جامعة سيدي محمد بن عبد الله بفاس
ⵜⴰⵙⴷⴰⵡⵉⵜ ⵙⵉⴷⵉ ⵎⵓⵃⴰⵎⵎⴷ ⴱⵏ ⵄⴱⴷⵍⵍⴰⵀ ⴷⵖ ⴼⴰⵙ
Université sidi Mohamed ben abdellah de fés

المدرسة الوطنية لالمعلوم المتطبيقية
ⵜⵉⵍⵉⵏ ⵜⴰⵏⴰⵎⵓⵔⵜ ⵏ ⵜⵙⵙⵉⵏⵜⵉⵏ ⵜⵉⵎⵓⵔⵜⵉⵏ
Ecole Nationale des Sciences Appliquées de fés

Génie Systèmes Embarqués et Informatique Industrielle

RAPPORT DE STAGE D'APPLICATION

AU SEIN DU LABORATOIRE SYSTÈMES INTELLIGENTS, GÉORESSOURCES & ÉNERGIES RENOUVELABLES (SIGER)

Sujet : Entrainement d'un système multi-capteur pour la détéction du Covid-19 sur des images thermiques existantes

---

Auteurs:
Youssef EL KANTRI
Anas MANSOURI

Superviseurs:
Dr. Ali AHAITOUF
Dr. Malika ALAMI MARKTANI

SIGER
Laboratoire Systèmes Intelligents, Géoressources et Energies Renouvelables

AU : 2019 - 2020

Remerciements

Les premières personnes que nous tenons à remercier sont nos encadrant Mr ali ahaitouf et Mme Malika Alami Marktani , pour l'orientation, la confiance, la patience qui ont constitué un apport considérable sans lequel ce travail n'aurait pas pu être mené au bon port.
Nos remerciements s'étendent également à tous les professeurs pour leurs bonnes explications qui nous ont éclairé le chemin et leurs collaboration avec nous dans l'accomplissement de ce modeste travail. Enfin, on remercie tous ceux qui, de près ou de loin, ont contribué à la réalisation de ce travail.

Résumé :

Rapport ayant pour objectif premier de présenter les activités d'un stage au Laboratoire siger . Celui-ci comporte les outils et la démarche qu'on peut utiliser pour lutter contre le covid-19 en utilisant l'intelligence artificielle. Ce rapport est le fruit d'un stage de recherche de deux mois effectué au sein du Laboratoire SIGER à Fès, afin de valider ma 2eme années en cycle d'ingénieur en système embarqué . L'expérience, les relations et les interventions de mes encadrants de stage ont été très utiles pour recadrer une étude dont le champ est très vaste. J'ai voulu étudier un sujet d'actualité et encore peu exploré : Entraînement d'un système multi-capteur pour la détection du Covid-19 sur des images thermiques existantes. Le choix du sujet est, en effet, directement issu de l'actualité récente avec l'annonce faite par Organisation Mondiale de la Santé sur le danger de ce virus et comment il a paralysé l'économie mondiale . Le Laboratoire SIGER est spécialisée dans les recherches innovantes et nous a donc permis d'étudier cette tendance de près et d'essayer de trouver une solution pour lutter contre ce virus . notre sujet propose une solution basée sur l'intelligence artificiel et le machine learning pour prédire si une personne a le covid 19 ou non seulement on se basent sur la température ,la toux et le rythme respiratoire qui vont être collectés par une caméra thermique et des microphones .et pour l'entraînement du modèle on a utilisé une data-set créé artificiellement en attendant d'avoir la data set réel , pour les algorithme d'entraînement on a décidé d'utiliser le SVM et le randome forrest car ils donnent des bonnes résultats au niveau des problèmes de classification .

Mots-clés : Science, Apprentissage automatique, Covid-19

summary:

Report whose primary objective is to present the activities of an internship at the SIGER Laboratory . It includes the tools and the approach that can be used to fight covid-19 using artificial intelligence. This report is the result of a two-months research and hard work at the SIGER Laboratory in Fez, in order to validate my second year of engineering cycle in embedded systems. The experience, relationships and interventions of my internship supervisors have been very useful to reframe a study whose scope is very wide. I wanted to study a topical and still little explored subject: " Training of a multi-sensor system for Covid-19 detection on existing thermal images". The choice of the subject is, in fact, directly related to the recent news with the announcement made by the World Health Organization on the danger of this virus and how it has paralyzed the world economy. The SIGER Laboratory is specialized in innovative research and therefore allowed us to study this trend closely and try to find a solution to fight against this virus . Our subject proposes a solution based on artificial intelligence and machine learning to predict whether a person has the covid-19 or not based on temperature, cough and respiratory rhythm which will be collected by a thermal camera and microphones. For the training of the model we used an artificially created data set while waiting to have the real data set, for the training algorithm we decided to use the SVM and random forest because they give good results on classification problems.

Keywords : Science, Machine learning, Covid-19

Contents

Liste des figures

Introduction générale

En décembre 2019, le SARS-CoV-2 est apparu dans le centre de la Chine qui a pris rapidement des mesures drastiques de confinement et de désinfection pour près de 60 millions de personnes. Cependant, le virus, comparable à celui de la grippe espagnole, se répand partout dans le monde paralysant des pays entiers, suscitant la psychose et plongeant le monde dans une crise durable et inédite depuis la seconde guerre mondiale. Et pour bloquer la propagation du Covid-19, il faudrait que plus de 60% de la population ait été infectée et présente ainsi des anticorps ce qui est un peu loin à atteindre , ou on doit avoir un vaccin , et en attendant une de ces deux condition d'être vérifié l'équipe de chercheur du laboratoir siger ont décidé de travailler sur un portique intelligents qui a pour mission de détecter les gens qui ont le covid 19 on utilisant une caméra thermique et de l'intelligence artificiel . précisément notre groupe est le responsable de la partie traitement des données et de la partie prédiction en utilisant l'apprentissage automatique et l'intelligence artificiel .

1. présentation de SIGER

1.1. Fiche de présentation du Laboratoire de recherche

Intitulé du laboratoire	:	Laboratoire Systèmes Intelligents, Géoressources et Énergies Renouvelables
Type de structure	:	❑ Type 1 (18 - 27 membres permanents) x Type 2 (28 – 37 membres permanents) ❑ Type 3 (plus de 38 membres permanents)
Période d'accréditation	:	Du 01 janvier 2020 au 31 décembre 2025
Directeur	:	Pr. AHAITOUF Ali
Etablissement d'accueil	:	Faculté des Sciences et Techniques
Etablissements membres		FST - ENSA
Université	:	Université Sidi Mohamed Ben Abdellah, Fès.

