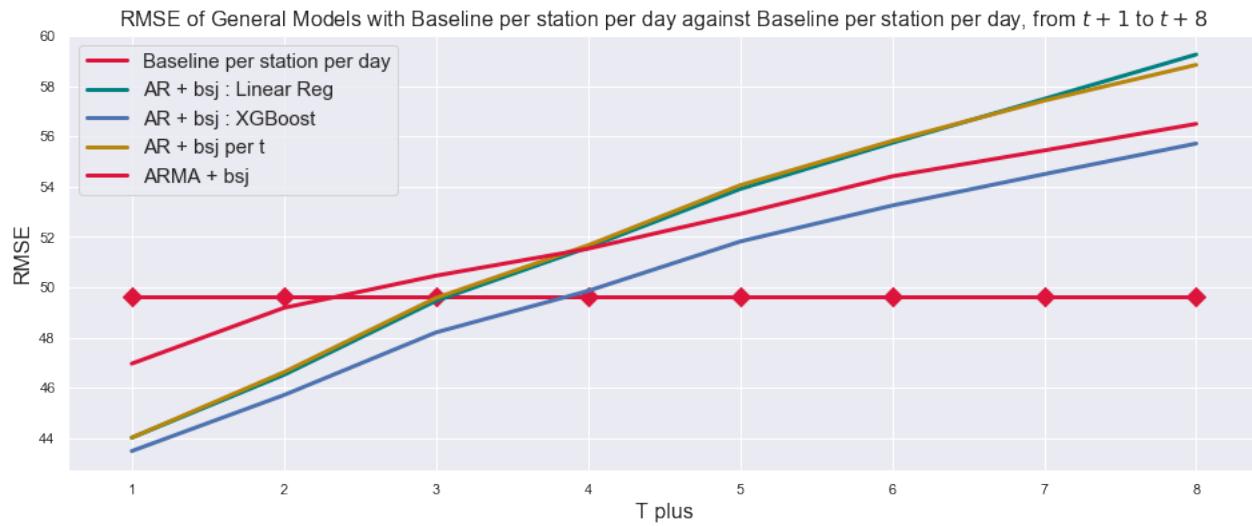
Scores for General Model AR per t with Baseline per station per day : Linear Regression

6.2.3.3 ARMA

\diamond	$t+1 \diamond$	$t+2 \diamond$	$t+3 \diamond$	$t+4 \diamond$	$t+5 \diamond$	$t+6 \diamond$	$t+7 \diamond$	$t+8 \diamond$
R2	0.982187	0.980465	0.979436	0.978554	0.977388	0.976091	0.975179	0.974222
RMSE	46.968062	49.185685	50.464444	51.535614	52.917280	54.414544	55.442380	56.501482
MSE	2205.998838	2419.231623	2546.660143	2655.919521	2800.238553	2960.942594	3073.857503	3192.417434
MAE	0.210900	0.219888	0.224216	0.229161	0.234825	0.241359	0.246557	0.252150
MAPE	21.090036	21.988802	22.421578	22.916135	23.482543	24.135858	24.655749	25.215028
MPE	-5.285707	-5.638120	-5.750998	-5.831480	-6.113920	-6.169062	-6.181191	-6.178409

Scores for General Model ARMA with Baseline per station per day : Linear Regression

