REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche

Scientifique Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات

Ecole Nationale Polytechnique

Département d'Electronique

Mémoire de projet de fin d'études

pour l'obtention du diplôme d'ingénieur d'état en Electronique

Conception d'un système sur puce dédié à la prévention du diabète

Amel TIBHIRT

Karima CHENIKHA

Sous la direction de M .Rabah SADOUN Pr.

Présenté et soutenu publiquement le (01/07/2019)

Composition du Jury :

Président	M. Hicham BOUSBIA-SALAH	Dr.	ENP
Promoteur	M. Rabah SADOUN	Prof.	ENP
Examinateur	M. Mourad ADNANE	Dr.	ENP
Co-promotrice	Mme. Nour El Houda BENALIA	Dr.	ENP
Co-promotrice	Mme. Widad Kartous	Dr.	ENP

REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Ecole Nationale Polytechnique

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

Département d'Electronique

Mémoire de projet de fin d'études

pour l'obtention du diplôme d'ingénieur d'état en Electronique

Conception d'un système sur puce dédié à la prévention du diabète

Amel TIBHIRT

Karima CHENIKHA

Sous la direction de M .Rabah SADOUN Pr.

Présenté et soutenu publiquement le (01/07/2019)

Composition du Jury :

Président	M. Hicham BOUSBIA-SALAH	Dr.	ENP
Promoteur	M. Rabah SADOUN	Prof.	ENP
Examinateur	M. Mourad ADNANE	Dr.	ENP
Co-promotrice	Mme. Nour El Houda BENALIA	Dr.	ENP
Co-promotrice	Mme. Widad Kartous	Dr.	ENP

Dédicace

Du profond de mon cœur, je dédie ce travail à ceux qui me sont chers,

A mes chers parents Hassina et Mohammed,

Aucune dédicace ne saurait exprimer mon respect, mon amour éternel et ma considération pour les sacrifices que vous avez consenti pour mon instruction et mon bien être.

Que ce modeste travail soit l'exaucement de vos vœux tant formulés, le fruit de vos innombrables sacrifices. Puisse Dieu, le Très Haut, vous accorder santé, bonheur et longue vie.

A mes chères sœurs Soumeya, Lynda, Souad, Wissem et mon cher petit Frère Ibrahim,

Qui m'ont toujours poussé et motivé dans mes études, et qui m'ont assisté dans les moments difficiles. En témoignage de mon affection fraternelle, de ma profonde tendresse et reconnaissance, je vous souhaite une vie pleine de bonheur et de succès et que Dieu, le tout puissant, vous protège et vous garde.

A mes grands-parents,

A la mémoire de mes grands-parents,

A mes chers tantes et oncles,

A mes chers cousins et cousine,

A mon binôme Karima, en souvenir de notre sincère et profonde amitié et des moments agréables que nous avons passés ensemble.

Amel

Je dédie ce travail à

La mémoire de mes grands-parents.

Mes Parents, aucun mot ne saurait exprimer, mon respect, mon amour et ma considération pour les sacrifices consentis pour mon instruction et mon bien être

Mes sœurs : Sarah, Nesrine et Nassima, pour leur soutien, leurs encouragements et leurs sacrifices

Ma chère Tante Zoubida et son époux

Mon neveu Anes

Mes cousins : Amel & Nabil

Mon Beau-frère Messaoud

Mon binôme, Amel, pour son amitié, sa patience et son soutien

Mes amies : Hiba, Mira, Randa, Roumaissa, Samia et toute personne, m'ayant soutenu durant mes moments difficiles

Karima

Remerciements

Nous tenons tout d'abord à remercier Dieu le tout puissant de nous avoir donné le courage durant les périodes les plus difficiles et qui nous a donné la force et la patience d'accomplir ce modeste travail.

Nous remercions vivement, notre encadreur, Monsieur ***RABAH SADOUN***, pour son aide et son soutien durant la réalisation de ce travail

Nos vifs remerciements, vont également aux membres du jury, M. ***HICHAM BOUSBIA*** et
M***. MOURAD ADNANE*** pour l'intérêt qu'ils ont porté à notre travail en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.

Nous remercions également, Madame **Nour El Houda BENALIA** et Madame **Wiadad KARTOUS**, pour leur aide.

Enfin, nous tenons également à remercier toute personne ayant participé de près ou de loin à la réalisation de ce travail.

ملخص

يعتبر مرض السكري اليوم، مرض العصر. عدد الاشخاص المصابين بهذا المرض يتزايد بطريقة أسية وقد بلغ 350 مليون مصاب.