Figure 1: Fiche de présentation du laboratoire SIGER

1.2. Equipes

les chercheurs se sont organisés, en interne en quatre équipe, et ont identifié les thématiques de recherche suivantes:

- Mathématique, Informatique et Intelligence Artificielle (M2IA)
- Géologie et Géophysique appliquées aux Géoressources et Géorisques (2GA2G)
- Systèmes Embarqués, Électronique et Télécommunications (S2ET)
- Energies Renouvelabes et Développement Durable (ER2D)

1.3. *Organigramme*

organigramme du laboratoire SIGER

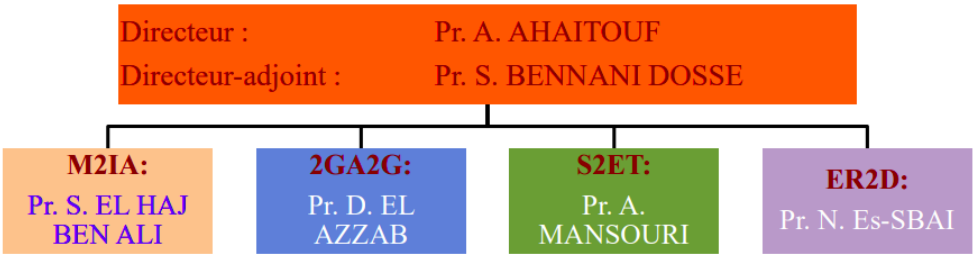

Figure 2: organigramme du laboratoire SIGER

1.4. *Effectif & Etablissement d'accueil*

- Le laboratoire a un effectif de :
  - 30 enseignants chercheurs permanents
  - 77 doctorants
  - 29 chercheurs associés
- Etablissement d'accueil: FST de Fès
- Etablissement membre: ENSA de Fès

1.5. *Thématiques de recherche*

L'approche adoptée par le laboratoire (SIGER) se veut résolument multidisciplinaire. Cette multidisciplinarité que le rechercheur au sein du SIGER veuillent développer et mettre en œuvre dans ce laboratoire a pour objectif d'encourager la recherche dans des thématiques qui s'inscrivent parmi les priorités du pays. En effet, la convergence des intérêts scientifiques des chercheurs et la complémentarité des savoirs seront investis, notamment autours des problématiques liées:

- Traitement de signaux, traitement d'image et traitement vidéo;
- Description matérielle, architectures parallèles, systèmes temps réel;
- Problème de la médiane ordonnée d'un nombre fini de fonctions rationnelles, la Théorie de localisation et ses applications pratiques, comme en gestion et valorisation des ressources naturelles et gestion des risques.
- Calcul Scientifique, Optimisation Polynomiale, Optimisation Multi-objectif et la Programmation Semi-définie positive, utiles aux modélisations visant la connaissance, la maitrise et la gestion des systèmes naturels et environnementaux;
- Services Web, Ingénierie dirigée par les Modèles, Adaptations et applications aux problèmes de gestion, voire d'anticipation de l'évolution des systèmes;

- Risques et sécurité des systèmes d'information;
- Physique des composants de l'échelle micro à l'échelle nanométrique, Physique des capteurs, capteurs intelligents et leurs applications;
- Conception des circuits intégrés analogiques et numériques multi-usages.
- Télécommunications, Antennes microonde, propagation et RFID et exploration des opportunité de leurs usages prospectifs, y compris dans le domaine de l'environnement;
- Géophysique de sub-surface (méthodes électriques, électromagnétisme, magnétisme, gravimétrie et sismiques) : application à la prospection des ressources naturelles, aux risques naturels et à l'environnement.
- Géophysique aéroportée (magnétomètres embarqués avec GPS, Spectromètre γ, capteurs électromagnétique émetteur - récepteur, caméra vidéo, altimètre radar, télédétection) : Traitements de données appliquées : 1/ à l'exploration minière et hydrique, 2/ aux structures géologiques, 3/ aux géorisques, 4/ à la pédologie, 5/ la désertification et 6/ à la cartographie thématique.
- Tectono-physique et dynamique terrestre appliquées à différentes échelles: identification des zones vulnérables (urbaines et ruraux) aux aléas naturels et de sites destinés à l'installation des capteurs de surveillance de ces aléas, mise en place de gisements miniers et exploration des énergies fossiles.
- Physique des roches (essais géotechniques, rhéologie): qualité des sols et identification et valorisation des géo matériaux .

2. Introduction à la modélisation de la classification

2.1. Modélisation

Selon le temps que nous avons passé dans un endroit particulier et le temps que nous avons passé à nous y rendre, nous avons probablement une bonne connaissance des temps de trajet dans notre région. Par exemple, nous nous sommes rendus au travail ou à l'école en utilisant une combinaison de métro, de bus, de trains, de bus urbains, de taxis, de covoiturage, de marche, de vélo, etc. Tous les humains modèlent naturellement le monde qui les entoure. Au fil du temps, nos observations sur les transports ont permis de constituer un ensemble de données et un modèle mental qui nous aident à prédire ce que sera la circulation à différents moments et en différents lieux. Nous utilisons probablement ce modèle mental pour nous aider à planifier nos journées, à prévoir les heures d'arrivée et bien d'autres tâches.

Essayer de rendre notre compréhension des relations entre différentes quantités plus précise en utilisant des données et des structures mathématiques et statistiques.

Ce processus s'appelle la modélisation.

Les modèles sont des simplifications de la réalité qui nous aident à mieux comprendre ce que nous observons.

Dans un contexte de science des données, les modèles se composent en général d'une variable dépendante (ou sortie) d'intérêt et d'une ou plusieurs variables indépendantes (ou entrées) censées avoir un effet sur la variable dépendante.

La régression linéaire est un outil de modélisation extrêmement courant et d'une importance capitale.

2.2. Inférence basée sur un modèle (Model-based inference)

Nous pouvons utiliser des modèles pour effectuer des inférences.
Grâce à un modèle, nous pouvons mieux comprendre les relations entre une variable indépendante et la variable dépendante ou entre plusieurs variables indépendantes.

Voici un exemple de cas où l'inférence à partir d'un modèle mental serait utile : *Déterminer les moments de la journée où nous travaillons le mieux ou où nous sommes fatigués.*

2.3. Prédiction

Nous pouvons utiliser un modèle pour faire des prédictions, ou pour estimer la valeur d'une variable dépendante (sortie) étant donné la valeur d'au moins une variable indépendante.
Les prédictions peuvent être utiles même si elles ne sont pas tout à fait exactes.
De bonnes prédictions sont extrêmement précieuses pour une grande variété d'objectifs.

