Projet Machine Learning

Prévision du nombres de crimes par mois à Philadelphie

23/12/2016

Jean-Baptiste REMY Robin BEAUDET

Professeur : Xavier DUPRE

Table des matières

1 Introduction

A travers ce projet, nous proposons plusieurs algorithmes de machine learning afin de répondre à une problématique qui est plus que jamais d'intérêt : *peut-on prédire le crime ?* Aujourd'hui, ce qui semblait être de la science-fiction il y a quelques années est d'actualité. Des algorithmes comme PredPol (predictive policing) ont déjà fait leurs preuves aux Etats-Unis et s'exportent désormais au Royaume-Uni.

En tenant compte des crimes passés et de variables sociodémographiques, nous avons tenté nous aussi d'implémenter des modèles de prédiction. Pour cela, nous avons utilisé deux jeux de données issus de la plateforme *Kaggle*, recensant les rapports de police relatifs aux crimes et délits enregistrés dans deux villes des Etats-Unis, à savoir *Minneapolis* et *Philadelphie*. Pour la première ville, nous disposons de données couvrant les années 2010 à 2016. Pour la seconde, les données couvrent les années 2006 à 2016.

Pour des raisons pratiques que nous évoquerons, il a été décidé de ne se concentrer que sur la ville de *Philadelphie* et de limiter nos modèles à cette seule ville. Ce document est une synthèse de notre travail. Y sont exposés notre démarche, le cheminement nous ayant conduit à adopter telle ou telle direction, les principaux résultats liés au travail descriptif réalisé, ainsi que l'explication des modèles mis en œuvre et l'exploitation de leurs résultats. Pour une meilleure compréhension, il est donc nécessaire de se référer aux *notebooks* afin d'avoir accès aux détails du code.

Note : les graphiques présents dans ce rapport ont été modifiés pour mieux respecter l'harmonie visuelle du rapport, mais tous les résultats se retrouvent dans les notebooks. La principale modification est le passage de ggpolot à seaborn-deep.

2 Genèse du projet : travail sur la ville de Minneapolis

Pour le détail du code, se référer au notebook "Travail sur la base (Minneapolis)"

La première base sur laquelle nous avons travaillé est mise à disposition sur la plateforme Kaggle, à cette adresse : `https://www.kaggle.com/mrisdal/minneapolis-incidents-crime/kernels`

Comme évoqué précédemment, le jeu de données recense les crimes et délits commis dans la ville de Minneapolis entre 2010 et 2016. Mis sous forme de Data Frame, chaque ligne correspond à un crime. Pour chaque crime, nous disposons de plusieurs informations comme la date et l'heure à laquelle il a été commis, le lieu, ou encore le type de crime.

Plusieurs informations étant redondantes (notamment des variables relatives à la date), nous avons réduit le champ d'étude pour ne conserver que quelques variables. En prévision d'une analyse plus poussée du crime, en particulier à l'échelle d'une seule journée, nous avons créé des variables de temps permettant facilement d'extraire l'heure. Enfin, s'inspirant de la hiérarchie des crimes et délits établie par le FBI, nous avons regroupé les crimes par types en créant les catégories de crimes suivantes :

Burglary : cambriolages

Robbery : vols de type braquage

Rape : viols et autres agressions sexuelles

Homicide : meurtres

Vehicle_theft : vols de véhicules

Theft : vols mineurs

Assault : agressions physiques

La ville de Minneapolis s'organise par communes, au nombre de 11. Chaque commune est composée de quartiers (76 en tout). Etant donné que la ville est relativement petite (392,000 habitants), il ne nous a pas semblé forcément pertinent de faire une analyse quartier par quartier. Nous avons donc pris la décision de construire la variable *Community* à partir de la variable *Neighborhood* déjà présente. Le regroupement des quartiers en *Comunities* est disponible ici : `https://en.wikipedia.org/wiki/Neighborhoods_of_Minneapolis`

A ce stade, plusieurs champs d'étude s'offraient à nous. Réaliser d'une part une analyse temporelle, et d'autre part une analyse spatiale.

2.1 Statistiques temporelles

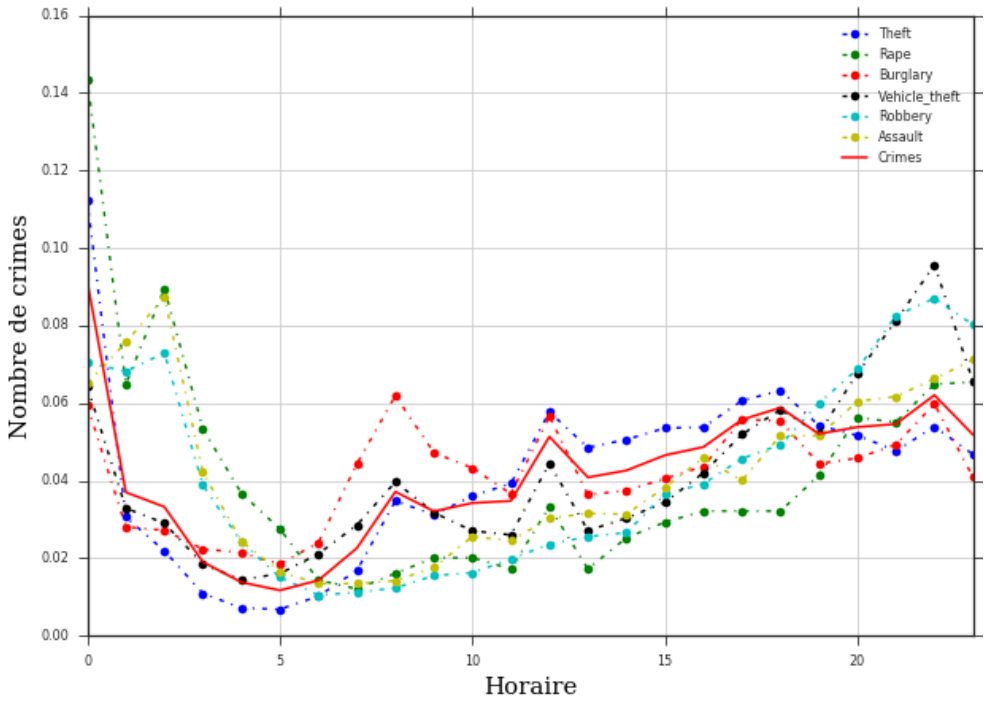
Pour le détail du code, se référer au notebook "Statistiques sur le temps (Minneapolis)"

Nous nous sommes dans un premier temps intéressés à une analyse du crime «à la journée». Quelques graphes permettent de rendre compte d'un premier phénomène intéressant, mais loin d'être surprenant. Le crime n'est pas réparti de façon homogène sur une journée. Il est minimal aux alentours de 5h du matin et ne fait qu'augmenter au cours de la journée pour atteindre son maximum à minuit.

Toutefois, il est important de garder en tête que, dans nos données, l'heure d'un crime correspond à l'heure rentrée lors du dépôt de plainte ou lors du rapport de police. Celle-ci n'est donc pas tout à fait exacte, ce que confirme notre analyse du crime heure par heure. En effet, la plupart des crimes sont enregistrés à une heure pile ou arrondie à la demi-heure près. Encore une fois, ceci n'est pas surprenant : il est plus simple d'enregistrer un évènement ayant lieu à 9h30 plutôt qu'à 9h38 par exemple. Il a donc été décidé de poursuivre l'analyse uniquement heure par heure.

Enfin, notons que quel que soit le type de crime, la tendance observée au cours de la journée est la même : le nombre de crimes est au plus bas vers 5h du matin et ne cesse d'augmenter jusqu'à minuit.

