

Rapport Projet Runlight

Projet réalisé dans le cadre du projet de semestre de la majeure Ingénierie et numérique



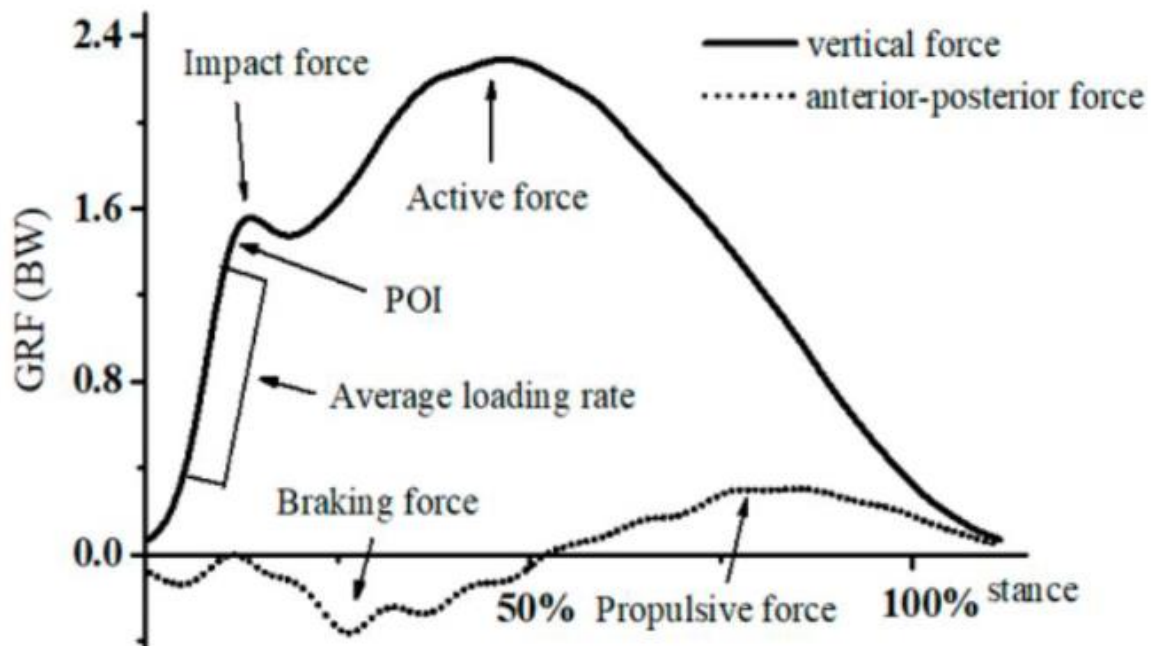
Nissrine AKKIYAT – Tristan Bourhis - Valentine Mercent – Ying Ren - Meini Xue

Sommaire

Sommaire	2
• Introduction.....	3
• Présentation de l'équipe	4
• Organisation du travail	4
• Diagramme de Gantt	4
• Traitement des données.....	6
1. Contextualisation	6
• Format des données.....	6
• Préanalyse	6
2. Régression polynômiale	7
• Définition.....	7
• Les avantages de la régression polynomiale	7
• Les inconvénients de la régression polynomiale.....	8
• Résultats d'entraînement de modèle Régression polynomiale	9
3. SVM « RBF »	12
• Définition.....	12
• Les avantages des SVM	12
• Les inconvénients des SVM	12
• Résultats pour le SVM RBF	13
4. Apprentissage profond par CNN	15
• Définition.....	15
• Les avantages d'un réseau de neurones à convolution	16
• Les inconvénients d'un réseau de neurones à convolution	16
• Résultats des tests.....	17
5. Comparaison et analyse des résultats.....	20
• Quel est le meilleur modèle ?	20
• Les résultats sont-ils assez bon ?.....	21
6. Axes d'amélioration possibles	22
• Modèles plus complexes	22
• Créations de nouvelles données artificielles.....	22
• Récolter des nouvelles données.....	22
• Prise en compte de contrainte de régularisation.....	22
Conclusion	23

• Introduction

Le projet porte sur la façon dont on peut prédire la force des pas effectués par des sportifs lors de leur course. Il est actuellement possible de mesurer la force exercée contre le sol lors de sa course. On voit par exemple sur ce graphique la force qu'il exerce lors d'un pas.



Cela permet au coureur d'analyser sa course et d'essayer de s'améliorer. Cependant pour obtenir ces données il est nécessaire de courir sur une plaque très coûteuse, et à part des plaques isolées, il n'existe qu'une piste avec ces plaques, qui est longue de 50 mètres. Ce qui n'est pas suffisant pour analyser une course complète.

Notre objectif est donc d'obtenir cette courbe donnée par la plaque en utilisant des petits capteurs que l'on place sur le coureur et, grâce à un modèle d'intelligence artificielle, de retrouver ces données.

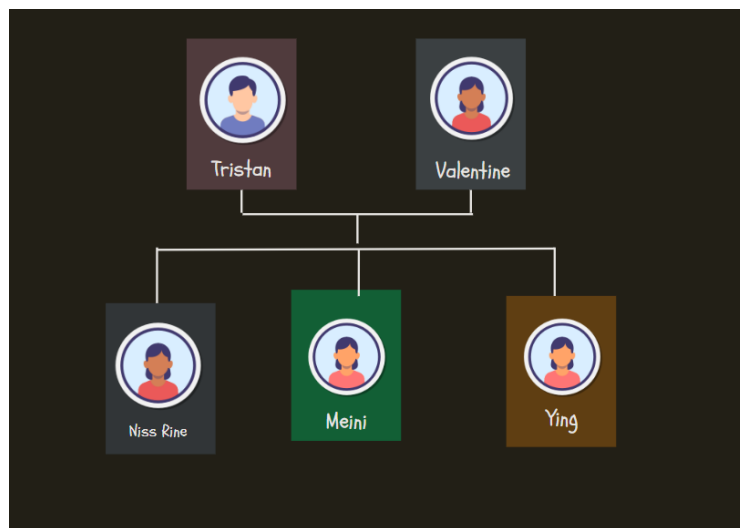
Pour cela, on munit des sujets de 2 capteurs (accéléromètres) positionnés à deux endroits différents du corps (au niveau des lombaires, et du tibia), chacun d'entre eux nous donnera les accélérations selon 3 axes différents (6 données au total). Une plaque nous permet de récupérer la courbe que l'on souhaite retrouver en sortie. Les sujets vont alors courir en posant leurs pieds sur la plaque afin d'obtenir une base de données.