Voici un exemple de cas où une prédiction issue d'un modèle mental pourrait être utile : *Prédire le temps qu'il faudra pour se rendre d'un point A à un point B.*

2.4. Différence entre la prédiction et l'inférence d'un modèle

- L'inférence consiste à juger de la relation, le cas échéant, entre les données et les résultats.
- La prédiction consiste à faire des suppositions sur des scénarios futurs en se basant sur des données et un modèle construit sur ces données.

Dans ce projet, nous parlerons de 2 modèles particulières d'apprentissage automatique appelés Machine à vecteurs support **(SVM)** et le forêt aléatoire **(random Forest classifier)**.

3. Machines à Vecteurs Support (SVM)

3.1. Présentation

SVM (Support Vector Machines) ou Machines à vecteurs supports sont des classifieurs faisant partie des techniques d'apprentissage supervisé. Introduits par Vapnik (1995) pour résoudre des problèmes de classification, ils ont connu depuis un grand succès car utilisés massivement dans divers domaines : reconnaissance de formes, OCR, bioinformatique... L'usage s'est également répandu vers la résolution des problèmes de régression, aussi bien gérés que les classifications. Cette technique fait appel à un jeu de données dit **d'apprentissage** dont les instances contiennent une valeur cible (Target) également appelée **étiquette de classe** ainsi que plusieurs attributs (attributes) représentant l'ensemble des variables observées, et ce pour produire un modèle permettant de

prédire les valeurs cibles d'un autre ensemble dit **de test**, en ne fournissant à ce modèle-là que les attributs des données de test. En d'autres termes, si on considère un jeu de données divisé en deux groupes : un groupe pour les exemples connus et un autre pour les exemples non connus, le but des SVM est d'apprendre une fonction qui traduit le comportement des exemples connus pour prédire les cibles des exemples inconnus.

3.2. Fonctionnement des SVM

Les SVM sont également appelés *classifieurs à vaste marge* car leur objectif est de trouver l'hyperplan séparateur optimal qui maximise la marge entre les classes dans un espace de grande dimension. La marge est la distance entre la frontière de séparation et les échantillons les plus proches, ces derniers sont appelés vecteurs supports. Une marge maximale permet d'obtenir une plus petite dimension de VC [1] , ce qui assure de bonnes performances en généralisation.

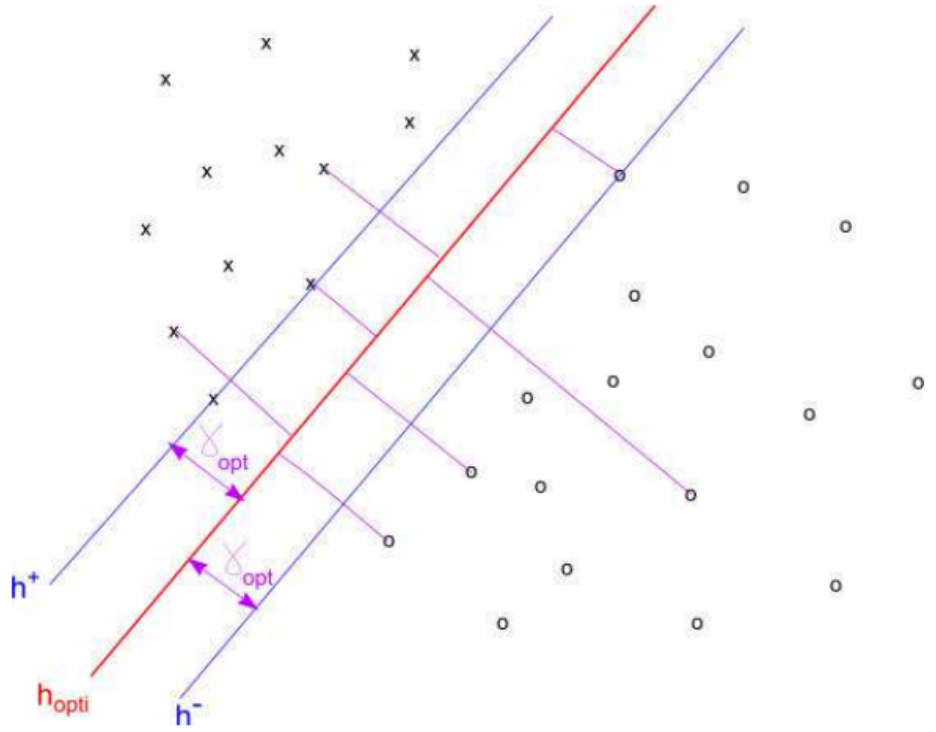

Figure 3: Hyperplan séparateur optimal

3.3. Temps d'exécutions

Les SVM sont considérés plus faciles à utiliser que les réseaux de neurones. Leur avantage principal est leur performance remarquable en généralisation avec des données de grandes dimensions. On leur reproche cependant d'être lents en convergence et gourmands en mémoire dès qu'il s'agit de traiter des jeux de données volumineux. En fait, entrainer un jeu de données d'une taille n, revient à résoudre un problème de programmation quadratique (QP) de taille n2, qui prend un temps de l'ordre O(n3). Cependant

plusieurs recherches allant dans ce sens ont permis d'améliorer les temps d'exécution vers **O(n2)**.

3.4. Avantages

Les avantages des machines à vecteurs de soutien sont les suivants :

- Efficace dans les espaces de grande dimension.
- Toujours efficaces dans les cas où le nombre de dimensions est supérieur au nombre d'échantillons.
- Utilise un sous-ensemble de points d'entraînement dans la fonction de décision(appelés vecteurs d'appui), donc également efficace en mémoire.
- Polyvalent : différentes fonctions du noyau (ou kernel) peuvent être spécifiées pour la fonction de décision. Des noyaux communs sont fournis, mais il est également possible de spécifier des noyaux personnalisés.

3.5. Inconvénients

Les inconvénients des machines à vecteurs de support sont notamment les suivants :

- Si le nombre de fonctions est beaucoup plus important que le nombre d'échantillons, il faut éviter de trop s'adapter dans le choix des fonctions du noyau et le terme de régularisation est crucial.
- Les MVS ne fournissent pas directement d'estimations de probabilité, celles-ci sont calculées à l'aide d'une validation croisée coûteuse au quintuple [2].