FIGURE 1 – Répartition des crimes en fonction de l'horraire

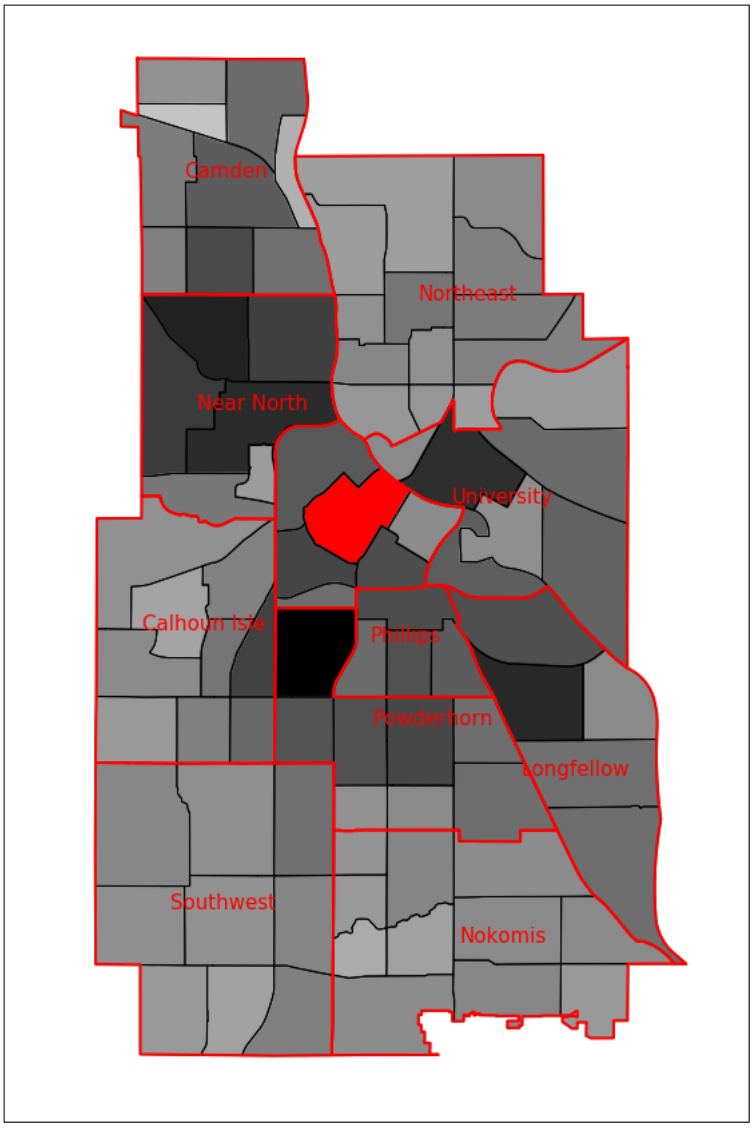


2.2 Visualisation spatiale

Pour le détail du code, se référer au notebook "Visualisation spatiale (Minneapolis)"

Afin de tracer des cartes de la ville de Minneapolis, nous avons récupéré les shapefiles des quartiers et des communes de Minneapolis sur le site officiel de la ville de Minneapolis (`http://opendata.minneapolismn.gov`). On visualise ainsi le nombre de crimes par quartier, selon le type de crime, ce qui nous permet d'identifier des quartiers à risques et des quartiers plus calmes. Comme on le voit, la plupart des crimes et délits sont concentrés dans le centre de la ville. La partie Nord-Ouest (quartiers Near North et Camden) est également un lieu de concentration des crimes. Pour le reste, on voit qu'il y a peu de crimes.

FIGURE 2 – Répartition par quartier des crimes



2.3 Changement d'orientation

Le problème majeur auquel nous faisions face à ce stade du projet est qu'il y a relativement peu de données pour la ville de Minneapolis. Celle-ci est en effet une «petite» ville de 400,000 habitants. Implémenter un algorithme d'apprentissage, notamment pour prédire le nombre de crimes par quartiers, nous semble difficilement réalisable.

L'autre souci que nous avons rencontré est le traitement du problème en tant que série temporelle. Pour rappel, nous disposons de données pour 6 ans. Cela nous semblait peu, d'autant plus que l'année 2016 n'est pas complète : on pourrait alors prédire l'année 2015, mais nous n'aurions donc que 4 années pour entrainer notre algorithme.

De plus, il est évident que la criminalité dépend d'autres facteurs, notamment des facteurs sociaux. Nous pensons que la crise de 2008 qui a frappé de plein fouet certaines villes des Etats-Unis a probablement eu un impact non négligeable. Ne pas disposer de données datant d'avant 2008 nous a alors paru dommageable.

Plutôt que de réaliser une analyse spatiale qui demanderait surement un travail trop important, peut-être était-il préférable se concentrer dans un premier temps sur la prédiction du nombre de crimes dans la ville à un instant donné.

Heureusement, nous avons pour cela à notre disposition un jeu de données similaire dans la ville de *Philadelphie*. La ville de Philadelphie est beaucoup plus grosse, il y a environ 1.5 millions d'habitants, et nous disposons de données pour les 10 dernières années, ce qui est conséquent et permet de comparer l'état de la ville avant et après la crise. Enfin, il est important de souligner que la ville de Philadelphie est l'une des villes les plus pauvres et les plus violentes des Etats-Unis. Le nombre de données disponibles est en conséquence beaucoup plus important.

Nous avons donc changé d'orientation et décidé de travailler sur la ville de Philadelphie, avec l'idée de prédire le nombre de *crimes par mois* sur une année. A ce stade, les idées auxquelles nous pensions étaient les suivantes :

— prédire le nombre de crimes pour la dernière année du jeu de données relatif à la ville de Philadelphie
— entrainer notre algorithme sur la ville de Philadelphie et le tester sur la ville de Minneapolis, en incluant des variables socio-économiques telles que le taux de pauvreté ou le taux de chômage.

3 Crimes à Philadelphie

Décision ayant été prise de se concentrer sur la ville de Philadelphie pour les raisons évoquées ci-dessus, il a fallu recommencer le travail de construction de la base et refaire une étude descriptive. Le jeu de données est là aussi issu de la plateforme Kaggle, disponible à l'adresse suivante : `https://www.kaggle.com/mchirico/philadelphiacrimedata`. Nous avons assez rapidement abandonné l'idée de réaliser une étude spatiale du crime pour se concentrer sur la prédiction du nombre de crimes par mois, ce qui représentait déjà une charge de travail conséquente.

3.1 Construction de la base

Pour le détail du code, se référer aux notebooks «Travail sur la base (Philadelphie)» et «Variables socioéconomiques (Philadelphie)».

Le travail de nettoyage de la base est en tout point similaire à celui effectué sur la base de Minneapolis, les données étant du même type (pour rappel, date et lieu du crime, type de crime tel que renseigné par la police, etc.). Ainsi, il fut décidé de recréer des variables de catégories de crimes, sur le modèle de ce que nous avions fait sur le jeu de données concernant la ville de Minneapolis. Les types de crimes suivants ont donc été créés : Other Assaults, All Other Offenses, Assault, Theft Burglary, Order, Arson, Fraud, Robbery, Vagrancy/Loitering, Rape, Other Sex Offenses (Not Commercialized), Homicide.

Notre Data Frame d'étude a été réorganisé de façon à ce qu'une ligne corresponde à un mois, la première ligne correspondant ainsi à Janvier 2006, la seconde ligne à Février 2006, etc. En raison du manque de renseignement pour les deux derniers mois de l'année 2016, nous avons décidé de les supprimer.

La nouveauté par rapport au premier jeu de données concerne clairement l'ajout de nouvelles variables. Nous sommes persuadés que le crime est corrélé à des variables sociodémographiques telles que le taux de pauvreté, le taux de chômage, le niveau d'éducation moyen, le nombre d'hôpitaux et d'infrastructures publiques, etc. Malheureusement, collecter toutes ces données est fastidieux et nous avons dû nous résoudre a n'en sélectionner que quelques-unes. Les variables retenues ont été extraites du site du Bureau of Labor Statistics (`https://www.bls.gov/`) et concernent les données suivantes :

Evolution du taux de chômage unemployment rate

Taux d'activité labor force

Emploi employment

Chômage unemployment

Emploi par secteur d'activité Mining, Logging and Construction ; Manufacturing ; Trade, Transportation and Utilities ; Information ; Financial Activities ; Professional and Business Services ; Education and Health Services ; Leisure and Hospitality ; Other Services ; Government

L'avantage de ces variables est qu'elles sont disponibles pour chaque mois des années qui nous intéressent (2006 à 2016). Il nous semblait intéressant de disposer de ces données afin notamment de voir l'impact de la crise sur certains secteurs d'activité ou sur le taux de chômage ou, à l'inverse, de voir dans quelle mesure les emplois du service public ont évolué (ce qui, dans le cas d'une hausse, traduirait plutôt un plan de sauvetage économique de l'Etat américain).

Enfin, nous avons ajouté plus tard deux autres variables à notre jeu de données (*pour le détail du code, se référer au notebook «Population totale et taux de pauvreté (Philadelphie)»*) :

Population totale Total population

Taux de pauvreté Poverty rate

En effet, il ne serait pas judicieux de simplement étudier le nombre de chômeurs ou le nombre d'emplois sans le rapporter à la population totale. Concernant la ville de Philadelphie, comme le montre le graphique décrivant l'évolution de la population totale au cours du temps que nous avons tracé, elle a atteint un niveau historiquement bas en 2006 mais n'a, depuis, cessé d'augmenter. Il est également important de noter que la population totale est une donnée annuelle, et non mensuelle. Afin de disposer des valeurs de la population à chaque mois de l'année, nous avons réalisé une simple *interpolation linéaire*. Gardons cependant en tête que c'est une approximation : les mouvements de population ne sont probablement pas linéaires dans le temps. Par exemple, les mois précédents les rentrées scolaires sont certainement sujets à des départs ou des arrivées plus importants de population.