On possède deux bases de données l'une décrivant pour chaque capteur les données d'un individu lors de 10 séances et 10 passages (DATASUBJECTRAW). L'autre en revanche, décrit le comportement de 10 individus lors de 10 passages (DATATESTRAW). Nous avons dans un premier temps travaillé avec la base de données relative à un seul sujet : on a pour cela divisé les 10 passages du sujet entre l'entraînement (8) et le test (2). Pour résumer, nous entraînerons les modèles d'IA sur 80% de la base de données, et nous testerons ces modèles sur les 20% restant, qui n'ont pas servi pour l'entraînement.

Nous avons choisi de nous intéresser à trois modèles d'IA différents en Machine comme en Deep Learning : la régression polynomiale, le modèle des machines à vecteur de support (SVM) ainsi qu'au CNN (Convolutional Neural Network). Pour chaque modèle que nous expliciterons, nous présenterons ces avantages et ses inconvénients ainsi que les résultats qui ont été obtenus.

- **Présentation de l'équipe**

Notre équipe était composée de deux chefs de projet : Tristan et Valentine ainsi que de trois développeuses : Meini, Ying et Nissrine.



- **Organisation du travail**

- **Diagramme de Gantt**

Nous nous sommes réparti le travail selon le diagramme de Gantt suivant. Tristan et Ying se sont chargés de la partie SVM, Meini et Valentine de la partie CNN et Nissrine de la partie Régression polynomiale. Toutes les tâches ont été menées à bien.

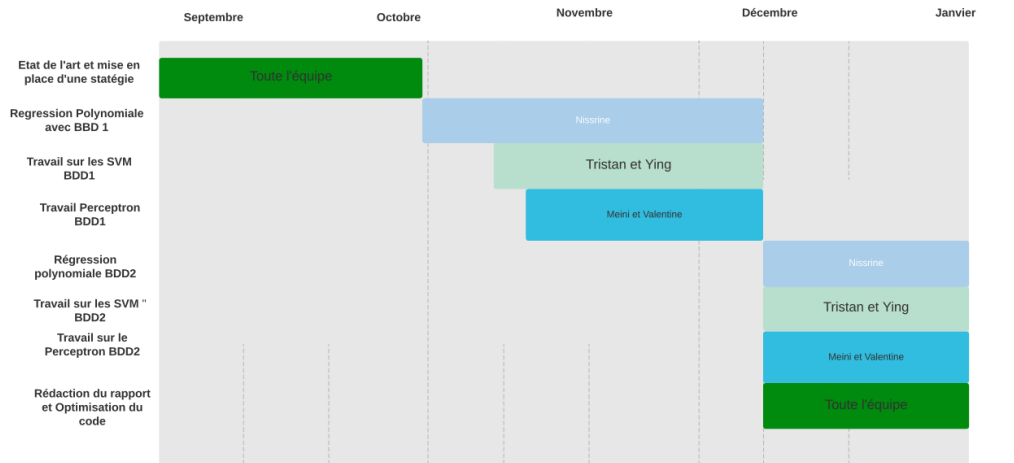


Figure 1 : Diagramme de Gantt illustrant la répartition du travail

- **Traitement des données**

Il existe plusieurs façons de traiter les données selon si on choisit d'utiliser le Machine Learning ou le Deep Learning.

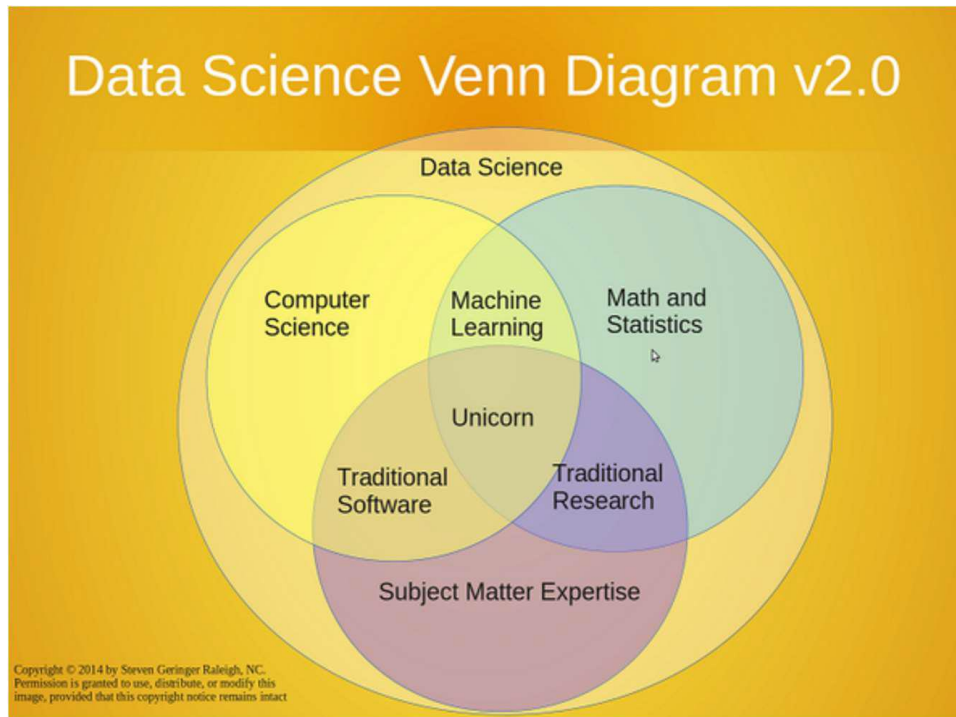


Figure 2 : Diagramme de Venn pour Data Science

1. Contextualisation

- **Format des données**

Comme expliqué dans l'introduction, les capteurs nous donneront 6 données, qui nous serviront d'entrées pour les modèles, et la plaque nous fournit une donnée, qui servira de sortie. Nous utiliserons donc une entrée à 6 dimensions pour prédire la courbe de sortie.