6.2.3.4 Comparaison

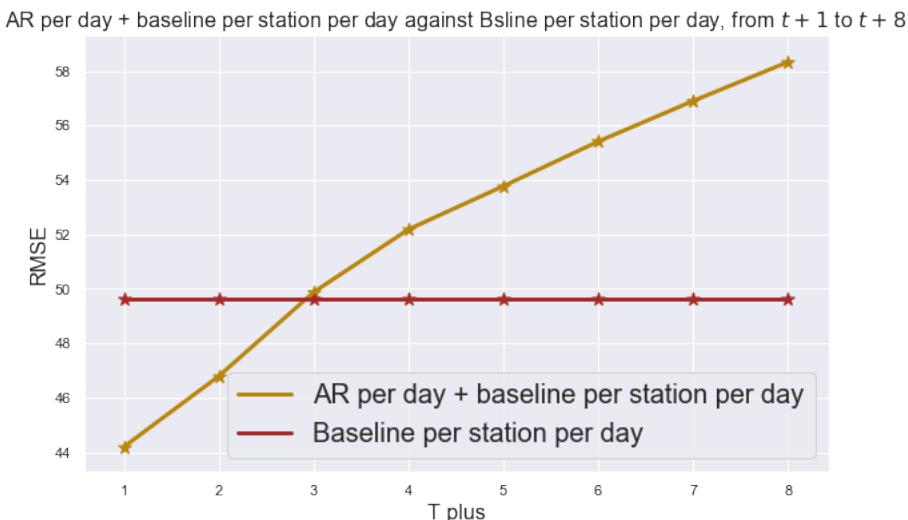


6.3 Un modèle par jour

6.3.1 AR avec *Baseline* par station et par jour

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.984225	0.982319	0.979910	0.978007	0.976647	0.975202	0.973851	0.972521
RMSE	44.199805	46.793824	49.879283	52.187799	53.777270	55.416318	56.905834	58.335652
MSE	1953.622796	2189.662010	2487.942899	2723.566409	2891.994759	3070.968315	3238.273998	3403.048261
MAE	0.208509	0.225731	0.240655	0.254394	0.251802	0.259197	0.266177	0.272068
MAPE	20.850858	22.573117	24.065545	25.439419	25.180229	25.919673	26.617698	27.206849
MPE	-5.553942	-6.077548	-6.478765	-6.840266	-6.849233	-6.917163	-6.943871	-6.929604

Scores for AR per day Model with Baseline per station per day : Linear Regression



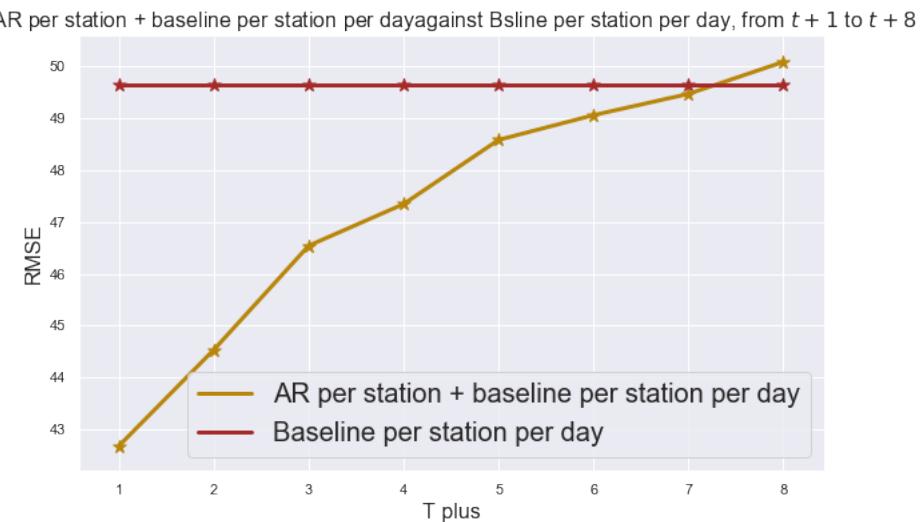
Soit 7 modèles, un pour chaque jour de semaine. Parvient à produire une performance acceptable.

6.4 Un modèle par station

6.4.1 AR avec *Baseline* par station et par jour

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.985293	0.983987	0.982509	0.981899	0.980940	0.980564	0.980237	0.979740
RMSE	42.676357	44.532034	46.541817	47.346066	48.584078	49.061264	49.471771	50.090080
MSE	1821.271448	1983.102050	2166.140694	2241.649968	2360.412598	2407.007642	2447.456148	2509.016114
MAE	0.186558	0.191870	0.195506	0.197972	0.199466	0.200667	0.201648	0.202422
MAPE	18.655753	19.187000	19.550649	19.797160	19.946642	20.066658	20.164804	20.242246
MPE	-5.225469	-5.540440	-5.733638	-5.879379	-6.015169	-6.068297	-6.108166	-6.140788

Scores for AR per station Model with Baseline per station per day : Linear Regression



Ce modèle est probablement le plus performant que nous ayons obtenu, avec une *RMSE* de 50 à $T+8$. Rappelons que c'est un modèle AR par station, soit 301 modèles appris sur des données sur lesquelles ont été soustraite la *la baseline par station et par jour*.