4. Forêt aléatoire (Random forest)

4.1. Présentation

Random forest est un algorithme d'apprentissage supervisé. La "forêt" qu'il construit est un ensemble d'arbres de décision, généralement formés par la méthode de "bagging".

L'idée générale de la méthode de "bagging" est qu'une combinaison de modèles d'apprentissage augmente le résultat global. En d'autres termes, la forêt aléatoire construit plusieurs arbres de décision et les fusionne pour obtenir une prédiction plus précise et plus stable.

L'un des grands avantages de la forêt aléatoire est qu'elle peut être utilisée pour les problèmes de classification et de régression, qui constituent la majorité des systèmes d'apprentissage machine actuels. Examinons la forêt aléatoire fonctionnementatoire dans la classification. Ci-dessous, vous pouvez voir à quoi ressemblerait une forêt aléatoire avec deux arbres :

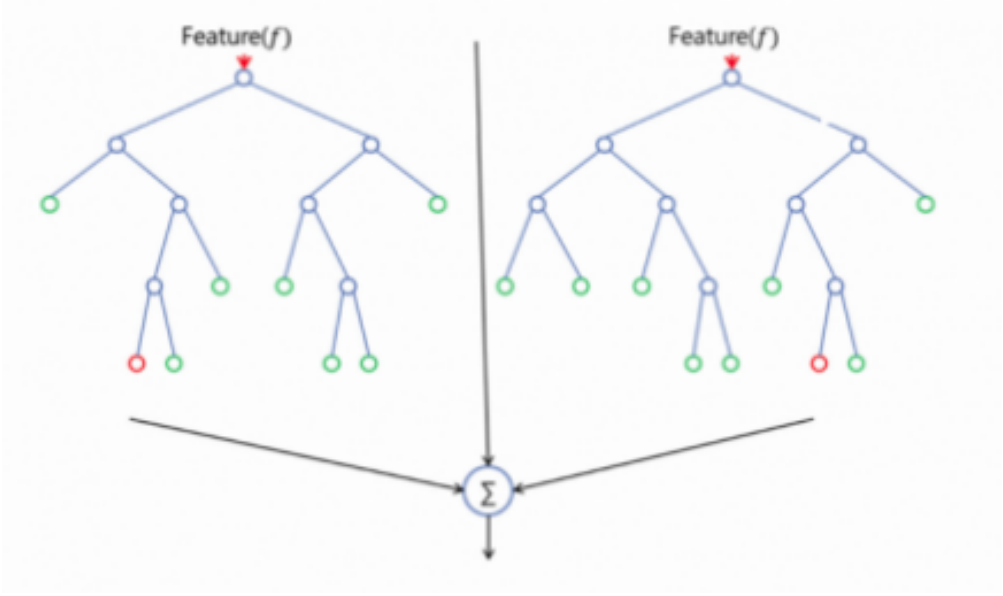

Figure 4: arbre de decisions et forêt aléatoire

4.2. Fonctionnement des random forest

Pour bien expliquer comment cet algorithme fonction andrew a expliquer le fonctionement en utilisant une situation reel : Andrew veut décider où aller pendant ses vacances d'un an, il demande donc des suggestions aux personnes qui le connaissent le mieux. Le premier ami qu'il cherche à rencontrer lui demande ce qu'il aime et n'aime pas dans ses voyages passés. En fonction des réponses, il donnera quelques conseils à Andrew.

Il s'agit d'une approche typique de l'algorithme de l'arbre de décision. L'ami d'Andrew a créé des règles pour guider sa décision sur ce qu'il devrait recommander, en utilisant les réponses d'Andrew.

Ensuite, Andrew commence à demander de plus en plus à ses amis de lui donner des conseils et ils lui posent à nouveau différentes questions dont ils peuvent tirer des recommandations. Enfin, Andrew choisit les endroits qui lui sont le plus recommandés, ce qui est l'approche typique de l'algorithme de la forêt aléatoire.

4.3. Avantages et inconvénients de l'algorithme de la forêt aléatoire

L'un des plus grands avantages du random forrest est sa polyvalence. Elle peut être utilisée à la fois pour des tâches de régression et de classification, et il est également facile de visualiser l'importance relative qu'elle attribue aux caractéristiques d'entrée. La forêt aléatoire est également un algorithme très pratique car les hyperparamètres par défaut qu'elle utilise produisent souvent un bon résultat de prédiction. La compréhension des hyperparamètres est assez simple, et il n'y en a pas beaucoup non plus. L'un des plus gros problèmes de l'apprentissage machine est le surajustement, mais la plupart du temps, cela ne se produit pas grâce au classificateur de forêt aléatoire. S'il y a suffisamment d'arbres dans la forêt, le classificateur ne surclassera pas le modèle. La principale limite de la forêt aléatoire est qu'un grand nombre d'arbres peut rendre l'algorithme trop lent et inefficace pour les prédictions en temps réel. En général, ces algorithmes sont rapides à entraîner,

mais assez lents à créer des prédictions une fois qu'ils sont entraînés. Une prédiction plus précise nécessite plus d'arbres, ce qui se traduit par un modèle plus lent. Dans la plupart des applications du monde réel, l'algorithme de la forêt aléatoire est assez rapide, mais il peut certainement y avoir des situations où la performance d'exécution est importante et où d'autres approches seraient préférables.

Et, bien sûr, la forêt aléatoire est un outil de modélisation prédictive et non un outil descriptif, ce qui signifie que si l'objectif est d avoir une description des relations entre les données, d'autres approches seraient préférables.

4.4. Cas d'utilisation du random forrest

L'algorithme des forêts aléatoires est utilisé dans de nombreux domaines différents, comme la banque, la bourse, la médecine et le commerce électronique. Dans la finance, par exemple, il est utilisé pour détecter les clients plus susceptibles de rembourser leur dette à temps ou d'utiliser plus fréquemment les services d'une banque. Dans ce domaine, il est également utilisé pour détecter les fraudeurs qui cherchent à escroquer la banque. Dans le domaine du commerce, l'algorithme peut être utilisé pour déterminer le comportement futur d'une action. Dans le domaine de la santé, il est utilisé pour identifier la bonne combinaison de composants en médecine et pour analyser les antécédents médicaux d'un patient afin d'identifier les maladies. Les forêts aléatoireS est utilisée dans le commerce électronique pour déterminer si un client va réellement aimer le produit ou non.