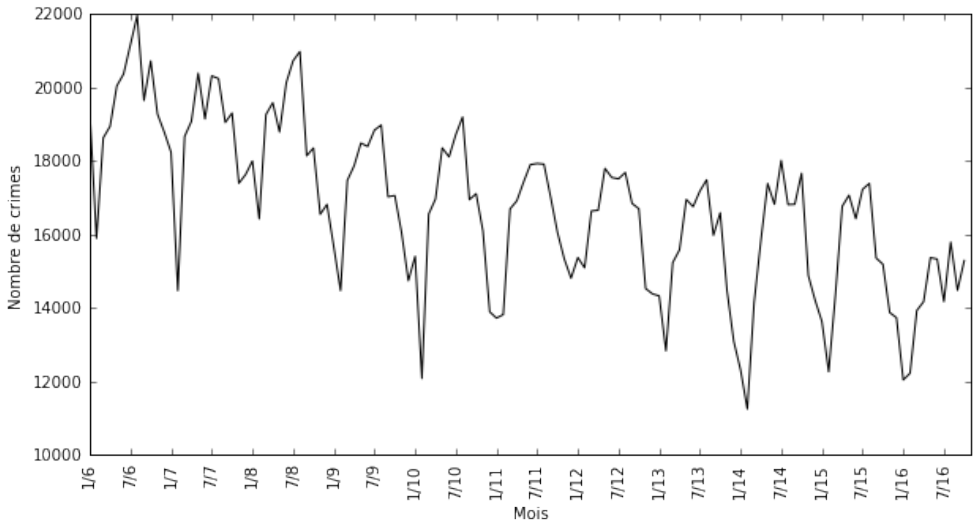
Concernant le taux de pauvreté, nous avons tenu à rajouter cette variable pour la simple et bonne raison que plus de 20% de la population de Philadelphie vit en deçà du seuil de pauvreté, classant tristement la ville comme l'une des plus pauvres des Etats-Unis. Là encore, nous n'avons trouvé que des mesures annuelles du taux de pauvreté, celui-ci n'évoluant quasiment pas d'une année à l'autre. De plus, les chiffres avancés selon les sources étaient parfois contradictoires. Bien qu'ayant là aussi réalisé une interpolation linéaire afin de disposer des données pour chaque mois, nous avons très peu utilisé cette variable par la suite, par manque de fiabilité.

3.2 Statistiques descriptives

Pour le détail du code, se référer au notebook «statistiques descriptives (Philadelphie)».

3.2.1 Evolution du nombre de crimes dans le temps

FIGURE 3 – Nombre de crimes par mois



Graphiquement, on observe deux phénomènes intéressants : depuis 2006, le nombre de crimes connait une tendance à la baisse. De plus, pour chaque année, on observe une saisonnalité (le nombre de crimes étant maximal en été). C'est cette observation qui nous a poussés à traiter le problème sous forme de série temporelle, en identifiant d'une part la tendance (la trend), et d'autre part la saisonnalité et les résiduts. L'étude de la moyenne glissante permet ainsi de rendre compte de la tendance à la baisse du nombre de crimes.

Remarque : nous avons pris pour le calcul de la moyenne glissante et de la variance glissante une fenêtre de 12 mois, soit une année. Ceci nous paraissait logique au vu de la saisonnalité annuelle évoquée précédemment.

3.2.2 Stationarisation de la série

Pour traiter la série sous forme de série temporelle, il est nécessaire de stationnariser celle-ci. Pour ce faire, nous avons différencié notre série (donnant lieu à la création d'une nouvelle variable, nommée Crime_Diff_1). Graphiquement, on peut voir que différencier une fois semble suffisant afin de rendre la série stationnaire. On pourrait effectuer un test de stationnarité (test de Dickey Fuller) pour confirmer le résultat, mais une simple analyse graphique nous a apparu suffisante.

FIGURE 4 – Differentielle du Nombre de crimes par mois

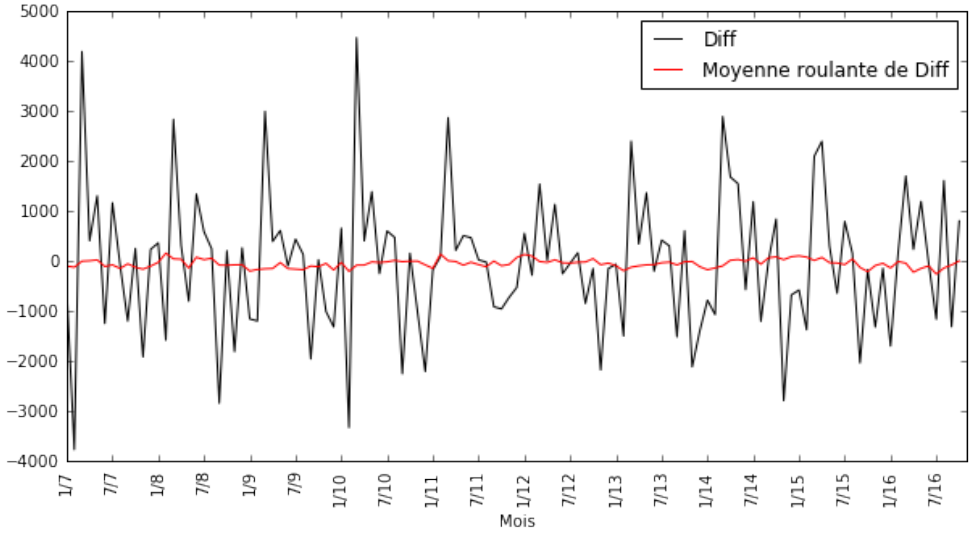
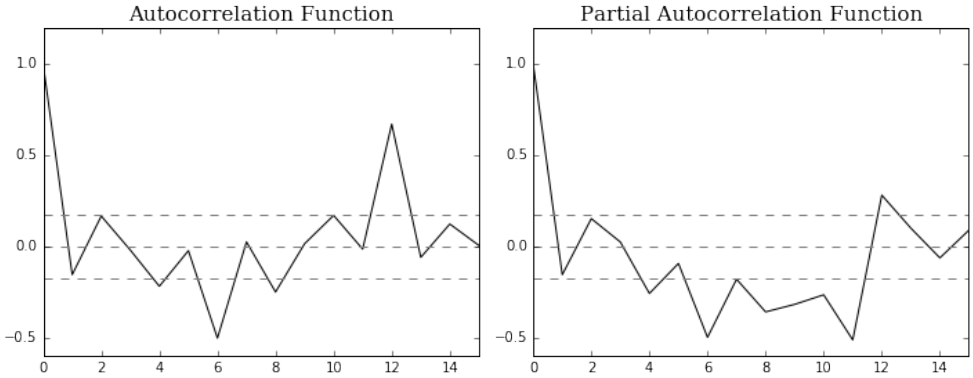


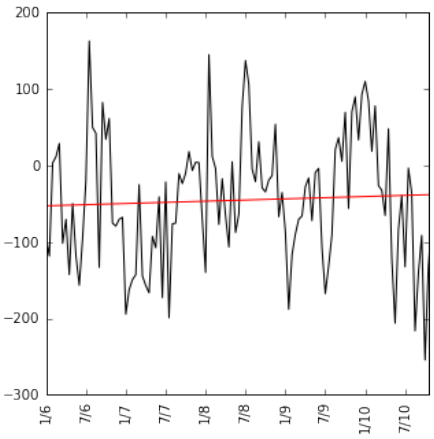
FIGURE 5 – Fonctions d'autocorrelation de la Diff de Crimes



L'analyse de la fonction d'autocorrélation de notre série différenciée permet une nouvelle fois de rendre compte de la saisonnalité que nous observions. En effet, la série est très fortement corrélée positivement avec elle-même décalée d'une année.