يتلخص هذا المشروع في تصميم نظام على رقاقة مخصص للوقاية من مرض السكري، النموذج المطبق على الأجهزة هو شجرة القرار، تم الحصول عليها باستخدام التعلم الآلي، وهو أحد أكثر أشكال الذكاء الاصطناعي انتشارا.

تتكون قاعدة البيانات المستخدمة في هذه الدراسة من صفات تنبؤية طبية

الكلمات الدالة: مرض السكري، الذكاء الاصطناعي، التعلم الآلي، شجرة القرار،

Abstract

Diabetes is nowadays considered as the disease of the century. The number of persons affected by this disease increases exponentially and affects over 350 million people worldwide.

The project includes the conception of a system on chip dedicated to diabetes prevention, the model implemented on hardware is a decision tree, obtained by using Machine Learning, one of the most widespread forms of artificial intelligence.

The database used in this study consists of medical predictive attributes and was used with the Scikit Learn library to obtain the model.

Key words: Diabetes, Artificial intelligence, Machine Learning, Decision Tree, System on Chip, Scikit Learn.

Résumé

Le diabète est considéré aujourd'hui comme la maladie du siècle. Le nombre de personnes touchées par cette maladie croît de manière exponentielle et touche, au jour d'aujourd'hui, plus de 350 millions de personnes à travers le monde.

Ce projet consiste en la conception d'un système sur puce dédié à la prévention du diabète. Le modèle implémenté sur hardware est un arbre de décision obtenu en utilisant l'apprentissage automatique, une des formes de l'intelligence artificielle les plus répandues.

La base de données utilisée dans cette étude est constituée d'attributs prédictifs médicaux et a été utilisé avec la bibliothèque Scikit Learn pour l'obtention du modèle.

Mots clés : Diabète, Intelligence Artificielle, Apprentissage automatique, Arbre de Décision, Système sur puce, Scikit Learn.

TABLE DES MATIERES

LISTE DES TABLEAUX

LISTE DES FIGURES

LISTE DES ABREVIATIONS

IA	Intelligence Artificielle
ML	Machine learning
DT	Decision Tree
SVM	Support Vector Machine
RF	Random Forest
RMSE	Root Mean Square Error
AUC	Area Under Curve
KNN	K-Nearest Neighbour
ID3	Iterative Dichotomizer3
CART	Classification And Regression Tree
HLS	High Level Synthesis
DP	Design Productivity
VHDL	VHSIC (Very High Speed Integrated Circuit) Hardware Description Language
LUT	Look Up Table
RTL	Register Transfer Language
FPGA	Field-Programmable Gate Array
RAM	Random Access Memory
BSV	BlueSpec System Verilog
CWB	Cyber Work Bench
BDL	Behavioral Description Language
FF	Flip Flop
SoC	System on Chip
AMBA	Advanced Microcontroller Bus Architecture
DMA	Direct Memory Access
AXI	Advanced eXtensible Interface
GPIO	General Purpose Input/Output
I2C	Inter-Integrated Circuit
UART	Universal Asynchronous Receiver Transmitter
CAN	Convertisseur Analogique/Numérique

Abréviations

SPI	Serial Peripheral Interface
PL	Programmable Logic
PS	Processing System
DSP	Digital Signal Processor
AXI	Advanced eXtensible Interface
SDK	Software Development Kit
INS	Institut National de Santé Publique
OMS	Organisation Mondiale de la Santé
MSPRH	Ministère de la Santé de la Population et de la Réforme Hospitalière
GPU	Graphic Prossesor Unit
ASIC	Application Specific Integrated Circuit

Introduction Générale

Le diabète sucré - une maladie chronique qui présente des complications sévères, touche des millions de personnes dans le monde. On le retrouve dans presque toutes les populations et il apparaît comme un problème qui prend de l'ampleur dans les pays en développement. Le coût de la maladie, en termes de souffrances, de soins de santé et de pertes humaines, est élevé. Une meilleure connaissance des causes et des mécanismes des principaux types de diabète sucrés constitue désormais la base des activités de prévention [1].

On peut catégoriser cette dernière en deux classes. Celle traditionnelle se basant fondamentalement sur une approche basée sur des recommandations (alimentation saine, poids correct, activité physique régulière) couplée à des campagnes de sensibilisation et de dépistage et l'autre, basée sur l'exploitation des nouvelles technologies de l'information et de la communication aussi bien sur le volet recueil de données que celui du stockage et surtout du traitement. Dans ce dernier cas, l'intelligence artificielle se présente comme une solution émergente dans l'analyse et l'exploitation des données. Son application au domaine du traitement du diabète s'avère être pertinente.