5. Mise en œuvre du projet

5.1. génération de la data-set artificielle:

En attendant la data-set réelle on a généré en une artificielle avec 25000 échantillons et 3 colonnes(température, toux, rythme_respiratoire),

- *température* : une valeur aléatoire de type réel dans l'interval [36.1,40]
- *toux* : une valeur aléatoire Booléenne (1 : pour la présence du toux)
- *rythme respiratoire* : une valeur aléatoire de type réel dans l'interval [18,30]

Pour la génération de la sortie(target) de chaque ligne on a suit une logique bien déterminée; lorsqu'on a deux symptomes qui sont présentes à la fois dans le corps d'une personnes on déclare que celle-ci a le Covid-19 les 3 symptomes sont :

- ***température > 37.5 °C***
- ***tous = 1***
- ***rythme respiratoire > 25 respiration/min***

```
In [2]: import random
```

```
In [3]: symptoms = []
        results = []
        for i in range(0, 25000):
            # generating some random data
            #generate 36.1 < temperature < 40
            temperature = random.uniform(36.1, 40)
            #generate toux = 1 or 0
            toux = random.choice([0, 1])
            #generate 36.1 < rythme respiratoire < 40
            rythme_respiratoire = random.uniform(18, 30)
            # initialise tested positive variable by 0
            tested_positive = 0
            # conditions :
            if temperature > 37.5:
                if toux or (rythme_respiratoire > 25):
                    tested_positive = 1
            elif toux and rythme_respiratoire > 25:
                tested_positive = 1
            #adding features (X_values) to dataset
            symptoms.append([temperature, toux, rythme_respiratoire])
            #adding target (Y_values) to dataset
            results.append(tested_positive)
```

verification des valeurs de la data-set :

```
In [4]: #check dataset
        #for i in range(0, len(symptoms)):
        for i in range(0, 5):
          print("{},{},{},{}".format(symptoms[i][0],
                                     symptoms[i][1],
                                     symptoms[i][2],
                                     results[i]))
```

```
38.34034716215434,0,24.632550486756223,0
37.87383112962509,0,19.40223163368522,0
39.32351309976807,1,25.083945995405372,1
37.312460634903424,1,23.507115281752355,0
37.60535027418449,1,18.879043711382568,1
```

la forme la plus simple pour stocker les donnee est la forme CSV puisque La fonction principale des fichiers CSV est de permettre la portabilité des données tabulaires d'un programme à l'autre (on va l'utiliser dans le programme de creation des modèles)

```
In [5]: #convert dataset to a csv file : "dataset.csv"
        import csv
```

```
        #add features list to csv file
        with open("dataset.csv", "w", newline="") as f:
            writer = csv.writer(f)
            writer.writerow(['temperature',
                             'toux',
                             'rythme respiratoire'])
            writer.writerows(symptoms)
```

```
In [6]: import pandas as pd
        #add target list to csv file
        df = pd.read_csv("dataset.csv")
        df["target"] = results
        df.to_csv("dataset.csv", index=False)
```

affichage de la data-set générée :

```
In [7]: print(pd.read_csv("dataset.csv"))
```

```
       temperature  toux  rythme respiratoire  target
0        38.340347     0            24.632550       0
1        37.873831     0            19.402232       0
2        39.323513     1            25.083946       1
3        37.312461     1            23.507115       0
4        37.605350     1            18.879044       1
5        39.467000     0            20.654400       0
6        38.762410     0            26.672363       1
7        38.138864     1            22.637751       1
8        36.133977     0            29.222709       0
9        37.473241     1            21.635431       0
10       39.094175     1            18.758283       1
11       39.309816     1            23.391463       1
12       38.021440     0            23.971838       0
13       39.770998     0            20.518829       0
14       38.359025     0            27.434461       1
15       38.500163     1            29.528288       1
16       37.424009     1            29.972855       1
17       38.265235     0            23.332563       0
18       36.413651     1            23.385560       0
19       38.667047     0            20.293605       0
20       36.957631     0            18.056544       0

...            ...   ...                  ...     ...
24986    39.460932     0            23.475217       0
24987    36.759540     1            21.050197       0
24988    36.311838     0            29.034118       0
```

```
24989    36.336869     1            21.606784       0
24990    36.865084     1            28.317075       1
24991    37.435315     1            24.784504       0
24992    39.758472     1            20.574149       1
24993    37.902057     0            29.782288       1
24994    38.976040     1            23.012267       1
24995    36.520314     0            23.334001       0
24996    36.551838     0            28.743914       0
24997    36.642622     0            24.861501       0
24998    38.017509     1            20.495668       1
24999    39.271873     0            23.505268       0

[25000 rows x 4 columns]
```

5.2. création des modèles de classification

Dans cette partie, nous allons essayer d'expliquer ligne par ligne notre code et son fonctionnement. Cette partie est divisée en 4 parties :

- Chargement et visualisation de l'ensemble de données de notre dataset.
- Prétraitement de l'ensemble des données
- entrainement des classificateurs (SVM & Random Forest)
- Teste des modèles

5.2.1. Chargement et visualisation de l'ensemble de données de notre dataset

Importation des Bibliothèques et paquets Python nécessaires

```
In [1]: from sklearn.model_selection import train_test_split
        from sklearn.ensemble import RandomForestClassifier #random forest
        from sklearn import svm #SVM
        from sklearn import metrics
        import pandas as pd
        import numpy as np
        import seaborn as sns

        %matplotlib inline
```

Exploration de notre base de données; extraire l'ensemble de données sous forme de dataframe pandas à partir d'un fichier csv (ensemble de données artificielles) :

```
In [2]: cols_list = ['temperature', 'toux', 'rythme respiratoire']
        symptoms = pd.read_csv ('dataset.csv', usecols=cols_list)
        results = pd.read_csv ('dataset.csv', usecols=['target'])
        print(symptoms.head())
```

```
   temperature  toux  rythme respiratoire
0    38.909697     0            25.622143
1    38.501598     0            27.080123
2    36.262807     1            21.746695
3    36.432818     0            19.021566
4    39.146504     1            23.887650
```

"results" = Il est la caractéristique que nous essayons de prévoir (Output). Dans ce cas, nous essayons de prédire si notre cible "Target" est infectée par Covid_19 ou non. C'est-à-dire que nous allons utiliser ici la colonne "Target".
"symptoms" = Les prédicteurs qui sont les colonnes restantes (température, toux, rythme respiratoire)

```
In [3]: symptoms.shape

Out[3]: (25000, 3)
```

Comme on peut le voir, nous avons 25000 lignes (Instances) et 3 colonnes (Features)

```
In [4]: symptoms.columns

Out[4]: Index(['temperature', 'toux', 'rythme respiratoire'], dtype='object')
```