Il faut savoir que ces données ont été au préalable traitées, des filtres de fréquences passe-bas ont été appliqués dessus afin de multiplier les données. Nous avons donc les données brutes, puis celles filtrées de 5 Hertz à 50 Hertz (avec un pas de 5H).

- **Préanalyse**

Le problème étant que ces données filtrées sont trop différentes et nous ne pouvons pas les mélanger pour entraîner nos modèles, nous devons trouver lesquelles nous donnent les meilleurs résultats.

On a donc testé les données en créant un modèle d'IA différent pour chaque fréquence, en l'entraînant sur le passage d'une personne et en le testant sur le même passage. Nous avons fait cela pour tous les modèles d'IA que l'on vous présentera.

Le résultat fut le même sur tous les différents modèles, ils étaient plus performants sur les données avec un filtre de 20Hertz, c'est donc sur celle-ci que nous allons nous concentrer.

2. Régression polynômiale

- **Définition**

La régression polynomiale est une méthode utilisée pour modéliser une relation entre une variable indépendante x et une variable dépendante y . Elle consiste à utiliser une fonction polynômiale (issue d'un polynôme) pour faire une approximation de données. Cela permet de mettre en exergue des relations non linéaires entre les variables x et y qui ne peuvent pas être décrites par une simple droite de régression linéaire. La régression polynomiale est souvent utilisée dans les domaines de la statistique et de l'analyse de données, notamment pour ajuster des modèles à des données complexes.

Une variable indépendante est une variable qui est utilisée pour expliquer ou prédire une autre variable. Elle est généralement considérée comme la cause ou le facteur de la variation de la variable dépendante.

Une variable dépendante est une variable qui est étudiée ou mesurée dans une expérience ou une analyse. Elle est généralement considérée comme l'effet ou la conséquence de la variation de la variable indépendante.

Simple Linear Regression	$y = b_0 + b_1x_1$
Multiple Linear Regression	$y = b_0 + b_1x_1 + b_2x_2 + \dots + b_nx_n$
Polynomial Linear Regression	$y = b_0 + b_1x_1 + b_2x_1^2 + \dots + b_nx_1^n$

Figure 3 : les différents types de régression

- **Les avantages de la régression polynomiale**

La régression polynomiale présente des avantages certains :

- **Capacité à illustrer des relations non linéaires :**

Les fonctions polynômiales peuvent prendre des formes très variées, ce qui permet de modéliser des relations non linéaires entre les variables x et y

- **Simplicité de la méthode :**

La régression polynomiale est simple à appréhender et à mettre en œuvre car elle repose sur l'utilisation de fonctions polynômiales simples

- **Faible coût computationnel :**

La régression polynomiale nécessite peu de calculs pour ajuster un modèle, ce qui la rend efficace pour des jeux de données de grande taille

- **Flexibilité :**

La régression polynomiale permet de modéliser des relations complexes entre les variables x et y , en utilisant des termes polynomiaux de différents degrés. Cela permet de capturer des tendances à la fois simples et complexes dans les données.

Dans notre cas le plus grand avantage c'est que l'entraînement de modèle prends quelle que minute on compare avec la méthode du CNN dont le temps de compilation représente des heures.

- **Les inconvénients de la régression polynomiale**

La régression polynomiale présente également certains inconvénients :

- **Sur-ajustement potentiel**

Si le degré du polynôme utilisé est trop élevé, il peut y avoir un sur-ajustement des données d'entraînement, ce qui entraînera des résultats médiocres sur des données de test

- **Sensibilité aux fluctuations de données**

Les fonctions polynômiales peuvent être fortement influencées par des valeurs aberrantes ou du bruit dans les données et cela peut affecter la qualité du modèle

- **Instabilité numérique**

Lorsque les données contiennent des valeurs extrêmes, la résolution numérique des équations polynômiales peut devenir instable, ceci peut entraîner des résultats erronés

- **Difficulté d'interprétation**

Les modèles polynomiaux peuvent être difficiles à interpréter car ils peuvent contenir un grand nombre de termes et de coefficients, il est alors difficile de comprendre les relations entre les variables

- **Coût de calcul élevé**

Pour des modèles polynomiaux de haut degré, le coût de calcul peut devenir élevé, ce qui peut rendre difficile l'application de la régression polynomiale sur des jeux de données volumineux

• Résultats d'entraînement de modèle Régression polynomiale

La partie la plus importante dans la régression polynomiale c'est trouver le degré de polynôme, on a essayé plusieurs degrés d de $d = 2$ jusqu'au $d = 10$ et on a trouvé que le degré, $d = 3$ et le plus optimal dans notre cas.

Aussi on a utilisé Le LassoCV c'est un module de la bibliothèque scikit-learn qui implémente une régression Lasso à pénalité croisée. La régression Lasso est une méthode de régression linéaire qui utilise une pénalité L1 pour réduire la complexité du modèle en sélectionnant automatiquement un sous-ensemble de variables explicatives. La pénalisation croisée consiste à ajuster automatiquement le niveau de pénalisation en utilisant une stratégie de validation croisée. Cela permet de choisir automatiquement le meilleur niveau de pénalisation en fonction des données d'entraînement.