L'IA se définit, plus généralement, comme la capacité d'une machine à agir par elle-même ou sous le contrôle de l'homme pour reproduire des actions ou des fonctions qui sont habituellement dévolues aux humains. Aujourd'hui, nous la retrouvons dans plusieurs domaines ; on cite les équipements informatiques, les objets connectés, les applications réseaux, le transport et autre. Le domaine médical s'avère un champ d'application majeur. L'application de l'IA à la médecine offre une perspective d'application plus qu'intéressante de ces nouvelles technologies, qu'il s'agisse de renforcer le lien entre patients et médecins, de poser des diagnostics plus rapides et plus précis, ou encore d'optimiser la création de nouveaux traitements. La finalité se recentre sur la santé de l'humain (confort de vie, prévention, longévité, ...) en tant qu'individu mais aussi, et par extrapolation, sur la réduction des coûts économiques pour la société quant à la prise en charge de patients.

L'approche de la construction de notre système de prévention de diabète s'articule sur deux phases à savoir l'obtention du modèle et son implémentation sur une cible technologique. La première phase passe par l'exploitation de base de données conjuguée au choix, et d'un algorithme approprié que de la plateforme devant le produire. En deuxième phase, le choix de la cible technologique reste dépendant de la performance ciblée en fonction du temps de réaction préconisé (donc d'alerte dans le cas d'une prévention). La taille du vecteur de données à traiter peut conditionner ce dit temps de réaction. Une cible technologique flexible associée

à une méthodologie de développement adaptée pourra être un choix judicieux pour le développement de toute solution IA aussi bien sur le cloud, le edge que sur tout nœud d'extrémité (c'est à dire le patient via par exemple son smartphone ou sur tout capteur embarqué).

Pour décrire notre projet et la solution que nous avons implémentée, nous avons organisé notre mémoire en quatre chapitres.

Le premier chapitre, décrit le diabète, les méthodes de prévention du diabète et l'importance de diagnostiquer cette maladie d'une manière précoce.

Le deuxième chapitre, concerne notre méthodologie suivie pour l'obtention du modèle de prévention le plus précis. Nous nous sommes imposés deux degrés de libertés à savoir l'algorithme utilisé et la meilleure plateforme ML qui le porte.

Dans le troisième chapitre, nous présentons une méthode d'obtention du modèle décrivant le comportement du diabète en fonction des variables prédictives médicaux en utilisant l'algorithme ML déduit du chapitre précédent.

Et dans le quatrième chapitre, nous présenterons notre flot de conception pour l'implémentation du modèle obtenu sur une cible technologique; un SoC en l'occurrence dans notre cas. Ce SoC pourra aussi bien porter notre modèle soit sous une forme d'un accélérateur matériel, soit une implémentation exclusivement software.

On finira par une conclusion qui analysera les résultats obtenus et en donnera les perspectives que nous envisageons.

Chapitre I

Le diabète

1.1 Introduction

Le diabète est une maladie chronique connue aussi sous le nom de "tueur silencieux". Cette maladie peut être causée par l'incapacité de notre organisme à produire de l'insuline. Les effets du diabète sucré comprennent des dommages à long terme, un dysfonctionnement et une défaillance de divers organes [1]. Ce qui a causé une augmentation significative de la mortalité chez les patients.

Avec l'amélioration du niveau de vie, le diabète est de plus en plus courant dans la vie quotidienne des gens. Par conséquent, comment diagnostiquer et analyser le diabète rapidement et avec précision est un sujet qui mérite d'être étudié.

Une alimentation saine, un poids correct, activité physique régulière, des campagnes de sensibilisation, de dépistage et d'informations, construisent les méthodes traditionnelles de prévention du diabète. En outre l'avancement technologique a mis en question cette problématique, d'où l'utilisation de machine Learning.

Le MLpeut aider les gens à porter un jugement préliminaire sur le diabète en fonction des données de leurs examens physiques quotidiens. Il peut également servir de référence aux médecins.

1.2 Le diabète

1.2.1 Qu'est-ce que le diabète ?

Le diabète est une maladie chronique, qui peut être traité et contrôlé. Il est causé par un manque ou un défaut d'utilisation d'une hormone appelée insuline.

L'insuline diminue le taux de glucose dans le sang et favorise son utilisation par les tissus de l'organisme. Chez une personne non diabétique, l'insuline remplit bien son rôle et les cellules disposent de l'énergie dont elles ont besoin pour fonctionner [2].

Dans le cas d'absence ou insuffisance de l'insuline, le glucose ne peut servir de carburant aux cellules. Il s'accumule alors dans le sang et entraîne une augmentation du taux de sucre (hyperglycémie) [2].