Ci-dessus se trouve le nom de chaque symptôme de notre dataframe.
Visualiser nos données

```
In [5]: data=symptoms.copy()
        data[['target']] = results[['target']]

In [6]: symptoms.columns

Out[6]: Index(['temperature', 'toux', 'rythme respiratoire'], dtype='object')

In [7]: sns.pairplot(data, hue = 'target',
                     vars = ['temperature',
                             'toux',
                             'rythme respiratoire'])

Out[7]: <seaborn.axisgrid.PairGrid at 0x7f7fd62c4550>
```

Les graphs ci-dessus montrent la relation entre nos caractéristiques (features).
N.B :
1,0 (Orange) = Personne testée positif (Covid_19)
0,0 (bleu) = Personne estée négatif (not Covid_19)
Combien de personnes ont été testées positives et négatives dans notre ensemble de données

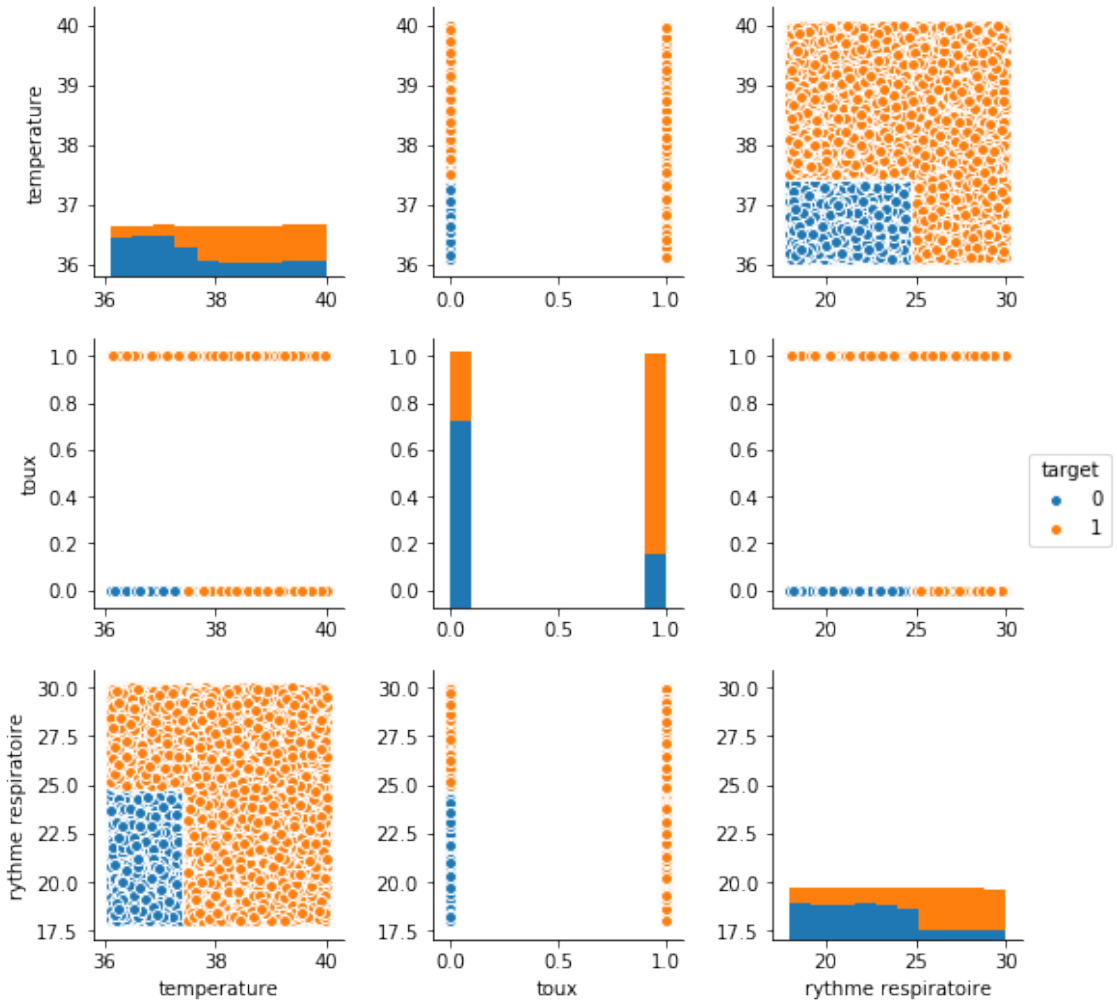

Figure 5: relation entre les caractéristiques deux par deux

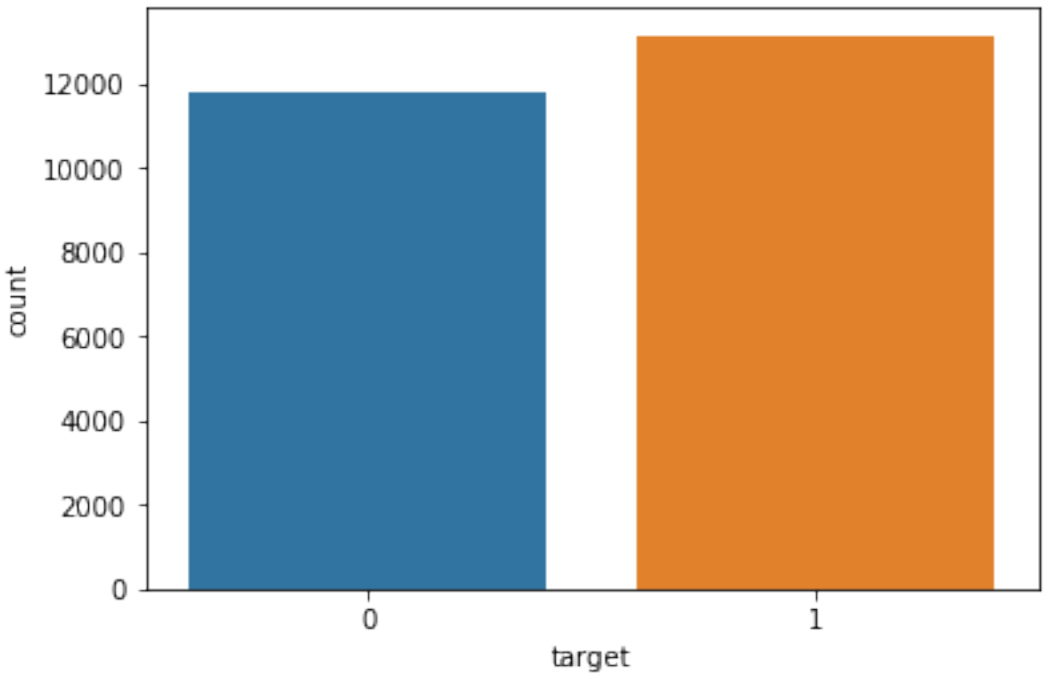

Figure 6: nombre des échantillons dans notre dataset

```
In [8]: sns.countplot(data['target'], label = "Count")
```

```
Out[8]: <matplotlib.axes._subplots.AxesSubplot at 0x7f7fce17d080>
```

5.2.2. Prétraitement de l'ensemble des données

Répartition des données, 80% pour la formation et 20% pour le test

- Données d'entrainement = C'est le sous-ensemble de nos données utilisées pour former(entrainer) notre modèle.