Voici comment lire les nomenclatures des données dans le tableau :

F : Correspond à la fréquence de filtrage des données

S : correspond au sujet

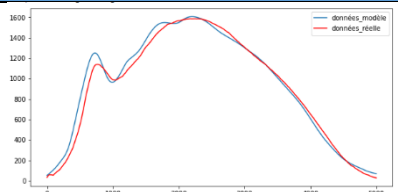
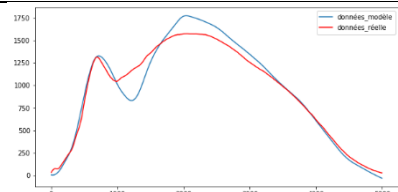
P : correspond au passage (un sujet fait 10 passage par séance)

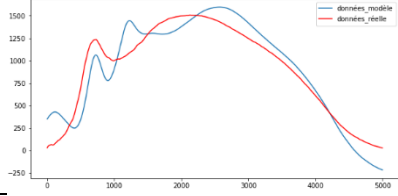
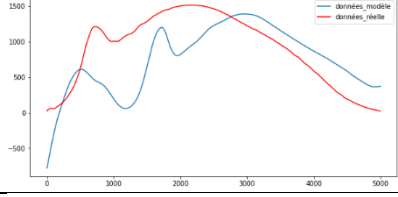
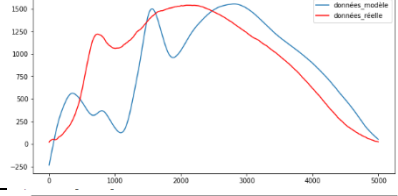
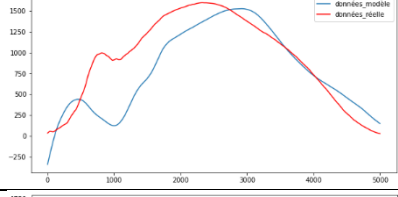
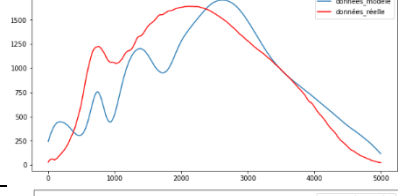
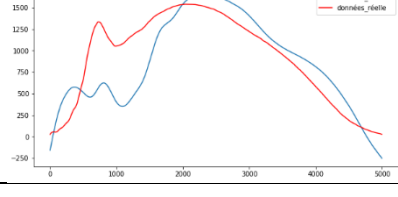
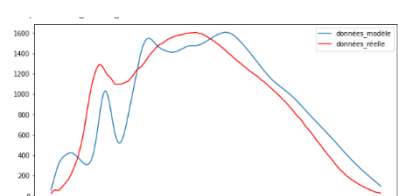
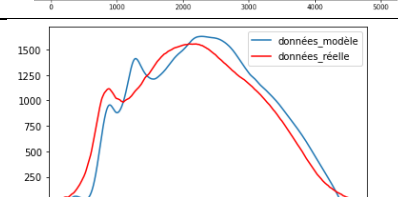
Dans le cas ou il manque un de ces paramètre, c'est que toutes les données correspondante ont été prises.

Dans le cas ou on a « 8P », ça signifie que 8 passages différents sont pris, et dans les cas où il y a « P8 », c'est qu'il n y a que le 8^{ème} passage qui a été pris

Les différents données (R2 et RMSE) permettent d'identifier la précision du modèle, donc sa performance. Plus R2 est proche de 1, plus le modèle est précis, inversement plus le RMSE est faible, plus le modèle est précis.

On superpose aussi les deux courbes, la courbe réelle (obtenue par la plaque) en rouge, et celle prédite par le modèle en bleu.

Dataset choisi (fréquence_ sujet_passage)	R2	RMSE	Résultats
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S1_P10	0.981366 (Test)	70.059926 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S2_P10	0.953643 (Test)	109.600371 (Test)	

Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S3_P10	0.870826 (Test)	176.57585 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S4_P10	0.094902 (Test)	472.185123 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S5_P10	0.367325 (Test)	401.73858 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S6_P10	0.551098 (Test)	342.28126 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S7_P10	0.715942 (Test)	286.74194 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S8_P10	0.567510 (Test)	333.59163 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S9_P10	0.798677 (Test)	235.65715 (Test)	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S10_P10	0.975208 (Test)	79.55121 (Test)	

D'après les résultats obtenus ci-dessus et comme l'un des inconvénients de la régression polynomiale est la **Sensibilité aux données imprécises**, les fonctions polynômiales peuvent être fortement

influencées par des valeurs aberrantes ou des bruits dans les données, ce qui a affecté la qualité du modèle. Donc on va passer à d'autres méthodes pour une optimisation maximale de notre modèle.

3. SVM « RBF »

- Définition

Les Support Vector Machines sont appelés en français : Séparateur à Vaste Marge (SVM). Ils font partie des algorithmes d'apprentissage automatique. Ceux-ci sont initialement utilisés pour la discrimination (la prédiction d'une variable qualitative binaire). Les SVMs permettent de résoudre des problèmes de classification, de régression ou encore de détection d'anomalie. Ils sont connus pour leur fiabilité.

Le fonctionnement d'un SVM utilisant un kernel RBF consiste à d'abord transformer les données d'entrée à l'aide de la fonction RBF, puis à utiliser ces données transformées pour entraîner un modèle SVM standard. Le modèle SVM utilise ensuite ces données transformées pour faire des prédictions en utilisant des techniques de séparation de données.

Ce kernel est basé sur une fonction qui mesure la distance entre les entrées et un point central ou noyau

La fonction de noyau RBF est définie comme : $K(x, x_i) = \exp(-\gamma ||x - x_i||^2)$, où γ est un paramètre de réglage qui contrôle l'influence de chaque point dans les données d'entraînement sur les données transformées.

Lors de la phase d'entraînement, l'algorithme de SVM sélectionne les points de données qui sont les plus importants pour déterminer la frontière de décision, ces points sont appelés les vecteurs de support. Lors de la phase de prédiction, le modèle utilise ces vecteurs de support pour effectuer la classification en utilisant uniquement les données d'entrée les plus significatives.