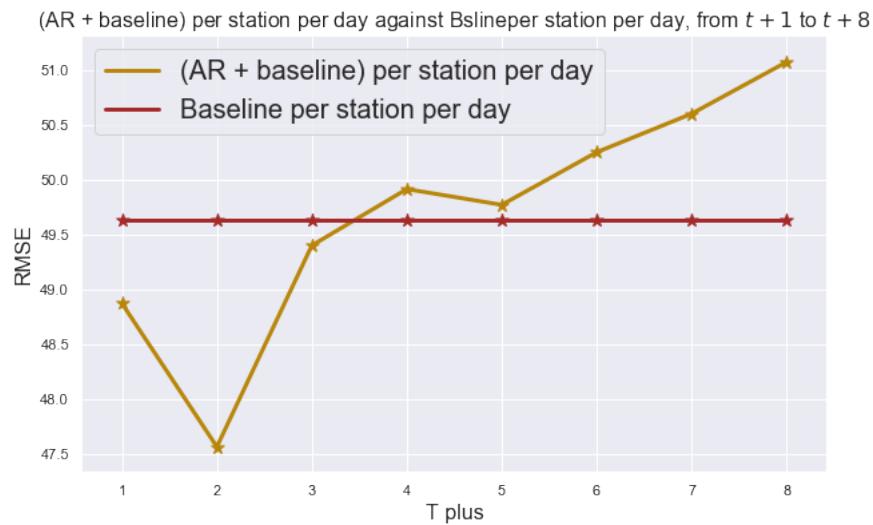
6.5 Un modèle par station et par jour

Soit 301 modèles pour chacun des 7 jours de la semaine, donc 2107 modèles appris.

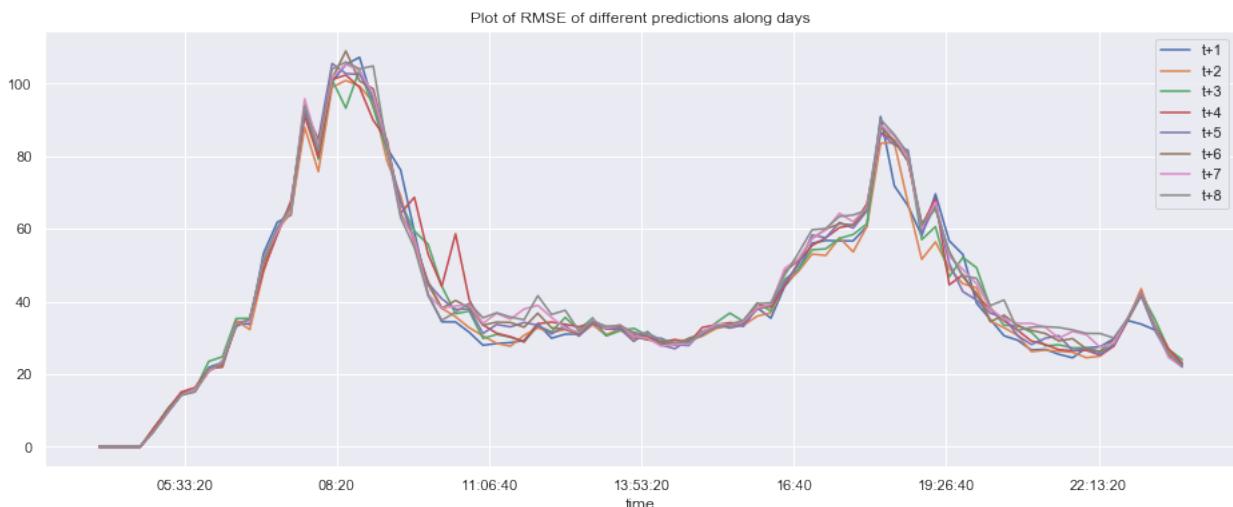
6.5.1 AR avec *Baseline* par station et par jour