- Données de test = Sous-ensemble de données que le modèle n'a jamais vu auparavant. Elles sont utilisées pour tester la performance de notre modèle.

```
In [9]: X_train, X_test, y_train, y_test = train_test_split(
            symptoms.values.tolist(),
            results.values.tolist(),
            test_size=0.2,
            random_state=20)
```

Amélioration notre modèle nous allons essayer est en Normalisant nos données
La normalisation des données est un processus de mise à l'échelle des features qui amène toutes les valeurs dans un intervalle [0,1].
X' = (X-X_min) / (X_max - X_min)-----> X_range

normalisation des données d'entrainement :

```
In [10]: X_train_range = (pd.DataFrame(X_train).max()
                          - pd.DataFrame(X_train).min())
         X_train_range.head()
         X_train_scaled = (pd.DataFrame(X_train)
                           - pd.DataFrame(X_train).min())/(X_train_range)
         X_train_scaled.head()
```

Out[10]:

	0	1	2
0	0.252834	0.0	0.452785
1	0.365551	1.0	0.719840
2	0.975454	1.0	0.491346
3	0.133140	1.0	0.847429
4	0.368953	1.0	0.570953

5.2.3. Entrainement des classificateurs SVM et RF

```
In [11]: #SVM
         SVM_clf = svm.SVC(kernel="rbf",
                           gamma='auto') # rbf Kernel
         #Random Forrest
         RF_clf=RandomForestClassifier(n_estimators=100)
```

entrainer le modèle SVM et RF avec l'ensemble de données d'entrainement

```
In [12]: #Train the models using the training sets
         y_train = np.ravel(y_train)
         SVM_clf.fit(X_train_scaled, y_train)
         RF_clf.fit(X_train_scaled, y_train)
```

```
Out[12]: RandomForestClassifier(bootstrap=True, class_weight=None,
                     criterion='gini',
                     max_depth=None, max_features='auto', max_leaf_nodes=None,
                     min_impurity_decrease=0.0, min_impurity_split=None,
                     min_samples_leaf=1, min_samples_split=2,
                     min_weight_fraction_leaf=0.0, n_estimators=100, n_jobs=1,
                     oob_score=False, random_state=None, verbose=0,
                     warm_start=False)
```

5.2.4. Teste des modèles

normalisation des données de test :.

```
In [13]: X_test_range = (pd.DataFrame(X_test) -
                         pd.DataFrame(X_test).min()).max()
         X_test_range.head()
         X_test_scaled = (pd.DataFrame(X_test) -
                          pd.DataFrame(X_test).min())/(X_test_range)
         X_test_scaled.head()
```

```
Out[13]:           0    1         2
         0  0.524939  1.0  0.265805
         1  0.935205  0.0  0.045115
         2  0.194249  1.0  0.997728
         3  0.756674  1.0  0.817467
         4  0.619889  1.0  0.799443
```

Utiliser le modèle entrainé pour faire une prédiction sur les données de test.

```
In [14]: #Predict the response for test dataset\
         y_pred_SVM = SVM_clf.predict(X_test_scaled)
         y_pred_RF = RF_clf.predict(X_test_scaled)

In [15]: X_test_scaled.head()

Out[15]:           0    1         2
         0  0.524939  1.0  0.265805
         1  0.935205  0.0  0.045115
         2  0.194249  1.0  0.997728
         3  0.756674  1.0  0.817467
         4  0.619889  1.0  0.799443
```

nous vérifierons l'exactitude de nos classifiacteur en la comparant à la sortie dont nous disposons déjà (y_test) du SVM et du RF. Nous allons utiliser la matrice de confusion pour cette comparaison

```
In [16]: #accuracy value :
         print("SVM accuracy : ",metrics.accuracy_score(y_test, y_pred_SVM))
         print("RF accuracy : ",metrics.accuracy_score(y_test, y_pred_RF))

SVM accuracy :  0.9514
RF accuracy :  0.9996
```

La matrice de confusion. Une matrice de confusion pour une tâche de classification binaire :

	PredictedNegative	PredictedPositive
Négatif réel	vrai négatif (TN)	faux positif (FP)
Positif réel	faux négatif (FN)	vrai positif (TP)

Dans un classificateur binaire, la classe "vrai" est généralement étiquetée avec 1 et la classe "faux" avec 0.

- **Vrai Positif** :une observation de classe positive (1) est correctement classée comme positive par le modèle.

- **Faux positif**: Une observation declasse négative (0) est incorrectement classée comme positive.
- **Vrai négatif** : une observation de classe négative est correctement classée comme négative.
- **Faux négatif** : uneobservation de classe positive est incorrectement classée comme négative.

Les colonnes de la matrice de confusion s'additionnent aux prédictions par classe. Les lignes de la matrice s'additionnent aux valeurs réelles à l'intérieur de chaque classe. Vous pouvez rencontrer des matrices de confusion où le réel est dans les colonnes et le prédit est dans les lignes : la signification est la même mais le tableau sera réorienté.