Il nous a semblé intéressant d'étudier également la tendance de la série différenciée. Une régression linéaire de la moyenne glissante de la série différenciée montre une légère tendance à la hausse. Ainsi, depuis 2006, le crime chute, mais

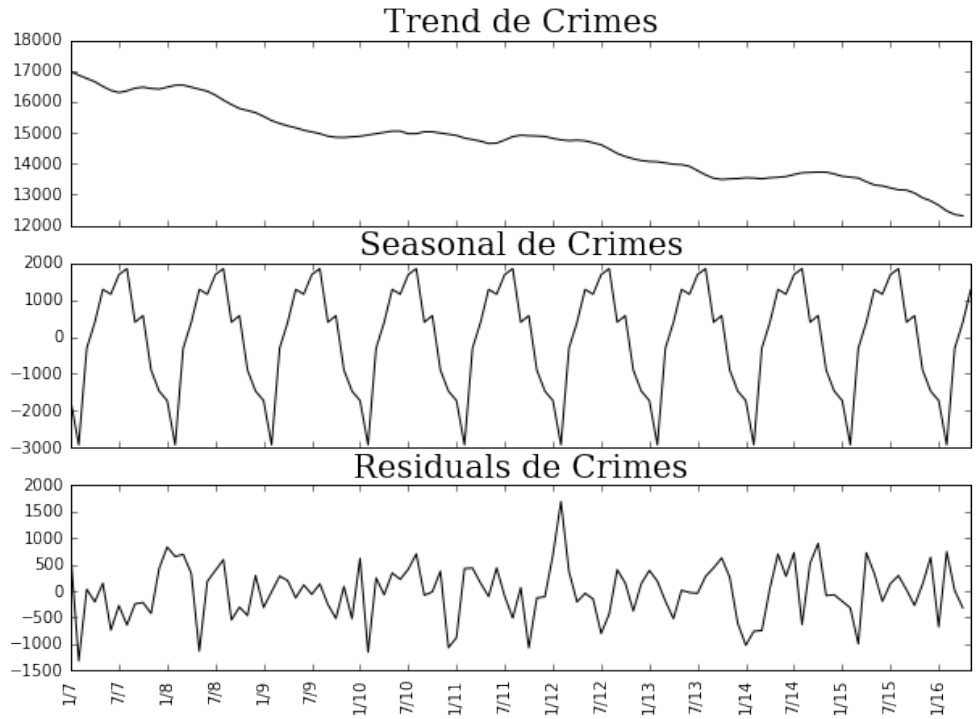
FIGURE 6 – Regression de la Rolling mean de Diff de Crimes



cette chute semble se ralentir. Potentiellement, cela peut-être un effet dû à la crise de 2008.

Après avoir stationnarisé la série, étudions désormais sa décomposition. La librairie statsmodels nous permet d'effectuer facilement ces analyses. Il en ressort que -sans surprise-la tendance et la saisonnalité devraient être faciles à prédire. A ce stade du projet, nous pensons que nos efforts seraient dirigés vers l'étude des résidus, qui ne sont expliqués ni par la tendance, ni par la saisonnalité. Malheureusement, la suite de l'étude nous prouvera le contraire.

FIGURE 7 – Décompositon de la série



3.2.3 Etude des types de crimes

Nous avons souhaité regarder comment se "comportent" les différents types de crimes pour éventuellement évincer ceux qui sont totalement imprévisibles ou qui se comportent aléatoirement. Pour cela, nous avons réalisé quelques graphiques qui permettent de comparer les évolutions de chaque type de crimes avec la variables Crimes. On peut ainsi étudier la part de chacun de ces types de crimes dans la variable Crimes.

Parce qu'ils sont moins intéressants à prédire ou alors trop aléatoires voire décorrélés du comportement des autres crimes, nous avons évincé de l'analyse les types de crimes suivants :

Arson Incendie volontaire

Fraud Fraude

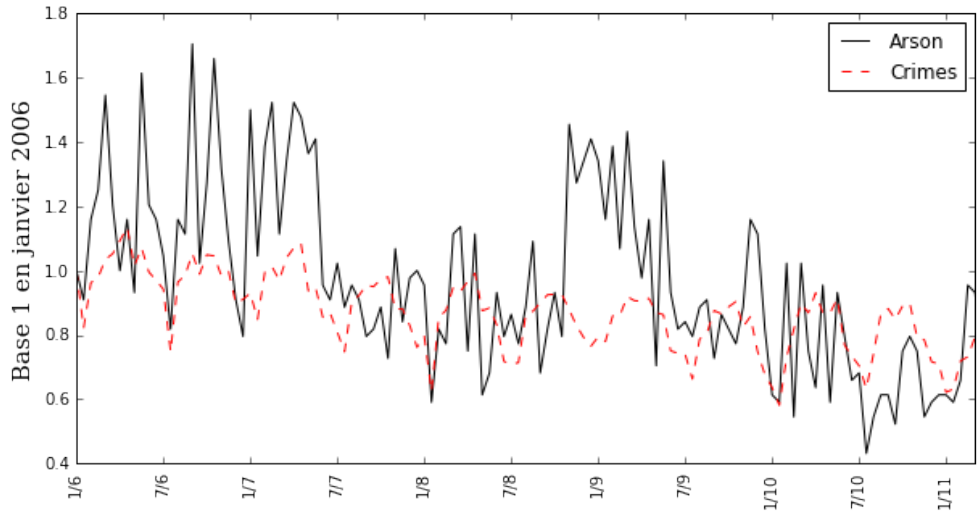
Vagrancy/Loitering Vagabondage

Rape viol

Other Sex Offenses (Not Commercialized) Autre types d'agressions sexuelles

Il n'est d'ailleurs pas étonnant que ces types de crimes soient plus aléatoires que les autres et dépendent moins de l'environnement.

FIGURE 8 – Evolution de Arson comparée a celle de Crimes (Arson = 0.003% de Crimes)



3.2.4 Etude des variables sociodémographiques

L'un des intérêts est de voir les effets de la crise. Le taux de chômage, par exemple, a explosé dans les années les plus dures de la crise, avant de baisser dernièrement. De même, on peut voir que certains secteurs d'activité ont été très fortement frappés, comme les secteurs financier, du BTP ou encore les emplois ouvriers.

En revanche, on observe en 2009 une hausse brutale de l'emploi, qui semble stabiliser le chômage. Cette hausse concorde avec une hausse assez nette du nombre d'emplois dans le secteur d'activité que nous avons appelé "Government", et qui peut s'apparenter à la fonction publique. Nous pensons que cet effet est le reflet d'une politique de dépenses publiques et de sauvetage de l'emploi organisée par l'Etat américain au début de la crise.

Ainsi, une augmentation des emplois liés à la fonction publique semble traduire une réaction à une situation défavorable. Nous aurions pu intuitivement penser qu'une telle augmentation irait de pair avec un environnement globalement sain. D'ailleurs, l'étude de la matrice des corrélation

met en évidence que le taux de crimes est positivement corrélé au nombre d'emplois dans le secteur public. Ce qui semble bien traduire le fait que l'emploi gouvernemental se fait en réaction à une situation de crise. Ce ne sera donc pas une bonne variable explicative pour prédire le taux de crimes.

3.2.5 Création des variables laggées et étude de la matrice des corrélations

Avant d'étudier la corrélation entre les variables, nous avons créé des variables laggées, ce qui nous semblait logique dans le cadre du traitement d'une série temporelle. Nous en avons notamment profiter pour créer un certain nombre de variables différenciées, et lagger celles-ci par la même occasion.

Les algorithmes de machine learning nécessitent d'être très attentif aux corrélations entre les variables. Par exemple, remarquons que les variables laggées Unemployment_rate sont toutes très corrélées positivement entre elles. Plus globalement, nous pouvons observer les effets de la saisonnalité dans les corrélations. Les variables sont souvent très corrélées avec leur variable laggée de 12 mois, beaucoup moins avec celle laggée de 6 mois.

La figure 18 en annexe représente la matrice de corrélation.

Notre dernier notebook est exclusivement consacré à l'implémentation des modèles. Pour les construire, nous nous sommes beaucoup référés à notre matrice des corrélations afin de sélectionner les variables pertinentes.

4 Les modèles

Pour le détail du code, se référer au notebook "Modèles (Philadelphie)"

Dans cette dernière partie, on s'attaque enfin aux modèles de prédiction. Notre premier modèle est un modèle ARIMA, choix qui semble logique pour l'étude d'une série temporelle. Les autres modèles sont des modèles de machine learning avec une random forest et un modèle SVR.

4.1 ARIMA

Le premier modèle n'est pas un modèle de machine learning. C'est un modèle ARIMA ("Auto Regressive – Integrated – Moving Average"). Pour rappel, un modèle ARIMA prend en compte trois paramètres *(p,d,q)* :

p est le nombre de termes auto-régressifs

d est le nombre de différences

q est le nombre de moyennes mobiles

Nous devrons donc identifier ces 3 paramètres pour notre série temporelle.

Le premier paramètre que nous pouvons déduire est le paramètre d, qui correspond à l'ordre de différenciation pour rendre notre série stationnaire. Dans notre cas, d=1.

Pour les deux autres paramètres, on introduit les fonctions d'autocorrélation et d'autocorrélation partielle de la série temporelle.

La fonction d'autocorrélation mesure la corrélation entre la série et une version laggée d'elle-même. Cette fonction permet de déterminer le paramètre q. La fonction d'autocorrélation partielle mesure la corrélation entre la série et une version laggée d'elle-même en ayant pris soin d'éliminer les effets déjà expliqués par les variables précédant la version laggée de la série.