- Les avantages des SVM

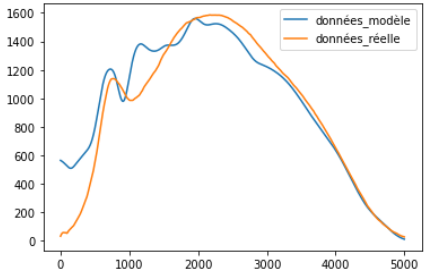
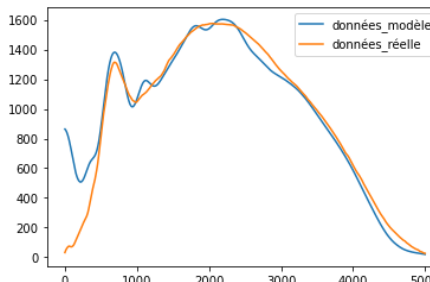
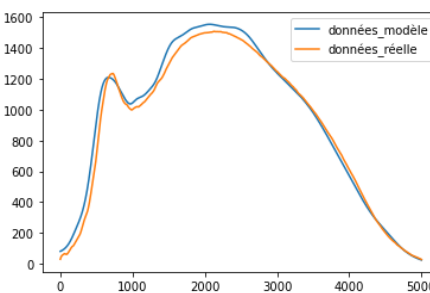
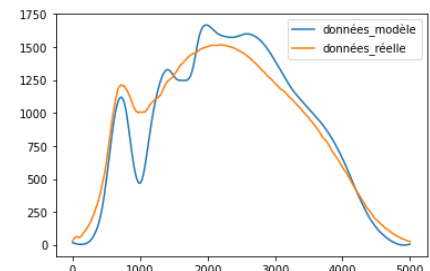
- Le modèle fonctionne relativement bien lorsqu'il existe une marge de séparation claire entre les classes.
- Le modèle est plus efficace dans les espaces à haute dimension.
- Le modèle est efficace dans les cas où le nombre de dimensions est supérieur au nombre d'échantillons.
- Le modèle est relativement efficace en termes de mémoire.
- Le modèle peut être utilisé pour la classification linéaire et non linéaire.

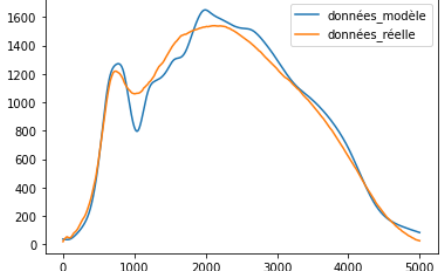
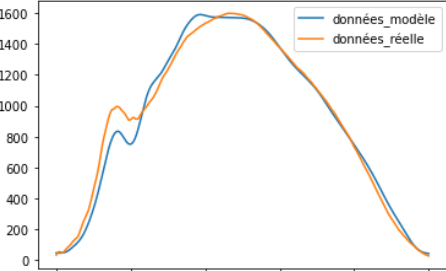
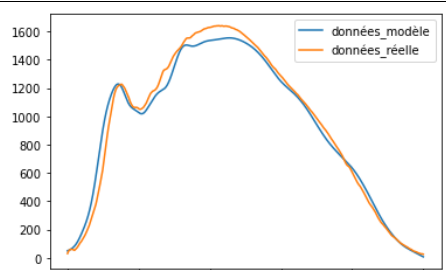
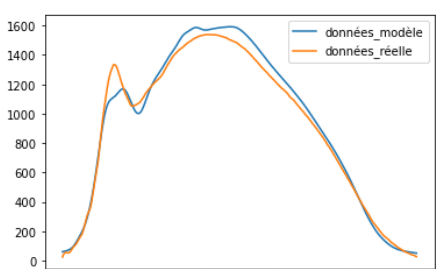
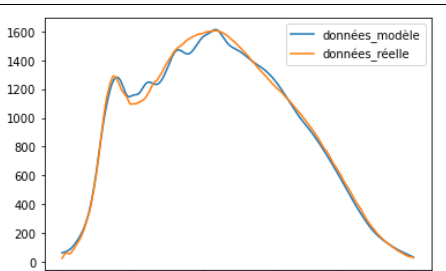
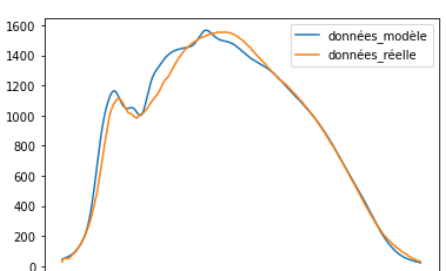
- Les inconvénients des SVM

- L'algorithme SVM ne convient pas aux grands ensembles de données.
- SVM n'est pas très performant lorsque l'ensemble de données comporte du bruit, c'est-à-dire lorsque les classes cibles se chevauchent.
- Dans les cas où le nombre de caractéristiques pour chaque point de données dépasse le nombre d'échantillons de données d'apprentissage, le SVM sera moins performant.

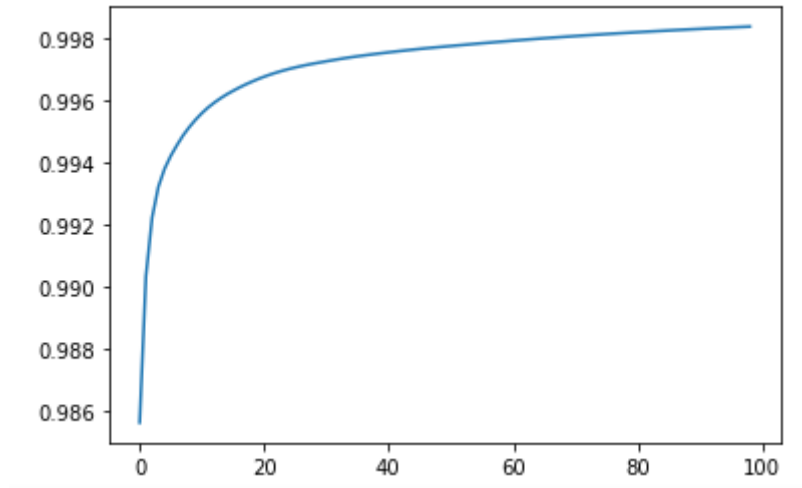
- Comme le classificateur à vecteur de support fonctionne en plaçant les points de données au-dessus et au-dessous de l'hyperplan de classification, il n'y a pas d'explication probabiliste pour la classification.

- Résultats pour le SVM RBF

Données (fréquence_sujet_pas sage)	R2	rmse	Résultats
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S1_P10	0.902099078835 4097	160.5884858 282137	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S2_P10	0.918909825472 932	144.9562900 1373624	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S3_P10	0.985156216740 2266	59.85715515 618273	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S4_P10	0.905203943680 0805	152.8127574 5130236	

Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S5_P10	0.977932057810 9062	75.02985381 811067	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S6_P10	0.981500408082 0301	69.48454868 302906	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S7_P10	0.982135535383 183	71.90888442 887709	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S8_P10	0.984734483462 7774	62.67333106 127908	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S9_P10	0.995493509808 203	35.25761593 762625	
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S10_P10	0.984211165308 6243	63.48518304 003747	

Une partie très importante de l'exécution du SVM est le réglage du paramètre C. Lorsque l'on utilise le code pour la prédiction, le meilleur paramètre du modèle est $C=100$, nous avons constaté que les modèles individuels ne donnaient pas de mauvais résultats lorsque $C=100$.