À la longue, un taux de sucre élevé dans le sang entraîne certaines complications, notamment aux niveaux des yeux, des reins, des nerfs, du cœur et des vaisseaux sanguins, d'où les différents types du diabète que nous allons voir un peu plus en détail.

1.2.2 Types de diabète

On distingue plusieurs types de diabète allant du plus courant au plus rare :

1.2.2.1 Diabète de type I

Appelé aussi diabète insulinodépendant, survient quand l'organisme ne produit pas suffisamment d'insuline de façon définitive. Ainsi, quand le corps ne secrète plus d'insuline, la glycémie est plus élevée que la normale. Le diabète de type 1 sera diagnostiqué si sa glycémie à jeun est supérieure à 1,26 g/l ou si sa glycémie, quel que soit le moment de la journée, est supérieure à 2 g/l [3]. Pour pallier le manque d'insuline que le corps ne secrète plus, le patient doit alors faire des injections de cette hormone, plusieurs fois par jour.

Le diabète de type 1 survient généralement chez les sujets jeunes (enfants, ados, adultes jeunes) et les symptômes sont : la fatigue, amaigrissement, besoin et envie d'uriner fréquentes.

1.2.2.2 Diabète de type II

Contrairement au diabète de type 1 qui, dans la majorité des cas survient dans l'enfance et chez le jeune adulte, le diabète de type 2 s'installe, quant à lui, chez un adulte d'âge « mûr ».

Comme il s'accompagne rarement de symptômes à ses débuts, il peut passer inaperçu pendant plusieurs années. Les causes sont multiples et souvent liées aux antécédents familiaux et au mode de vie : obésité, sédentarité et surconsommation alimentaire.

Le diabète de type 2 se distingue aussi du type 1 car, l'insuline est toujours produite, mais en quantité insuffisante par le pancréas qui fonctionne de moins en moins bien [4]. De plus, les cellules sont moins sensibles à l'action de l'insuline (insulino-résistance), ce qui augmente le taux du glucose dans le sang. Ainsi, la glycémie à jeun devient supérieure à 1,26 g/l de manière chronique. Les symptômes du diabète de type 2, quand ils sont présents, ne sont pas forcément les mêmes que ceux du diabète de type 1.

1.2.2.3 Diabète gestationnel

Ce type de diabète débute durant la grossesse (généralement au 2éme trimestre), puis disparaît

dans les semaines suivant l'accouchement. Il est caractérisé par une hyperglycémie (excès de sucre sanguin), due à une résistance temporaire à l'insuline. Cette résistance est liée aux hormones produites par le placenta lors de la grossesse : chez certaines femmes, le pancréas ne parvient pas à compenser leurs effets en produisant plus d'insuline. Avec un bon suivi médical, le diabète gestationnel est rarement dangereux [5]. Il peut néanmoins être révélateur d'un diabète antérieur. D'où la nécessité d'un bon suivi.

1.2.3 Diagnostic

Pour que le médecin puisse proposer un traitement à son patient atteint de l'un de ces différents types de diabète, il est nécessaire de le diagnostiquer. Le vrai problème du diabète est son diagnostic, aucun symptôme ne se présente.

Le principal moyen de savoir si l'on est diabétique est une prise de sang. 50 % des personnes atteintes du diabète ne le savent pas et la plupart d'entre elles sont concernées par le diabète du type 2. Nous savons que plus le diagnostic se fait rapidement, et la prise en charge du diabète commence tôt plus les risques liées à cette maladie se réduisent[6].

La figure ci-dessous montre que plus de 80% des victimes non diagnostiquées sont issues de pays à revenus faibles ou moyens [6].