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.980714	0.981730	0.980295	0.979883	0.979999	0.979612	0.979326	0.978940
RMSE	48.871503	47.566015	49.398828	49.912351	49.769067	50.248178	50.598615	51.069651
MSE	2388.423812	2262.525755	2440.244244	2491.242796	2476.960011	2524.879420	2560.219854	2608.109242
MAE	0.222593	0.226896	0.228734	0.227641	0.225980	0.227840	0.228173	0.229049
MAPE	22.259264	22.689580	22.873412	22.764111	22.598035	22.783950	22.817300	22.904900
MPE	-6.352620	-7.417192	-7.511497	-7.478582	-8.419354	-8.572126	-8.541936	-8.558475

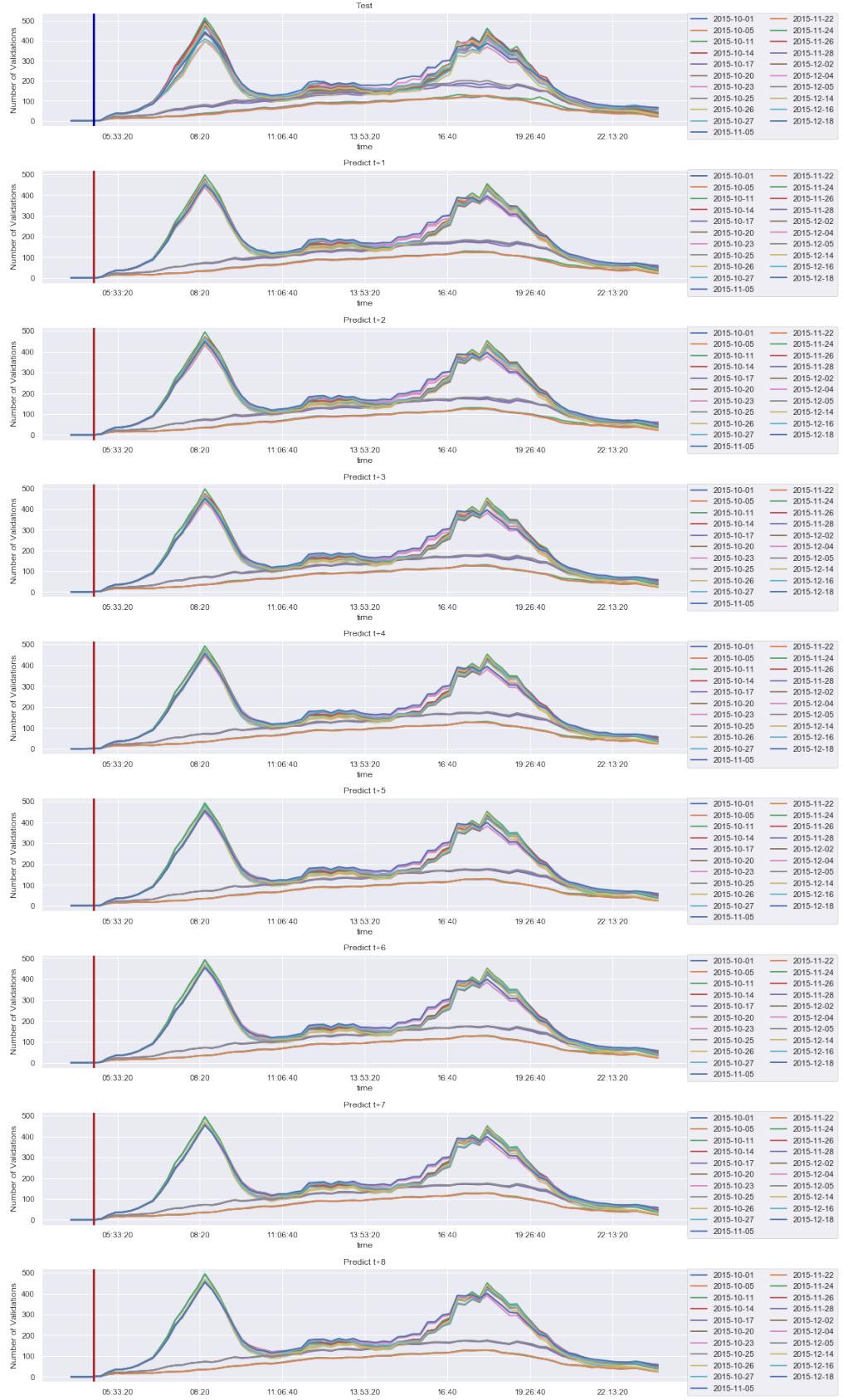
Scores for AR per station Model with Baseline per station per day : Linear Regression