Au vu des fonctions d'autocorrélation représentées en figure 5, on devrait choisir $p = q = 1$. Toutefois, les prédictions en utilisant ces valeurs donnent des résultats très mauvais. Nous avons donc décidé de prendre la valeur $p = 9$. Nous pouvons légitimement nous interroger sur les problèmes d'overfitting que cela engendre. Toutefois, rappelons que notre prédiction porte uniquement sur la ville de Philadelphie. Il n'y a donc pas de problème d'overfitting ici, ce qui ne serait évidemment pas le cas si nous transposions le modèle sur une autre ville comme Minneapolis (ce qui n'aurait pas beaucoup de sens pour ce modèle...).

A ce stade, nous pensions prédire la tendance et la saisonnalité avec un modèle ARIMA et utiliser du machine learning pour prédire les résidus. Les résultats obtenus sur la figure 9 étaient cependant relativement pauvres. De plus, nous avons essayé de prédire les résidus mais n'avons obtenu aucun résultat satisfaisant. Cela est sûrement dû à un manque de variables explicatives et à un comportement trop aléatoire des résidus. Comme on le voit sur la figure 9, même en prédisant parfaitement les résidus dans le modèle ARIMA, l'amélioration de la prédiction ne serait pas conséquente.

Nous nous sommes donc restreints à un model ARIMA prenant directement la série complète en entrée.

Nous pouvons d'ores et déjà dire que par rapport aux autres modèles que nous avons réalisés, ce modèle ARIMA performe moins bien à court terme mais mieux à long terme. Par la suite "modèle ARIMA" fera référence à ce dernier modèle.

FIGURE 9 – Résultats du premier modèle ARIMA

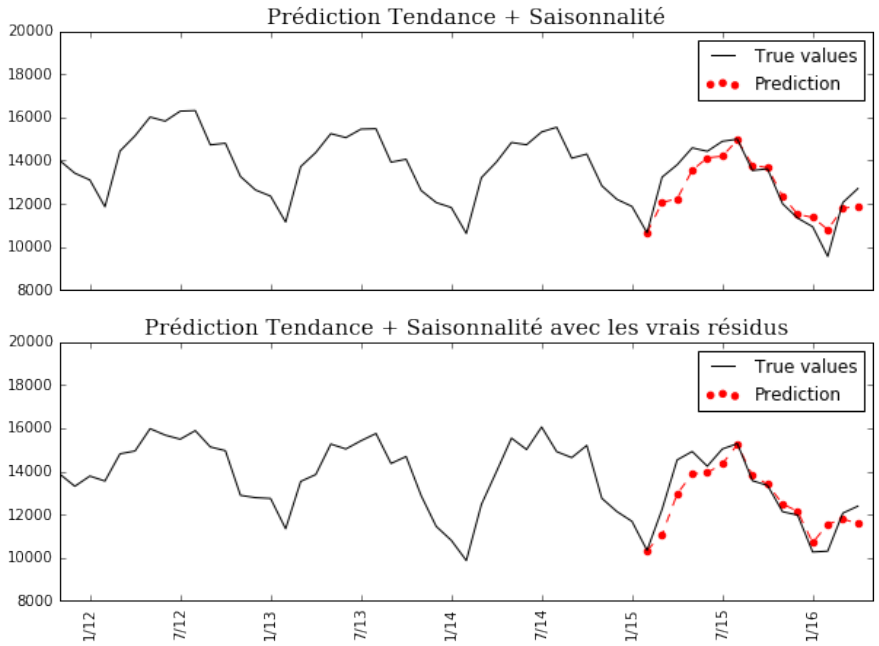
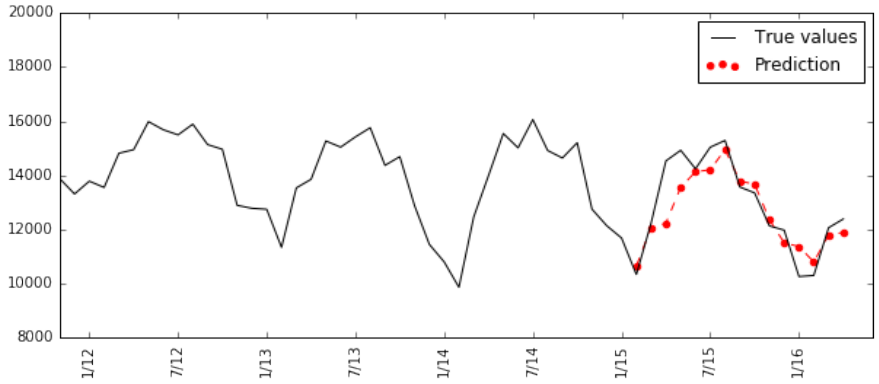


FIGURE 10 – Résultats du deuxième modèle ARIMA



4.2 Random Forest

Le deuxième modèle que nous avons utilisé est une Random Forest. La première étape consiste bien évidemment à sélectionner les variables à inclure dans le modèle.

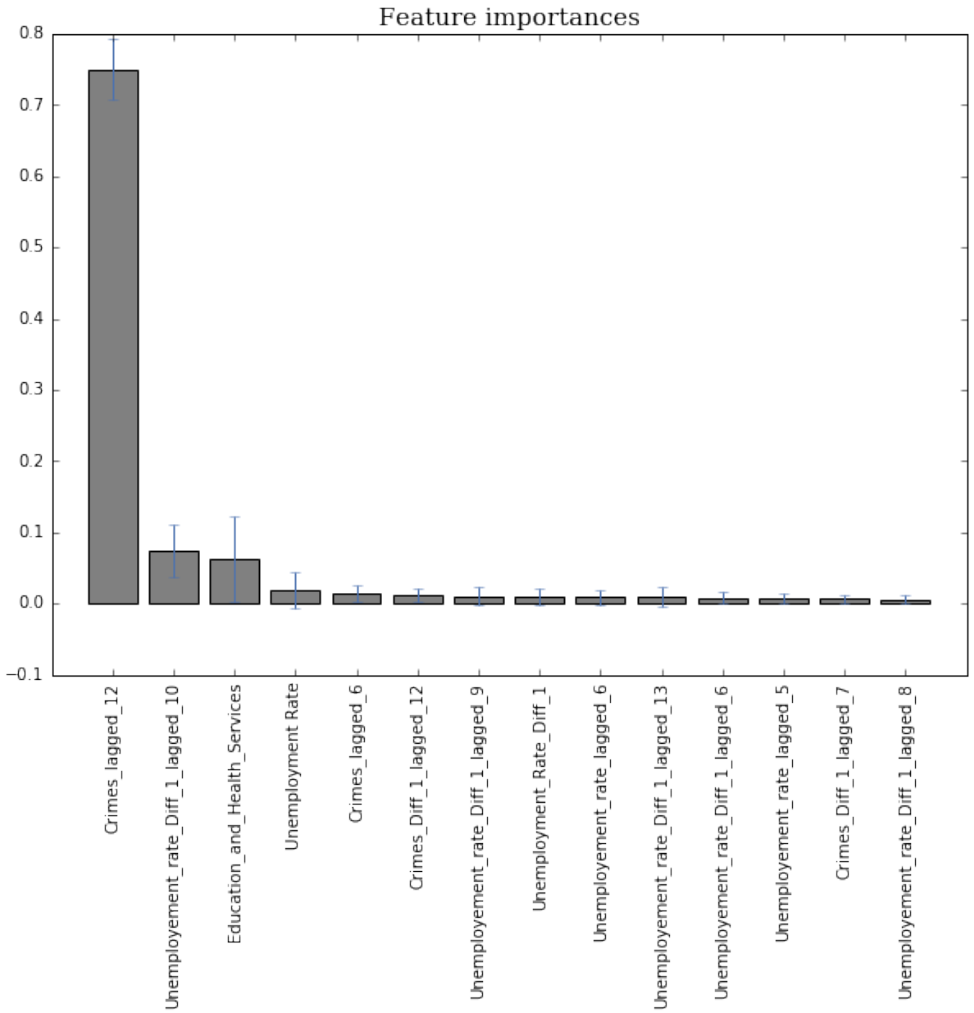
Pour ce faire, nous avons dans un premier temps fait tourner un modèle avec toutes les variables. Puis, en s'intéressant à l'ordre d'importance des variables, nous avons retiré progressivement les variables corrélées aux variables les plus importantes, en se référant à notre matrice des corrélations.

La variable la plus importante est la variable de crime laggée de 12 mois (nous ne pouvons pas inclure des variables laggées de moins de 12

mois pour éviter tout problème d'overfitting). Certaines variables incluses dans le modèle sont laggées de moins de 12 mois, mais elles ne sont pas significatives, ce qui ne pose en soi pas de problème majeur d'overfitting.

La deuxième variable contribuant le plus est "Education and Health Services". Toutefois nous nous sommes rendu compte à la fin que l'on n'est pas censé disposer de cette variable au moment où l'on veut prédire. En revanche, elle contribue à expliquer le crime : il y a donc clairement un rapport le entre taux de criminalité et cette variable.