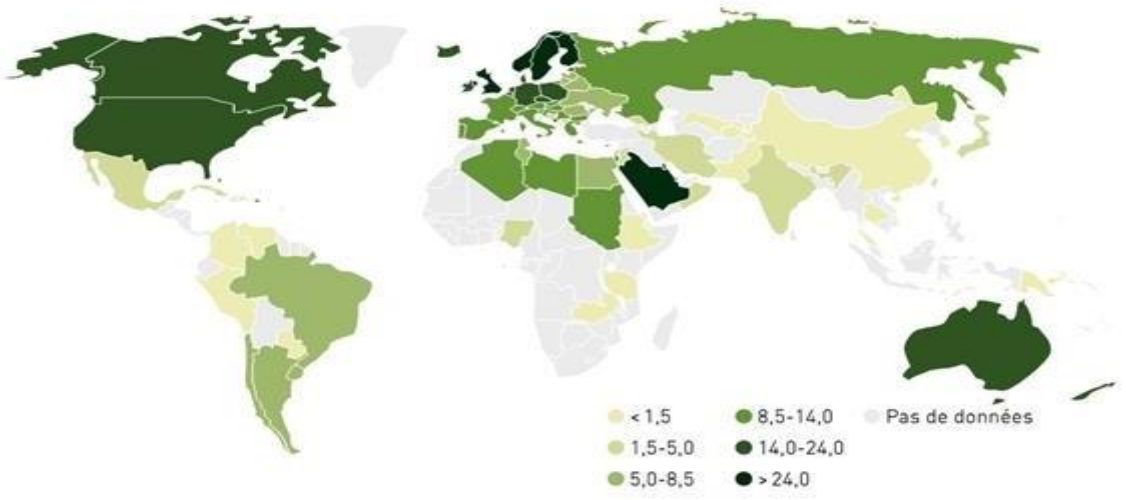


Figure I.1. Prévalence du diabète non diagnostiqué (20-79 ans) en 2011 [6]

1.2.4 Causes du diabète

Il faut bien savoir qu'il n'existe pas de cause précise et exclusive, mais plutôt un ensemble de facteurs favorisants l'apparition du diabète.

Elle peut être d'origine génétique ; des antécédents de diabète du même type sont souvent présents dans la famille, ou des facteurs comportementaux, une alimentation déséquilibrée et un manque d'activité physique contribuent au développement du diabète de type 2.
D'autres facteurs contribuent à l'apparition du diabète de type 2, notamment une surcharge pondérale, l'âge (dès 40 ans), l'origine hispanique, asiatique ou africaine, un taux de cholestérol élevé, des antécédents de diabète gestationnel, une infection par le VIH, certains médicaments et des désordres mentaux (le trouble bipolaire, la dépression, la schizophrénie) [7].

1.2.5 Le diabète dans le monde

Selon l'Organisation Mondiale de la Santé, environ 1,6 million de personnes dans le monde sont décédées des suites du diabète en 2016. On estime à 425 millions le nombre de personnes atteintes de diabète dans le monde entier, selon les projections, ce nombre devrait atteindre 629 millions de diabétiques dans le monde d' ici 2045 [8].

À l'échelle mondiale, le diabète frappe particulièrement les personnes «d'âge moyen» âgées de 40 à 59 ans, ce qui entraîne de graves conséquences économiques et sociales. En outre, le diabète touche particulièrement les pays à revenu faible et intermédiaire.

La Fédération Internationale du Diabète estime que 14,2 millions de personnes vivent avec le diabète en Afrique. La région d'Afrique a le pourcentage le plus élevé de cas de diabète non diagnostiqués, atteignant 80%, le taux de mortalité lié au diabète sucré le plus élevé et les dépenses de santé les plus faibles consacrées au diabète.

La figure ci-dessous illustre le pourcentage du diabète dans le monde en 2017 et une projection pour 2045.