Dans un modèle linéaire de SVM, plus la valeur C est grande, moins le classificateur est disposé à admettre des erreurs de classification ("outliers"). Si la valeur C est trop grande, le classificateur essaiera de réduire les erreurs sur les données d'apprentissage, alors qu'en réalité cela est impossible ou dénué de sens, ce qui entraîne un surajustement.

Mais dans un modèle non linéaire, plus la valeur de C est grande, meilleur est le modèle.

4. Apprentissage profond par CNN

- Définition

L'apprentissage profond par les CNN (Convolutional Neuronal Network) est un réseau de neurones artificiels à plusieurs couches qui comprend des entrées et sorties. L'information (les données) circulent de la couche d'entrée vers la couche de sortie. Chaque entrée possède un poids et la sortie est une fonction du poids et des entrées. L'objectif des réseaux de neurones est de simuler le fonctionnement d'un cerveau humain.

C'est un modèle de Deep Learning au sein duquel on peut notamment influencer sur le nombre de neurones par couche. L'objectif est alors de déterminer le nombre de neurone parfait pour éviter le sur/sous-apprentissage. La particularité topologique de ce réseau est que tous les neurones d'une couche sont connectés à tous les neurones de la couche suivante

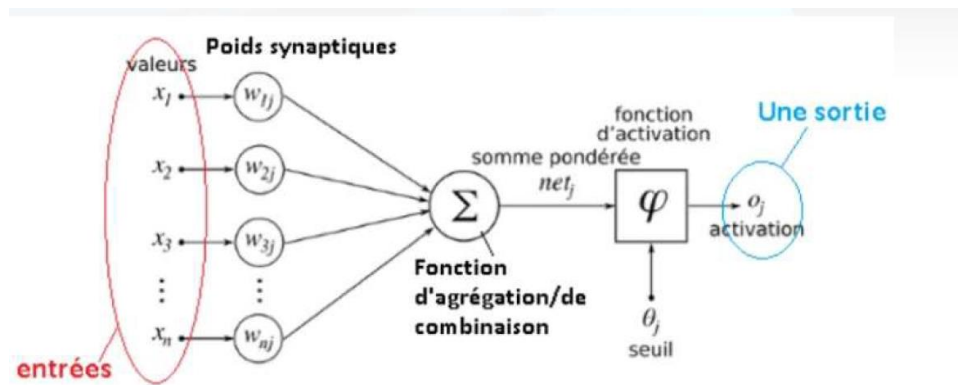


Figure 4 : Illustration du fonctionnement d'un neurone

- Les avantages d'un réseau de neurones à convolution

- 1- Scalabilité

Les CNN ont une capacité à être mis en œuvre sur de grandes bases de données

- 2- Partage des poids

Les filtres de convolution sont partagés par toutes les zones de la donnée d'entrée. Cela réduit considérablement le nombre de paramètres à prendre en compte, ce qui facilite l'apprentissage et réduit le risque de sur-apprentissage

- 3- Performances élevées pour les tâches de vision par ordinateur

Les CNNs sont par ailleurs particulièrement efficaces pour les tâches de reconnaissance d'images, de segmentation d'images et de classification d'images. Ils ont obtenu des résultats de pointe dans de nombreux benchmarks sur ces tâches

- Les inconvénients d'un réseau de neurones à convolution

1. Nécessité d'avoir grandes quantités de données pour obtenir des résultats satisfaisants

Il n'est pas toujours évident de se procurer autant de données et cela influe aussi sur le temps de calcul des modèles

2. Difficulté de connaissance de la contribution de chaque poids dans l'erreur globale du réseau

3. Temps de calcul élevé

Les CNNs sont généralement plus coûteux en temps de calcul que les réseaux de neurones classiques en raison de la nature de leurs calculs de convolution

4. Sensibilité aux perturbations

Les CNNs sont sensibles aux perturbations des données d'entrée mais également à un temps d'apprentissage trop rapide/trop lent ainsi qu'au bruit. En effet, un taux d'apprentissage trop lent peut amener le réseau à être bloqué dans un minimum local lors de la descente du gradient tandis qu'un taux trop rapide entraîne des effets d'instabilités dans le réseau.

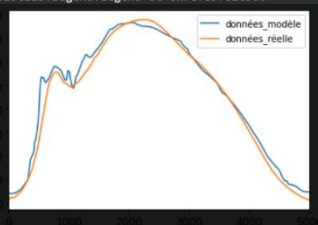
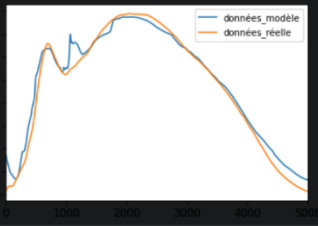
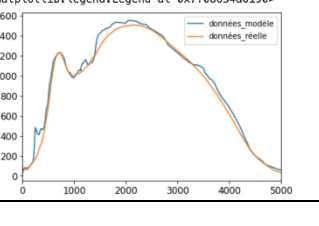
5. Problème de la mémoire disponible

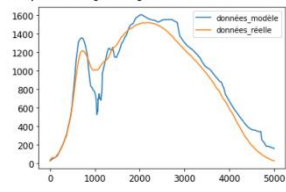
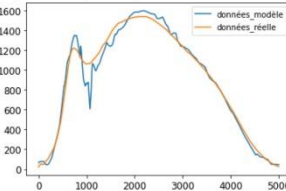
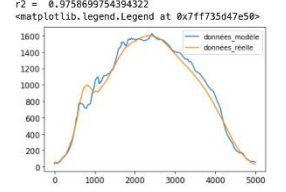
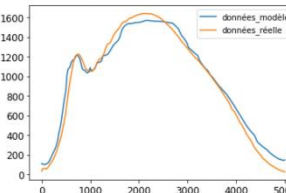
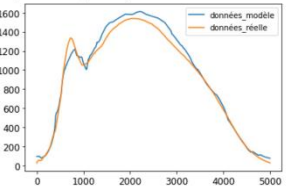
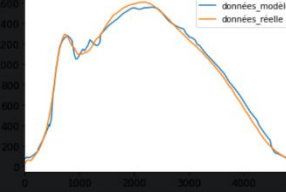
Les CNNs ont besoin d'une grande quantité de mémoire pour stocker les poids et les paramètres, ce qui peut limiter leur utilisation sur des appareils à faible puissance.