Note: Se souvenir de ce que représentent les cellules de la matrice de confusion peut être un peu délicat. Le premier mot (Vrai ou Faux) indique si le modèle était correct ou non. Le deuxième mot (Positif ou Négatif) indique la supposition du modèle (et non l'étiquette réelle)

```
In [18]: from sklearn.metrics import classification_report, confusion_matrix
         from pandas import DataFrame
         import matplotlib.pyplot as plt
         from seaborn import heatmap
```

Créons une matrice de confusion pour notre classificateur sur l'ensemble des données de test.

```
In [18]: y_test = np.array(y_test) # casting y_test type to numpy array
```

```
In [19]: #confusion matrix :
         cm_SVM = np.array(confusion_matrix(y_test,
                                            y_pred_SVM,
                                            labels =[1,0]))
         confusion_SVM = pd.DataFrame(cm_SVM,
                                      index=['testée positive',
                                             'testée négative'],
                                      columns=['prédit positive',
                                               'prédit négative'])
         print("la matrice du confusion du SVM est :")
         confusion_SVM
```

```
la matrice du confusion du SVM est :
```

```
Out[19]:                 prédit positive  prédit négative
         testée positive             2524              116
         testée négative              127             2233
```

```
In [20]: cm_RF = np.array(confusion_matrix(y_test,
                                           y_pred_RF,
                                           labels =[1,0]))
         confusion_RF = pd.DataFrame(cm_SVM,
                                     index=['testée positive',
                                            'testée négative'],
                                     columns=['prédit positive',
                                              'prédit négative'])
         print("la matrice du confusion du Random Forest est :")
         print(confusion_RF)
```

```
la matrice du confusion du Random Forest est :
                 prédit positive  prédit négative
testée positive             2524              116
testée négative              127             2233
```

générer un rapport sur les perfomances de notre modèle.

```
In [21]: print("les mesures pour SVM :")
         print(classification_report(y_test,y_pred_SVM))
         print("les mesures pour Random Forest :")
         print(classification_report(y_test,y_pred_RF))
```

```
les mesures pour SVM :
             precision    recall  f1-score   support

          0       0.95      0.95      0.95      2360
          1       0.95      0.96      0.95      2640

avg / total       0.95      0.95      0.95      5000

les mesures pour Random Forest :
             precision    recall  f1-score   support

          0       1.00      1.00      1.00      2360
          1       1.00      1.00      1.00      2640

avg / total       1.00      1.00      1.00      5000
```

PRECISION : La capacité du classificateur à éviter de classifier une classe comme membre d'une autre classe. *Precision = True Positives / (True Positives + False Positives)*

Une note de précision de 1 indique que le classificateur n'a jamais classé par erreur la classe actuelle comme une autre classe. Une note de précision de 0 signifie que le classificateur a mal classé tous les instances de la classe actuelle

RECALL/SENSITIVITY : La capacité du classificateur à identifier correctement la classe actuelle.

Recall = True Positives / (True Positives + False Negatives)

Un rappel (recall) de 1 indique que le classificateur a correctement prédit toutes les observations de la classe. Un rappel de 0 signifie que le classificateur a prédit incorrectement toutes les observations de la classe actuelle.

F1-SCORE : La moyenne harmonique est utilisée ici plutôt que la moyenne arithmétique plus conventionnelle, car la moyenne harmonique est plus appropriée pour le calcul de la moyenne des taux. *F1-Score = 2 * (Precision * Recall) / (Precision + Recall)*

la meilleure valeur du f1-score est 1 et la pire valeur est 0, comme les scores de précision et de recall. Il s'agit d'une mesure utile pour prendre en compte les deux mesures à la fois.

SUPPORT : La somme marginale des lignes dans la matrice de confusion, ou en d'autres termes le nombre total d'observations appartenant à une classe indépendamment de la prédiction.

enregistrer les modèles sous forme de pickle object :.

```
In [22]: # save the models to disk
         import pickle
         filename1 = 'finalized_model_SVM.sav'
         filename2 = 'finalized_model_RF.sav'

         pickle.dump(SVM_clf, open(filename1, 'wb'))
         pickle.dump(SVM_clf, open(filename2, 'wb'))
```

charger les modèles sous forme de pickle object :.

```
In [23]: # load the models from disk
         loaded_model_SVM = pickle.load(open(filename1, 'rb'))
         loaded_model_RF = pickle.load(open(filename2, 'rb'))
```

test d'une seule ligne après le chargemenet de fichier pickle de notre modèle:.

```
In [24]: temperature= 40;toux=0; rythme_respiratoire=30
         test = [temperature,toux,rythme_respiratoire]
         #print(pd.DataFrame(test))
         #print(pd.DataFrame(X_test).min())
         #normalisation de la ligne de test
         test_scaled = (pd.DataFrame(test)[0] -
                        pd.DataFrame(X_test).min())/(X_test_range)
         y_pred_SVM = loaded_model_SVM.predict([test_scaled,])
         if y_pred_SVM == 1:
             print("La personne a COVID-19 selon SVM Classifier")
         else :
             print("La personne n'a pas de COVID-19 selon SVM classifier")
         y_pred_RF = loaded_model_RF.predict([test_scaled,])
```

```
        if y_pred_RF == 1:
            print("La personne a COVID-19 selon RF Classifier")
        else :
            print("La personne n'a pas de COVID-19 selon RF classifier")
```

```
La personne a COVID-19 selon SVM Classifier
La personne a COVID-19 selon RF Classifier
```

Conclusion générale

Pour conclure, nous avons effectué le stage d'application de 2eme annees du cycle d'ingénieur en tant que stagiaires en intelligence artificiel au sein du Laboratoir SIGER à fès .

Lors de ce stage de 6 mois, on a pu mettre en pratique les connaissances théoriques acquises durant notre formation sur le machine learning et nous sommes confrontés aux difficultés du monde du travail et du management d'équipes dans le secteur de recherche.

Ce stage a été très enrichissant pour nous, car il nous a permis de découvrir le domaine du recherche scientifique au niveau des système embarqués et de l'intelligence artificiel , ses acteurs, contraintes. Il nous a permis aussi de participer concrètement à ses enjeux à travers nos missions en programmation et modélisation . Ce stage nous a aussi permis de comprendre que les solutions innovantes au futur vont être basé principalement sur la combinaison des systemes embarque et l'intelligence artificiel.

Cette expérience de stage fut très constructive et nous a permis de répondre aux questionnements que nous avions en ce qui concerne les algorithmes et les méthodes utilisés par les entreprises pour créer des modèles capable de décider et de choisir face à des situations réelle sans avoir l'aide des humains .

Fort de cette expérience et en réponse à ses enjeux, nous aimerons beaucoup par la suite essayer de nous orienter vers un prochain stage dans le secteur des objets connectés (internet of things) .

References

[1] Vapnik-Chervonenkis, « Théorie statistique de l'apprentissage »,1990, http://images.math.cnrs.fr/pdf2006/Catoni.pdf

[2] scikit-learn documentation, « Scores and probabilities », https://scikit-learn.org/stable/modules/svm.html#scores-probabilities

Lien vers les codes

https://github.com/ysfelkantri/Entrainement_d-un_systeme_multi-capteurs_pour_la_detection_du_COVID-19