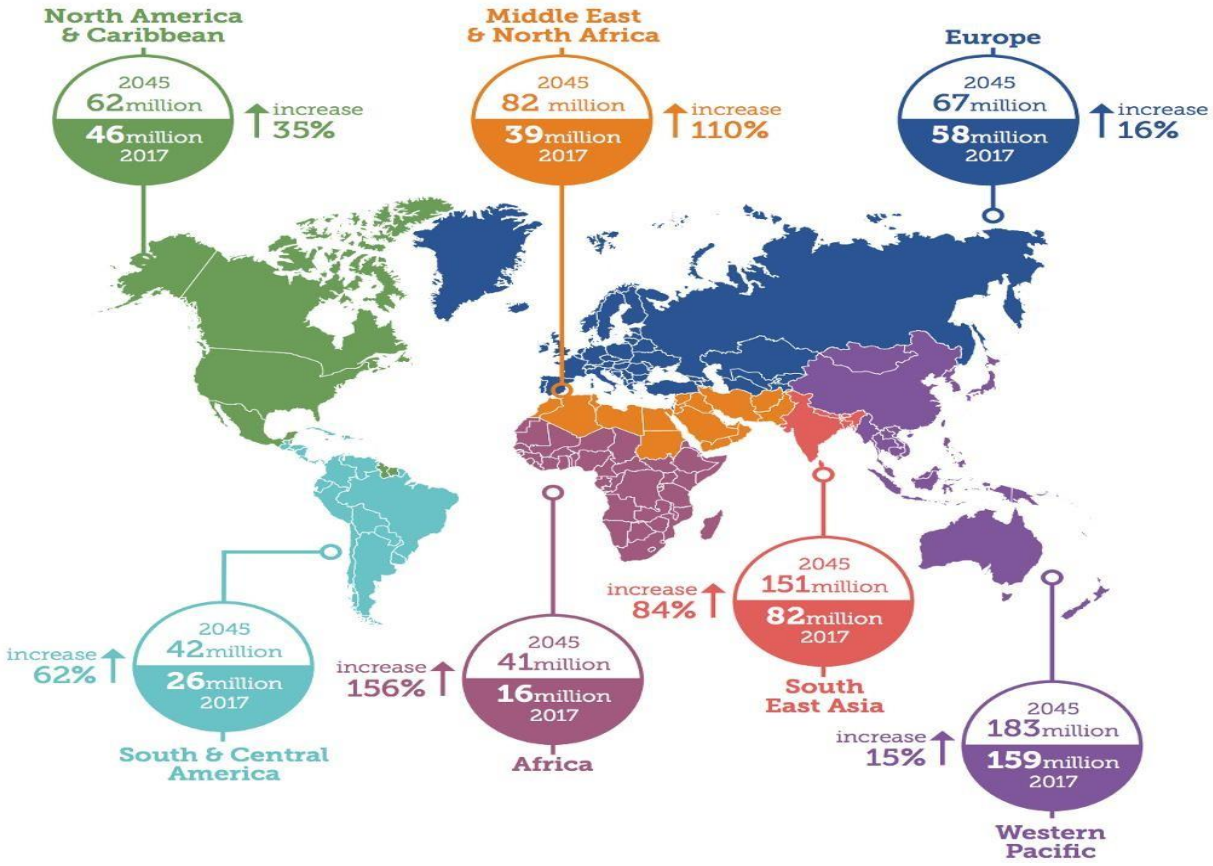


Figure I.2 Diabète dans le monde et par région en 2017. Projection pour 2045 [2]

1.2.6 Le diabète dans en Algérie

Le diabète en Algérie est devenu un véritable problème de santé publique du fait de son ampleur qui devient de plus en plus préoccupante. En effet, le nombre d'algériens atteints par cette maladie chronique s'est considérablement accru ces dernières années et selon les statistiques établies par l'OMS, l'Algérie comptait plus de 4.1 millions de diabétiques, soit une prévalence de 10%, selon la direction de prévention du MSPRH, 150 000 enfants sont diabétiques en 2010. Ces chiffres placent l'Algérie parmi les pays les plus touchés d'où la nécessité de tirer la sonnette d'alarme afin de faire face à cette maladie et ses conséquences désastreuses.

Le diabète est la cause d'une forte morbidité et mortalité au sein de la population algérienne et considéré parmi les principaux motifs d'hospitalisation notamment chez les personnes âgées ce qui confirme le poids de cette maladie parmi les affections chroniques en Algérie.

Selon l'étude TANINA de l'INSP [10], parmi les 10 pathologies chroniques, le diabète est classé en 2éme position juste après l'hypertension artérielle avec 12.33 %.