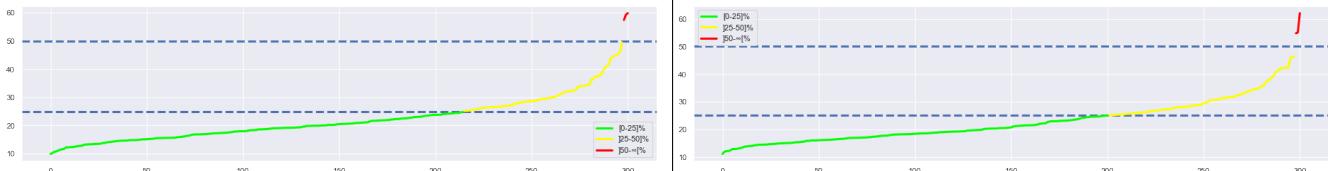


Une courbe d'erreurs assez singulière se dessine, probablement dû au fait qu'il y ait très peu de données à apprendre pour chacun des modèles. Quant à l'erreur moyenne ci-dessous, les courbes sont remarquablement proches d'un T à un autre, preuve irréfutable de l'amélioration de la performance au fil de temps.

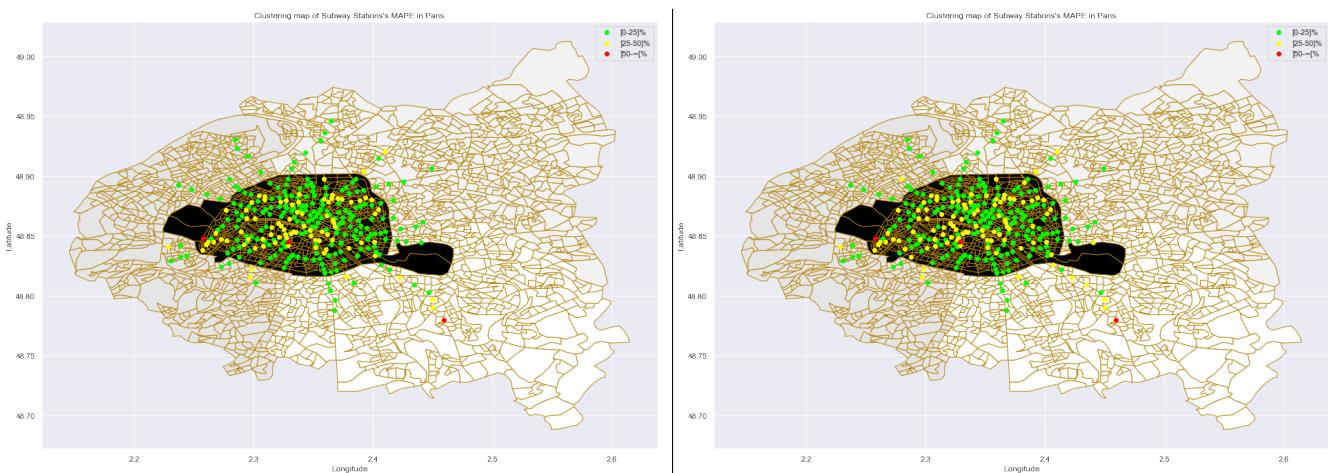


RMSE Score error for General Model with Baseline per station : XGBoost





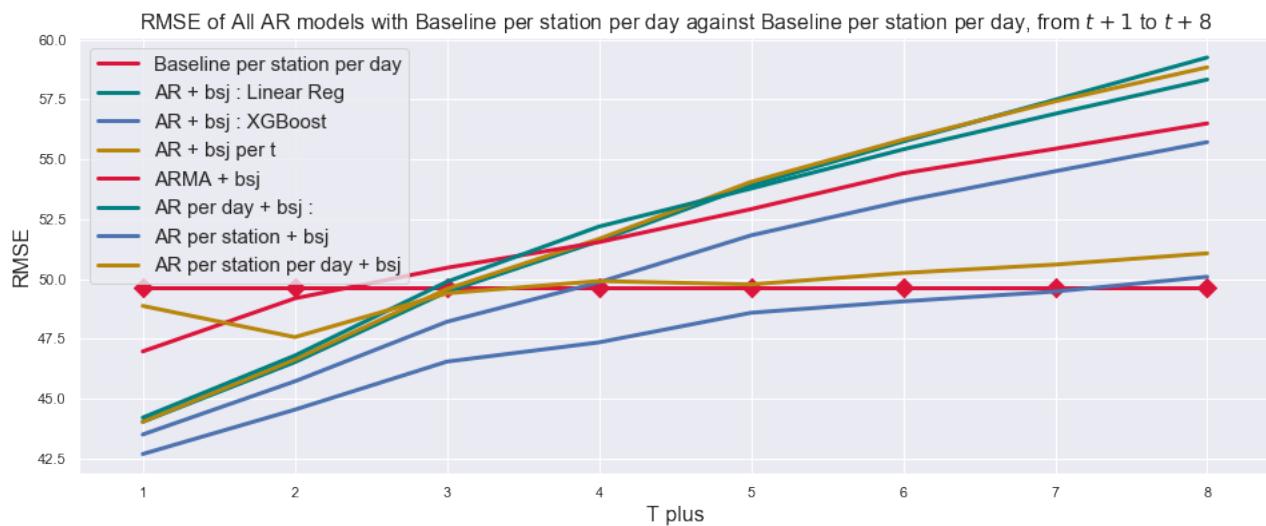