FIGURE 11 – Graphe d'importance

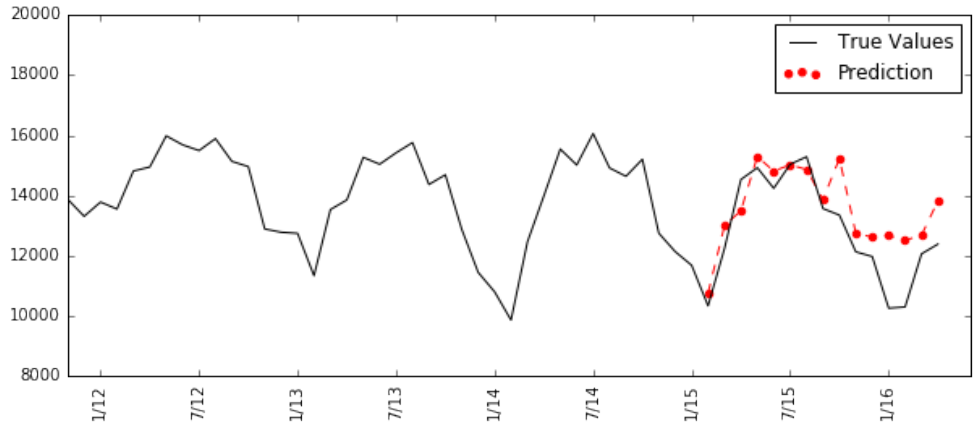


Faisons enfin une dernière remarque : bien que la variable la plus importante soit Crimes_lagged_12, d'autres variables ont une contribution significative. Ce modèle de machine learning apporte donc réellement quelque chose par rapport à un modèle ARIMA. Les résultats de la random forest sont bons à court terme mais moins bons à long terme. C'est

la raison pour laquelle on peut relativiser quant à l'inclusion de certaines variables laggées de moins de 12 mois. En effet, si nous devions utiliser ce modèle, nous l'utiliserions pour une prédiction à court terme, préférant un modèle ARIMA pour une prédiction à long terme.

De plus, il n'est pas anodin de remarquer que notre random forest se sert beaucoup de l'année précédente dans sa prédiction. Ce n'est pas étonnant quand on sait que la variable la plus contributive est Crimes_lagged_12. Ainsi, dans l'éventualité où la dernière année se comporterait «mal» par rapport aux autres années, les résultats fournis par la random forest seraient potentiellement meilleurs que ceux apportés par l'ARIMA. C'est d'ailleurs ce qu'on observe avec l'année 2015. Nuançons tout de même cette remarque : encore une fois, l'une des limites principales auxquelles nous avons fait face est le manque de variables. Peut-être que des variables laggées de plusieurs années contribueraient tout autant au modèle.

FIGURE 12 – Résultats de la Random Forest



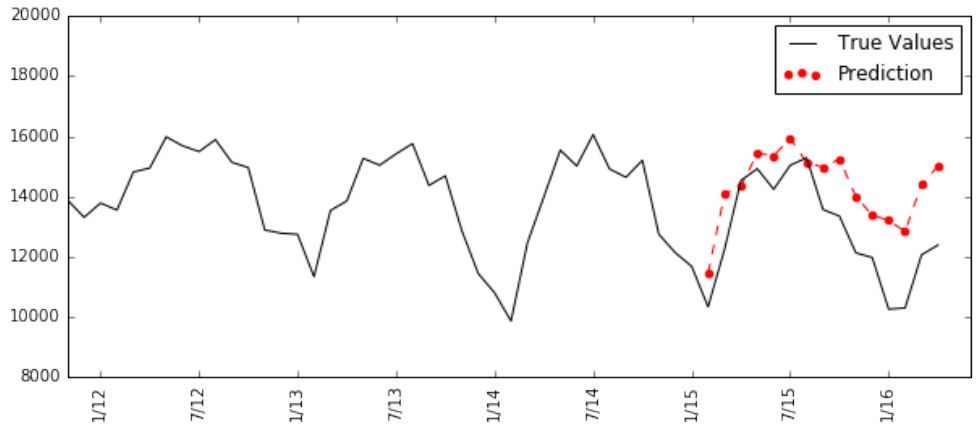
4.3 Support Verctor Regressor

Le troisième modèle que nous utilisons est un SVM (machine à vecteurs de support). Tuons de suite le suspens : les résultats ne sont pas bons. En effet, d'une part, l'algorithme a du mal à prédire au-delà de 6 mois. D'autre part, on remarque un saut dès le début de la prédiction (à très court terme).

Enfin, notons qu'il y a des problèmes d'overfitting au vu des variables laggées que nous avons retenues. Toutefois, c'est ce choix de variables qui nous donne les meilleurs résultats pour le modèle SVM. Nous vous épargnons donc les résultats obtenus avec les autres variables...

Nous avons passé très peu de temps sur ce modèle, comprenant assez rapidement que nous risquions de faire fausse route. Une explication possible est que ce choix de modèle est probablement inapproprié au vu des variables dont nous disposons. C'est surtout par curiosité que nous avons implémenté ce modèle.

FIGURE 13 – Résultats du SVR



4.4 Modèle combiné : modèle ARIMA et régression

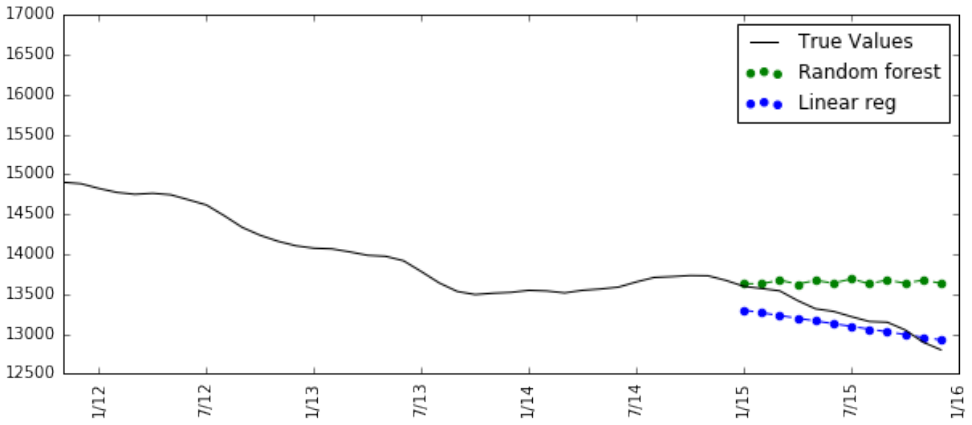
Dans celui-ci, nous avons essayé de combiner un modèle de prédiction de la saisonnalité et des résidus (saisonnalité et résidus étant regroupés) avec un modèle de prédiction de la tendance de la série temporelle. Dans le cadre de notre projet, l'idée consiste donc à prédire la saisonnalité et les résidus avec un modèle ARIMA et à prédire la tendance de la série avec une Random Forest. Toutefois, nous avons obtenu qu'une simple régression linéaire suffit pour prédire la tendance, celle-ci étant meilleure qu'une Random Forest.

Au final, ce dernier modèle combine une ARIMA et une régression linéaire sur la tendance. En comparaison aux précédents modèles, c'est celui qui nous donne globalement les meilleurs résultats.

Prédiction de la tendance

Comme nous l'évoquions à l'instant, nous avons prédit la tendance de la série de deux manières différentes. La première étant une Random Forest, la seconde étant une régression linéaire. Nous espérions que la Random Forest puisse capter les petites variations de la tendance, mais ce ne fut pas le cas. D'ailleurs, la variable contribuant le plus à la Random Forest est ni plus ni moins la variable de temps Time...

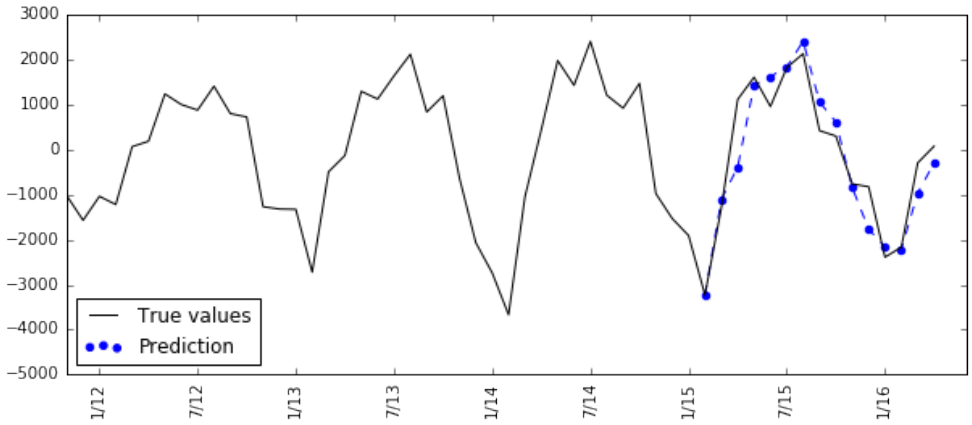
FIGURE 14 – Prèdiction de la tendance, $R^2 = 0.92$



Prédiction de la saisonnalité et des résidus

Pour prédire la saisonnalité et les résidus, nous utilisé une ARIMA, choix qui nous semblait le plus logique. Par souci de comparaison, nous avons aussi testé un modèle SVM, qui, sans grande surprise, performe moins bien que notre ARIMA (celle-ci détectant notamment mieux les creux dans la saisonnalité).