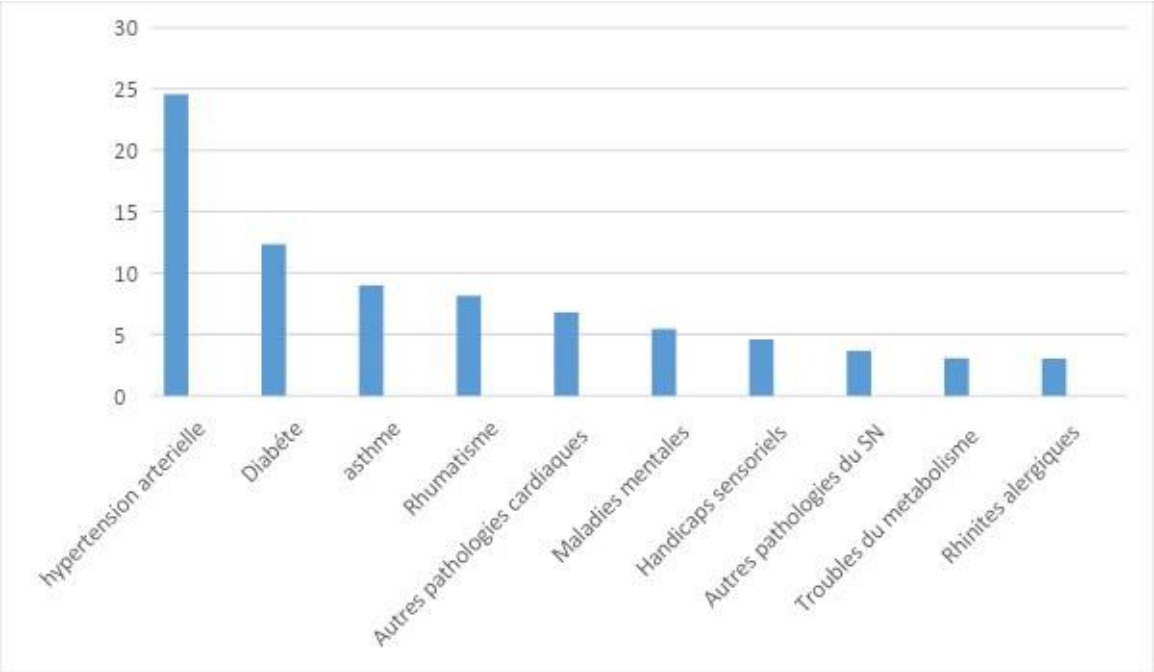


Figure I.3. Répartition des pathologies chroniques en Algérie [10]

Cette fréquence élevée du diabète est à l'origine d'un grand nombre de décès chaque année, ce qui rend cette pathologie la 4ème cause de mortalité en Algérie avec un taux de 7,4% de mortalité par les maladies non transmissibles et 4,4% de mortalité totale [10].

1.3 Importance de la prévention du diabète

La prévention du diabète devient de plus en plus un sujet important, qui doit être étudié dans la recherche scientifique, par rapport aux conséquences dangereuses que ce dernier peut engendrer sur la santé de la personne malade, mais aussi sur le plan économique.

Sur la santé de la personne, le diabète entraîne des complications graves à long terme, pouvant survenir après 10 à 20 ans de déséquilibre glycémique. En effet le diabète peut faire apparaitre plusieurs maladies comme : l'athérosclérose, l'artérite des membres inférieurs, la rétinopathie, des problèmes de cicatrisation, et parfois elle peut déclencher un AVC [8].

Cinq millions de personnes sont mortes, des suites du diabète en 2015. Une personne meurt du diabète toutes les 6 secondes dans le monde, soit plus que le sida, la tuberculose et la malaria [9].

En économie, les problèmes de santé associés au diabète réduisent la productivité par le biais de taux d'absentéisme plus élevés, de productivité réduite au travail et d'invalidités empêchant de travailler et de mortalité précoce.

Le budget total pour le suivi de cette maladie peut être énorme, par exemple aux Etat Unis, le coût total estimé du diabète diagnostiqué en 2017 est de 327 milliards de dollars, dont 237 milliards de dollars en coûts médicaux directs et 90 milliards de dollars en productivité réduite. [12]

1.4 Méthodes de prévention du diabète

Même si la recherche progresse et de nouveaux médicaments apparaissent, le vieil adage "mieux vaut prévenir que guérir" reste le maître mot de la prise en charge du diabète. Et à ce titre, différentes études prouvent aujourd'hui qu'une bonne méthode de prévention est la clé d'une vie plus saine et moins couteuse sur le plan économique.

Eviter le diabète, c'est tout d'abord trouver le bon moyen pour y remédier, dans cette partie nous allons aborder deux méthodes, à commencer par les moyens traditionnels, tels qu'une bonne hygiène de vie qui inclue une alimentation saine, un poids correcte et une activité physique régulière, mais aussi des compagnes de sensibilisation, de dépistage et d'informations. La deuxième méthode quant à elle, concerne les moyens technologiques, basés sur l'utilisation de ML qui se trouve être la nouvelle ère de la science médicale moderne.

1.4.1 Les méthodes traditionnelles de prévention du diabète

La réalité du diabète est méconnue, sous-estimée, voire ignorée dans une indifférence tristement et dangereusement partagée. Plus que jamais, l'information des populations, la formation des soignants, l'accès aux soins, sont des enjeux fondamentaux, ignorés des opinions publiques et négligés par les gouvernements, dans bien des pays du monde.

Les premières mesures pour vivre avec un diabète sont d'ordre alimentaire. Les règles hygiéno-diététiques sont primordiales. L'activité physique a également un rôle important à jouer. Et, surtout, il existe différents traitements, à vie, qui sont efficaces.

Le traitement de référence du diabète et qui doit être entamé avant tout autre, est la modification des habitudes de vie, incluant [11] :

- Une perte de poids quand elle est nécessaire.
- Une activité physique régulière.
- Une alimentation équilibrée.
- Gestion du facteur émotionnel (Stress).
- Le tabagisme, plusieurs études démontrent que les fumeurs sont plus nombreux à développer le diabète de type 2 que les non-fumeurs ou les personnes ayant cessé de fumer.