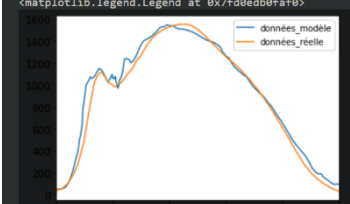
6. Danger du sur-apprentissage

Ce phénomène peut se produire quand il y a trop de neurones dans la couche cachée.

• Résultats des tests

Dataset choisi (fréquence_ sujet_passage)	R2 de test ?	RMSE de test	Résultats
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S1_P10	0.97890076508 97377	73.38895283663307	<pre>rmse = 73.38895283663307 r2 = 0.9789007650897377 <matplotlib.legend.Legend at 0x7efe9f82c3d0></pre> 
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S2_P10	0.97614216207 45739	78.62641777352403	<pre>rmse = 78.62641777352403 r2 = 0.9761421620745739 <matplotlib.legend.Legend at 0x7f9d03035490></pre> 
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S3_P10	0.98533552162 53435	59.49453571370388	<pre>157/157 [=====] - 0s 1ms/step rmse = 59.49453571370388 r2 = 0.9853355216253435 <matplotlib.legend.Legend at 0x7f08634a6190></pre> 

Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S4_P10	0.93696383815 52328	124.6118541975679	157/157 [=====] - 1s 4ms/step rmse = 124.6118541975679 r2 = 0.9369638381552328 <matplotlib.legend.Legend at 0x7fe74ed06640> 
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S5_P10	0.97927002134 96054	72.71978880580494	157/157 [=====] - 0s 1ms/step rmse = 72.71978880580494 r2 = 0.9792700213496054 <matplotlib.legend.Legend at 0x7ff735f91130> 
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S6_P10	0.97586997543 94322	79.35714752219127	157/157 [=====] - 0s 1ms/step rmse = 79.35714752219127 r2 = 0.9758699754394322 <matplotlib.legend.Legend at 0x7ff735d47e50> 
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S7_P10	0.97792197053 93288	79.94065510231083	157/157 [=====] - 0s 1ms/step rmse = 79.94065510231083 r2 = 0.9779219705393288 <matplotlib.legend.Legend at 0x7fe7432f9b50> 
Entraînement : datatest_F20_8P.xlsx Test : datatest_F20_S8_P10	0.98166439065 91485	68.68702390219202	157/157 [=====] - 0s 1ms/step rmse = 68.68702390219202 r2 = 0.9816643906591485 <matplotlib.legend.Legend at 0x7fa0372e1610> 
Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S9_P10	0.99320954833 59453	43.27958210144212	rmse = 43.27958210144212 r2 = 0.9932095483359453 <matplotlib.legend.Legend at 0x7f679dd307c0> 

<p>Entraînement : datatest_F20_8P.xlsx Test: datatest_F20_S10_P10</p>	<p>0.97501104799 84704</p>	
---	--------------------------------	---

5. Comparaison et analyse des résultats

- Quel est le meilleur modèle ?

Maintenant que nous avons obtenue toute ces données, nous allons les comparer afin de déterminer quelle est la meilleure solution.

On va baser notre analyse sur R^2 . C'est un indicateur utilisé en statistiques pour mesurer la qualité de la prédiction d'un modèle. Il mesure la proportion de la variance des données cibles qui est expliquée par les variables explicatives utilisées dans le modèle.

La valeur de R^2 varie entre 0 et 1, où une valeur de 1 indique que le modèle explique parfaitement les données cibles, tandis qu'une valeur de 0 indique que le modèle ne peut pas expliquer les données cibles. Une valeur proche de 1 est donc considérée comme un bon résultat.

On va donc récupérer les différentes valeurs obtenues lors des tests, en sortir une moyenne, la médiane et voire ce qu'on peut tirer comme conclusion.

On a donc le tableau récapitulatif suivant :

	Régression	SVM	CNN(Deep learning)
R^2 min	0.095	0.902	0.937
R^2 max	0.981	0.995	0.993
R^2 médian	0.750	0.982	0.978
R^2 moyen	0.688	0.962	0.976

Premièrement, on remarque que les résultats produit par le modèle de régression sont très inférieurs a ceux des 2 autres modèles avec un R^2 moyen et un R^2 médian plus faibles. Cela parait cohérent car il s'agit d'un modèle avec un fonctionnement moins élaboré que les deux autres donc moins adaptés pour notre ensemble de données complexe.

Nous allons donc nous concentrer sur les deux autres modèles. On remarque que les données sont très proches, le modèle SVM a un R^2 médian plus élevé mais un R^2 moyen plus faible que ceux du modèle CNN. Mais en mesurant l'écart entre ces deux valeurs, on se rend compte que le R^2 médian est très légèrement supérieur à celui du CNN (SVM : 0.982 > 0.978 : CNN) alors que la différence entre les R^2 moyens est plus important (SVM : 0.962 < 0.976 : CNN).