FIGURE 15 – ARIMA pour le saisonnalité et les résidus



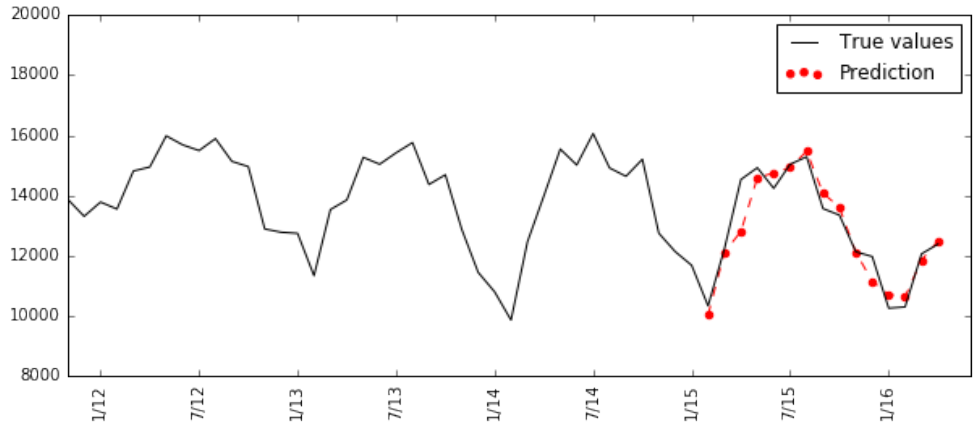
Combinaison

La combinaison de la régression linéaire (pour prédire la tendance) et de l'ARIMA (pour prédire la saisonnalité et les résidus) est notre meilleur modèle.

Remarquons toutefois quelque chose d'important. Dans l'année précédant notre année de prédiction, la série se comporte de façon assez

particulière. En effet, contrairement aux autres années, le crime connait une hausse brutale sans ralentissement dans les premiers mois, ce qui est aussi le cas pour l'année à prédire ! On voit clairement que notre modèle ne tient pas compte de ce changement, contrairement à l'algorithme de Random Forest.

FIGURE 16 – Résultat du modèle combiné



Choisir entre les deux modèles n'est donc pas tout à fait évident. Les deux présentent leurs avantages et leurs inconvénients. Ainsi, la Random Forest tient énormément compte de l'année précédente, tandis que le modèle combiné repose sur une tendance observée sur plusieurs années précédentes. A long terme, le modèle combiné serait probablement plus appréciable tandis qu'à court terme, le choix serait plus cornélien ! C'est donc en toute logique que nous proposons -dans une dernière et courte partie- une étude comparative des erreurs des différents modèles.

5 Comparaison des modèles

Dans l'optique de comparer les performances de nos différents modèles, nous avons créé une fonction de coût, qui n'est autre qu'une fonction de perte quadratique cumulée.

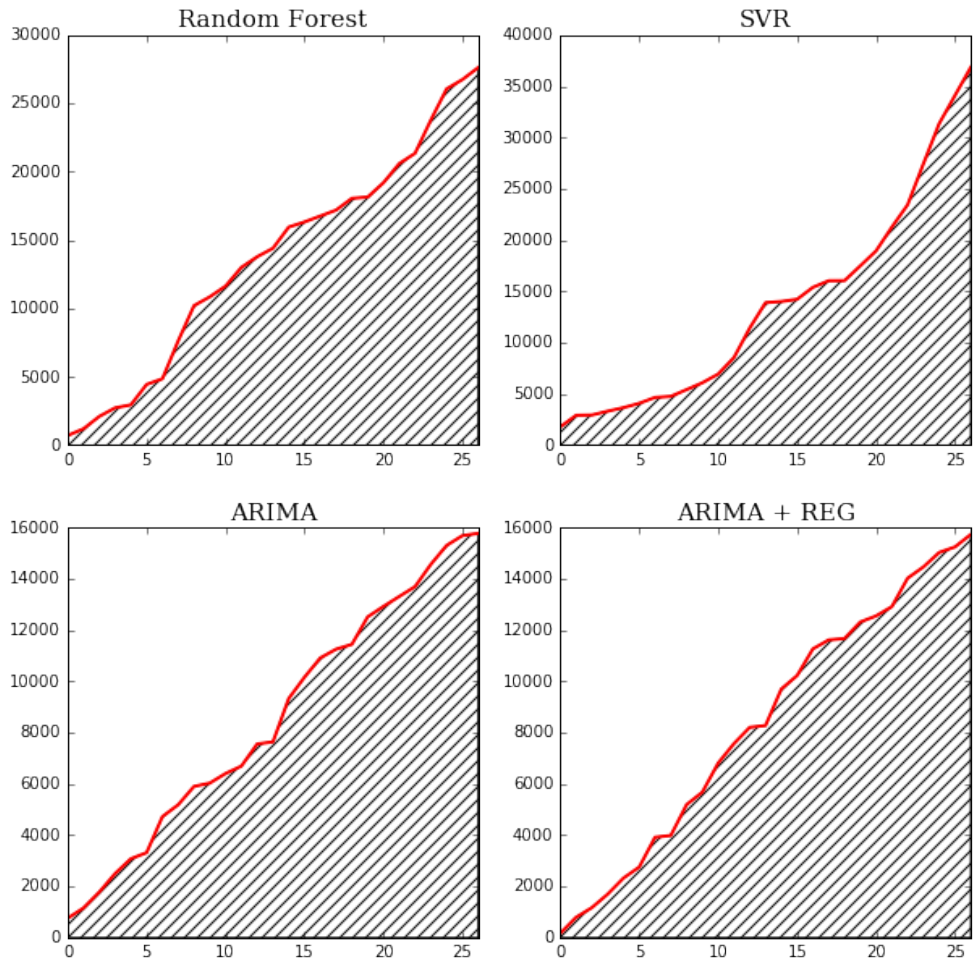
$$Cost(pred) = \sqrt{\int_{t_1}^{t_2} (pred(t) - true(t))^2 \, \mathrm{d}t}$$

On peut ainsi étudier le nombre d'erreurs de prédiction réalisées par les modèles, mais aussi la perte cumulée (dans l'optique de juger la qualité de la prédiction sur le long terme). En outre, nous avons aussi comparé les performances des modèles sur une échelle de prédiction de deux ans.

Le premier modèle (ARIMA) a du mal à prédire les premiers mois d'une année. Comme nous l'expliquions, celui-ci dépend de la forme des années précédentes. Or dans notre cas, les deux dernières années (donc celle juste avant notre prédiction, ainsi que l'année de notre prédiction) se comportent différemment. En effet, le ralentissement de l'augmentation du crime qui était observé pour les autres années n'est ici pas présent. Le modèle a donc du mal à prédire les petites variations.

Ce n'est pas forcément le cas avec un algorithme de machine learning. Dans notre cas, nous manquons malheureusement cruellement de variables (ce qui constitue la principale limite de notre travail), mais avec le peu dont nous disposons, on peut remarquer qu'un modèle comme la Random Forest «comprend» les spécificités et les changements liés à une année.

FIGURE 17 – Fonction de coût par modéle



Globalement, les modèles ARIMA ont de bonnes performances pour de la prédiction à long terme, l'erreur cumulée étant alors linéaire. Ce

n'est pas le cas du modèle SVR dont l'erreur explose avec le temps. Enfin, la Random Forest performe relativement bien à court terme mais moins bien à long terme. L'étude graphique permet de rendre compte des différences de performances entre les modèles.

6 Conclusion

De notre étude, il ressort que le modèle le plus performant globalement n'est pas un modèle de machine learning mais un modèle ARIMA, couplé à une régression linéaire pour prédire la tendance. Ce modèle est notamment performant pour de la prédiction à long terme, son erreur restant stable au cours du temps.

Notre Random Forest, quant à elle, donne une prédiction relativement bonne à court terme, l'erreur étant moins bonne à long terme. Toutefois, notons qu'un modèle de machine learning prend en compte de nombreuses variables, sociodémographiques et socioéconomiques notamment. Tirer des conclusions hâtives sur le manque de performances de ce modèle n'est pas judicieux. En effet, et c'est la principale limite de notre projet, nous manquons de variables. Intégrer des variables d'éducation, de santé, ou encore de pauvreté aurait probablement changé la donne et nous aurait permis non seulement de prédire le crime, mais aussi -et c'est le plus important- d'étudier ses déterminants.