MAPE curves of stations



MAPE clusters map of stations

Prédictions des différents T

6.5.1.1 Comparaison



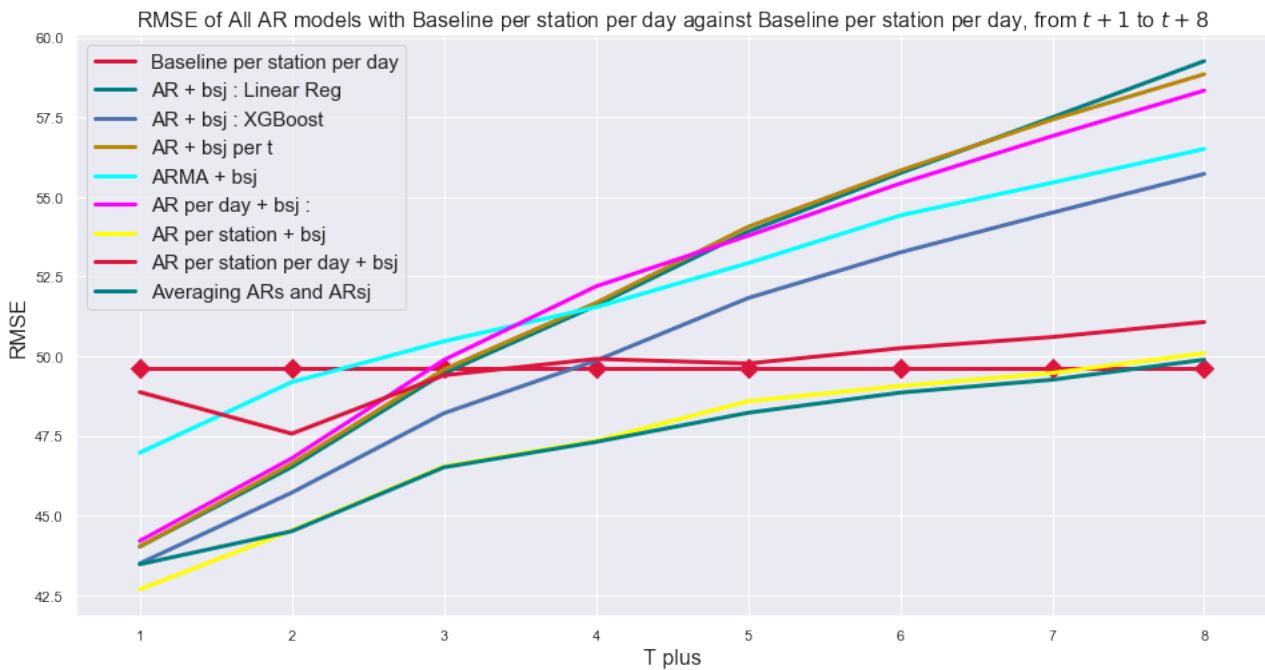
7 Apprentissage d'ensemble

7.1 Averaging Models

Moyenner les résultats de différents modèles est une technique qui nous permet le plus souvent d'améliorer nos résultats. Par exemple, en faisant la moyenne de seulement deux (2) modèles, le *modèle par station + bsj* ainsi que le *modèle par station et par jour + bsj*, nous arrivons à glisser légèrement en dessous du meilleur modèle.