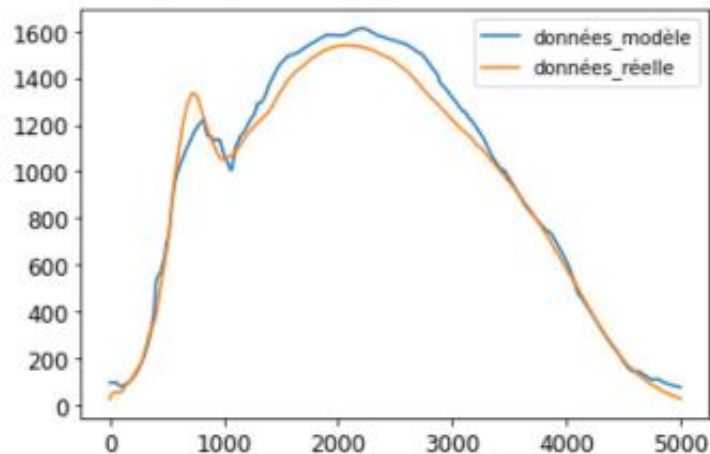
Il est donc assez difficile de conclure qu'un des deux modèles est bien meilleur que l'autre. Mais les résultats du CNN paraissent beaucoup plus constants que ceux du modèle SVM. En effet, la moyenne est plus élevée et les moins bons résultats obtenus avec ce modèle sont plus élevés que les moins bons résultats obtenus avec SVM.

Nous considérons donc que le modèle du CNN (Deep learning) est plus efficace.

- Les résultats sont-ils assez bon ?

Le modèle CNN semble être le meilleur modèle, mais est-ce qu'il est assez précis pour être vraiment utilisé ?

Globalement lorsqu'on superpose les courbes obtenues par le modèle avec les courbes réelles obtenues avec la plaque, on voit assez bien que les courbes sont très similaires.



On peut considérer que le modèle est assez efficace, dans la mesure où l'objectif du projet est de se rapprocher le plus de cette courbe.

Cependant on remarque pas mal de petites imperfections. Elles peuvent être dues à plusieurs facteurs :

- Modèle pas assez précis

Il est possible que le modèle que nous avons produit ne soit pas assez précis car il pourrait être mieux paramétré ou plus complexe (avec des architectures de réseaux de neurones plus avancé).

- Pas assez de données d'entraînement

Techniquement, plus la base d'entraînement est grande, plus le modèle sera précis car aura plus de données sur lesquelles se baser.

- Données imprécises

Ces petites imperfections peuvent aussi être dues à la façon dont ont été récoltés les données. Par exemple chaque capteur utilisé pour la collecte de ces données possède une incertitude, qui est présente sur chaque donnée d'entrée. L'incertitude est donc présente 6 fois, 1 pour chaque donnée d'entrée.

Il y a aussi plein d'autres incertitudes liées à la mise en place de la collecte des données. Par exemple lorsqu'un capteur est placé sur une personne, en fonction de sa morphologie ou du placement du capteur, les trois axes de chaque capteur peuvent être légèrement modifiés ce qui crée encore plus d'incertitude.

6. Axes d'amélioration possibles

- Récolter des nouvelles données

En récoltant plus de données et en entraînant les modèles sur ces données, il est possible d'obtenir des modèles plus fiables et plus performants. Cependant, plus on augmente la base de données d'apprentissage, plus le modèle prendra du temps pour être mis en place. On a vu précédemment qu'il était possible que les données n'aient pas été produites de manière optimale (capteur désaxé selon les différents passages, incertitudes ...). Récolter de nouvelles données en essayant de réduire au maximum toutes les sortes des variations et incertitudes lié à la récolte des données. Cela permettrait aussi de de confirmer/infirmier ou non cette hypothèse.

- Modèles plus complexes

Il est possible de créer des modèles plus complexes, par exemple dans le cadre du Deep Learning notamment en utilisant des architectures de réseaux de neurones plus avancé. Cependant, plus le modèle est complexe plus les calculs sont chronophages avec des machines standards.

- Créations de nouvelles données artificielles

Cette méthode consisterait à augmenter artificiellement le nombre de donnée d'entrée en augmentant la dimension de celle-ci. Par exemple, en déterminant les quelques données qui ont le plus d'impact on pourrait créer une 7^{ème} variable d'entrée qui serait une combinaison linéaire ou polynomiale de ces données. Il est aussi possible faire cela avec toutes les données d'entrée et même de créer plusieurs combinaisons. Ceci permettrait d'augmenter artificiellement les données qui ont plus d'impact sur le modèle, et potentiellement d'avoir des meilleurs résultats. Il est cependant important de noter que cela pourrait tout aussi avoir l'effet inverse, à savoir diminuer les performances du modèle. Tout dépend des combinaisons et coefficients qui seront choisis.

- Prise en compte de contrainte de régularisation

Les techniques de régularisations consistent à ajouter une contrainte supplémentaire au processus d'apprentissage d'un modèle afin de limiter la complexité du modèle et de réduire l'overfitting (le sur-apprentissage). Il existe plusieurs techniques de régularisation : régularisation des poids, L1, L2, dropout, Early stopping ... Il est par ailleurs important de noter que ces techniques de régularisation ne sont pas mutuellement exclusives, il est possible de les combiner afin d'obtenir de meilleurs résultats. En réduisant l'overfitting (sur-apprentissage), on éviterait aux modèles d'être trop adapté aux données d'entraînement et pas assez aux données de test.

Conclusion

En conclusion, notre projet visait à établir un modèle d'IA pour retrouver des données permettant l'analyse de la course d'un sportif, dans le but de diminuer les coûts et les ressources pour obtenir ces données, afin qu'elles soient plus facilement accessibles sur le plan matériel, économique et pratique pour les sportifs.

Après une analyse et une comparaison des différents modèles d'IA étudiés, il s'est avéré que le modèle CNN était celui qui présentait les meilleures performances pour résoudre ce type de problème. Ce modèle a montré de bonnes capacités à générer des prédictions précises et à s'adapter aux données.

Il faut cependant noter que notre modèle présente encore des imperfections qu'il est sûrement possible d'améliorer, il est donc important de continuer à les tester et à les améliorer en fonction des données et des besoins spécifiques.

En somme, cette étude nous a permis de solliciter différents modèles au service de l'analyse et du traitement des données. Nous avons pu utiliser aussi bien des modèles de Machine que de Deep Learning déjà étudiés en cours. Néanmoins, le travail de recherche poussé qui a été mené pour ajuster au mieux les modèles aux données couplé aux contraintes (notamment de temps) nous ont permis de mieux appréhender les exigences du monde de la recherche. Le travail effectué au sein d'une équipe internationale a été un challenge que nous avons su relever en nous y adaptant.