C'est aussi ce manque de variables qui nous a conduits à inclure certaines variables qui peuvent générer de l'overfitting. Toutefois, nous avons pris garde à ne conserver ces variables que si elles avaient un niveau de significativité très faible. Ainsi, nos modèles ne sont en l'état exploitables que pour la ville de Philadelphie. En revanche, il est tout à fait possible de les reproduire pour d'autres villes en les optimisant de façon adéquate.

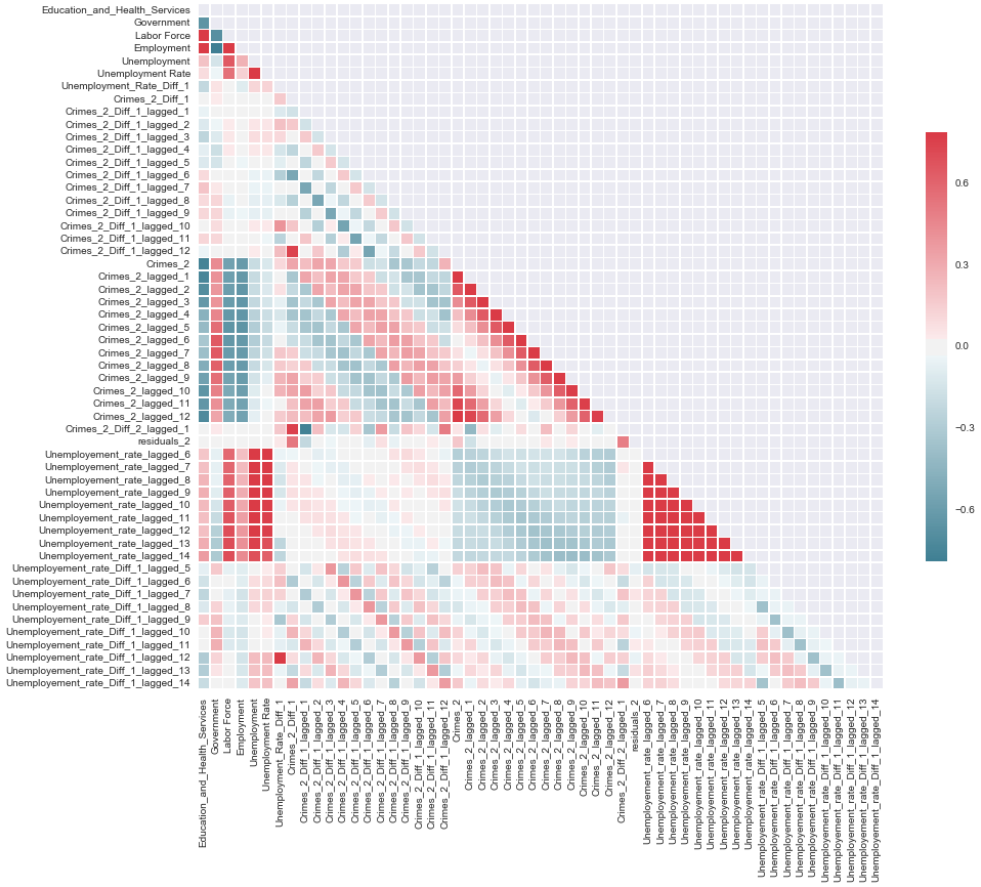
Selon nous, la meilleure façon de tirer profit de nos modèles est la suivante : utiliser le modèle combinant une ARIMA et une régression linéaire pour une prédiction de long terme, et utiliser ce modèle ainsi que la Random Forest pour de la prédiction à court terme (de l'ordre de quelques mois à un an). Les bénéficiaires de tels modèles pourraient alors être multiples. Policiers évidemment, mais aussi promoteurs immobiliers, acheteurs potentiels, mais aussi le gouvernement dans le cadre de politiques publiques.

Terminons enfin par un mot plus personnel. Nous avons pris plaisir à travailler sur ce projet, qui nous a demandé du temps et beaucoup d'investissement. Consigne était donnée de nous amuser, nous pensons de ce point de vue là avoir rempli le contrat. Bien qu'ayant touché du doigt seulement les perspectives offertes par l'analyse de données et le

machine learning, ce travail aura été une occasion en or pour progresser et découvrir. Enfin, nous espérons que la lecture de cette synthèse et des notebooks vous aura plu.

7 Annexes

FIGURE 18 – Matrice des corrélations



Listing 1 – Code du model ARIMA

```
from statsmodels.tsa.seasonal import seasonal_decompose
decomposition = seasonal_decompose(np.array(df.Crimes),
        freq=12)

df['trend'] =  decomposition.trend
df['seasonal'] =  decomposition.seasonal
df['residuals'] =  decomposition.resid

df['Crimes_clean'] = df.trend+df.seasonal

ts = np.array(df.Crimes_clean.astype(float))[6:124]
ts_residuals = np.array(df['residuals'])[6:124]

model = ARIMA(ts+ts_residuals, order=(9, 1, 1))
results = model.fit(disp =-1)
pred = results.predict(start=104, end = 117,
```

```
        dynamic=True)
prediction = pred.cumsum()
predicted_ts = np.concatenate((ts[103].reshape(1,),
        prediction+ts[103]))
```

Listing 2 – Code de la Random Forest

```
X = np.array(df[['Education_and_Health_Services',
                 'Unemployment␣Rate',
        'Crimes_lagged_6','Crimes_lagged_12',
        'Crimes_Diff_1_lagged_12',
        'Crimes_Diff_1_lagged_7',
         'Unemployement_rate_lagged_5',
        'Unemployement_rate_lagged_6',
        'Unemployment_Rate_Diff_1',
        'Unemployement_rate_Diff_1_lagged_6',
        'Unemployement_rate_Diff_1_lagged_8',
        'Unemployement_rate_Diff_1_lagged_9',
        'Unemployement_rate_Diff_1_lagged_10',
        'Unemployement_rate_Diff_1_lagged_13']])

features_names = ['Education_and_Health_Services',
                 'Unemployment␣Rate',
        'Crimes_lagged_6','Crimes_lagged_12',
        'Crimes_Diff_1_lagged_12',
        'Crimes_Diff_1_lagged_7',
         'Unemployement_rate_lagged_5',
        'Unemployement_rate_lagged_6',
        'Unemployment_Rate_Diff_1',
        'Unemployement_rate_Diff_1_lagged_6',
        'Unemployement_rate_Diff_1_lagged_8',
        'Unemployement_rate_Diff_1_lagged_9',
        'Unemployement_rate_Diff_1_lagged_10',
        'Unemployement_rate_Diff_1_lagged_13']

Y = np.array(df.Crimes)

X_train = X[16:110]
Y_train = Y[16:110]
X_test = X[109:124]
Y_test = Y[109:124]

RandomForest = RandomForestRegressor(n_estimators=10,
        min_samples_split=2,oob_score = True)
RandomForest.fit(X_train,Y_train)

Y_pred = RandomForest.predict(X_test)
```

Listing 3 – Code du SVR

```
from sklearn import svm
```

```
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline

X = np.array(df[['Education_and_Health_Services',
        'Unemployment␣Rate',
     'Crimes_lagged_6',
     'Crimes_lagged_12',
     'Crimes_Diff_1_lagged_12',
     'Crimes_Diff_1_lagged_7',
     'Unemployement_rate_lagged_5',
     'Unemployement_rate_lagged_6',
     'Unemployment_Rate_Diff_1']])

Y = np.array(df.Crimes)
Mean = Y.mean()
Std = Y.std()
Y = (Y-Y.mean())/Y.std()

X_train = X[16:110]
Y_train = Y[16:110]
X_test = X[109:124]
Y_test = Y[109:124]

Scaler = StandardScaler()
Svr = svm.SVR()
Regressor = Pipeline([('Scaler', Scaler),('Svr',Svr)])

Regressor.fit(X_train,Y_train)

Y_pred = Regressor.predict(X_test)

Y = Y*Std+Mean
Y_pred = Y_pred*Std+Mean
```

Listing 4 – Code pour l'erreur

```
def cost(pred,true):
    fig, ax = plt.subplots(1,2,figsize=(15,5))
    cost = np.sqrt((pred-true)*(pred-true))
    ax[0].plot(cost,color = 'r', linewidth= 2)
    ax[0].set_title('Perte␣par␣moi')
    cost = cost.cumsum()
    ax[1].plot(cost,color = 'r', linewidth= 2)
    ax[1].fill_between(range(0,np.shape(cost)[0]),
        0, cost,hatch = '///')
    ax[1].set_title('Perte␣cumulee')
```