	t+1	t+2	t+3	t+4	t+5	t+6	t+7	t+8
R2	0.984744	0.984006	0.982533	0.981928	0.981223	0.980728	0.980406	0.979903
RMSE	43.466483	44.504782	46.510035	47.307511	48.221752	48.853189	49.259252	49.887552
MSE	1889.335122	1980.675639	2163.183354	2238.000583	2325.337349	2386.634122	2426.473929	2488.767809
MAE	0.195326	0.201676	0.204690	0.205574	0.206805	0.208546	0.209395	0.210329
MAPE	19.532586	20.167597	20.469027	20.557366	20.680532	20.854559	20.939452	21.032937
MPE	-5.787017	-6.476981	-6.620684	-6.677144	-7.215490	-7.318423	-7.323243	-7.347823

Scores for AR per station Model with Baseline per station per day : Linear Regression



7.2 Staking Models

Pour aller plus loin, nous pourrions au lieu de faire la moyenne des modèles, apprendre de nouveau sur les nouvelles prédictions et cela pour chaque intervalle de temps. Ainsi pour prédire à T+2, en supposant que nous ayons déjà utilisé deux (2) modèles pour la même prédiction, $X1pred$ et $X2pred$, nous prendrons comme y l'intervalle de temps correspondant et comme X les deux mêmes intervalles dans $X1pred$ et $X2pred$.

8 Conclusion

En définitive, ce projet nous a tout d'abord permis d'aborder un sujet d'actualité et très demandé sur le marché de l'emploi, les séries temporelles.

Après avoir analysé et pré-traité les données avec parcimonie, nous avons étudié quelques modèles auto-régressifs et pour chacun d'eux, avons tout aussi étudié les cas de spécificités possibles notamment leurs applications par jour, par station, par station et jour, puis avons comparé les résultats contre les *Baselines*.

Nous concluons que dépendamment des données que nous disposons, le meilleur modèle en pratique après celui obtenu en faisant la moyenne d'autres modèles, est un modèle *AR* simple par station appliqué sur des données combinées à la *baseline par jour et par station*.

Toute fois, gardons-en tête que nous ne travaillons que sur 78 jours, à peine trois mois, après le pré-traitement, nous croyons que si nous avions plus de données, le modèle par station et par jour combiné à la *baseline par station et par jour* aurait été plus performant.

Pour finir, le travail réalisé, toute l'implémentation ainsi que les notebooks utilisés sont sur : <https://github.com/ATidiane/PLDAC19.git>.

Références

- Valentin Guiguet, Nicolas Baskiotis, Vincent Guigue, and Patrick Gallinari. Context-aware forecasting for multivariate stationary time-series. 2018.
- Miloš Milenković, Libor Švadlenka, Vlastimil Melichar, Nebojša Bojović, and Zoran Avramović. Sarima modelling approach for railway passenger flow forecasting. *Transport*, 33(5) :1113–1120, 2018.
- Wikipedia. Autoregressive model — Wikipedia, the free encyclopedia. <http://en.wikipedia.org/w/index.php?title=Autoregressive%20model&oldid=897705087>, 2019a. [Online ; accessed 01-June-2019].
- Wikipedia. Autoregressive-moving-average model — Wikipedia, the free encyclopedia. <http://en.wikipedia.org/w/index.php?title=Autoregressive%E2%80%93moving-average%20model&oldid=891722434>, 2019b. [Online ; accessed 02-June-2019].
- Zheng Zhao, Weihai Chen, Xingming Wu, Peter CY Chen, and Jingmeng Liu. Lstm network : a deep learning approach for short-term traffic forecast. *IET Intelligent Transport Systems*, 11(2) :68–75, 2017.