Mais aussi des compagnes de sensibilisation qui doivent incitées les gens à faire dépistage, faut bien savoir que dans pas mal de cas du diabète la raison peut être d'ordre génétique. Plus on détecte tôt la maladie - même avant l'apparition des symptômes - et plus on intervient tôt pour rétablir une glycémie normale, plus le risque de complications s'amenuise. Les moyens classiques sont nombreux mais le but reste le même.

1.4.2 Machine Learning pour la prévention du diabète

Les progrès remarquables réalisés dans les domaines de la biotechnologie et des sciences de la santé ont conduit à une importante production de données, telles que des données génétiques à haut débit et des informations cliniques, générées à partir de grands dossiers de santé électroniques. À cette fin, l'application des méthodes de ML en biosciences est plus que jamais indispensable afin de transformer intelligemment toutes les informations disponibles en connaissances utiles. L'application de ML et des méthodes dans la recherche sur le diabète est une approche clé pour utiliser de grandes quantités de données disponibles sur le diabète afin d'extraire des connaissances.

1.5 L'intérêt d'un système sur puce pour la prévention du diabète

La prise en charge de la prévention du diabète en faisant appel aux nouvelles technologies de l'information et de la communication se base sur le processus décrit par les deux figures suivantes.

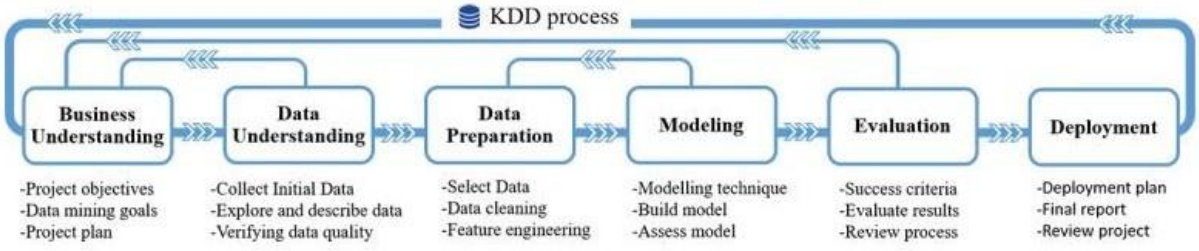


Figure I.4. Schéma explicatif du processus KDD [13]

La figure I.4. explicite le rôle et l'importance des données dans la recherche d'un modèle de prévention.

L'exploration des données se propose d'utiliser un ensemble d'algorithmes issus de disciplines scientifiques diverses telles que les statistiques, l'intelligence artificielle ou l'informatique, pour construire des modèles à partir des données, c'est-à-dire trouver des structures intéressantes ou des motifs selon des critères fixés au préalable, et d'en extraire un maximum de connaissances.

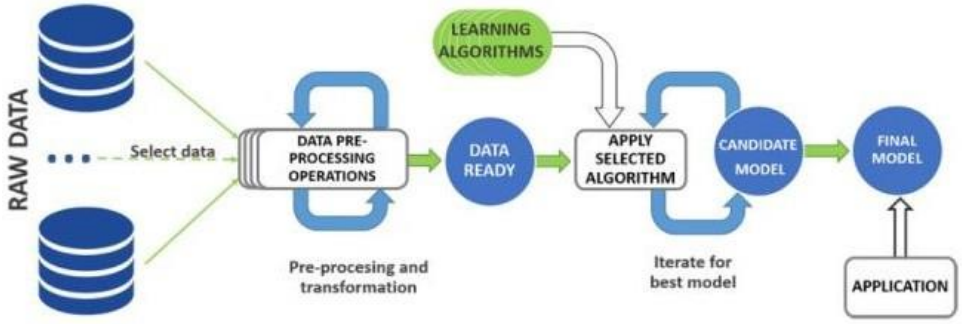


Figure I.5. Diagramme explicatif du processus d'algorithme d'apprentissage [13]

La figure I.5. met en valeur l'importance du traitement dans les phases aussi bien de la recherche du modèle que de son déploiement

Le déploiement peut être localisé aussi bien près des capteurs (donc le patient) qu'au niveau des centres de données (data centers). Le suivi temps réel de la prévention du diabète dépendant de la masse de données à traiter (pendant la phase d'apprentissage) que de la phase de détection peut nécessiter des temps de traitement conséquents en fonction de la taille des données à traiter que de la complexité du modèle à implémenter.

1.6 Architecture générale d'un système de prévention de diabète

Tout scientifique traiteur de données ou passionné de ML qui a essayé d'obtenir la performance sur des modèles d'apprentissage à grande échelle atteindra à un moment donné un plafond et

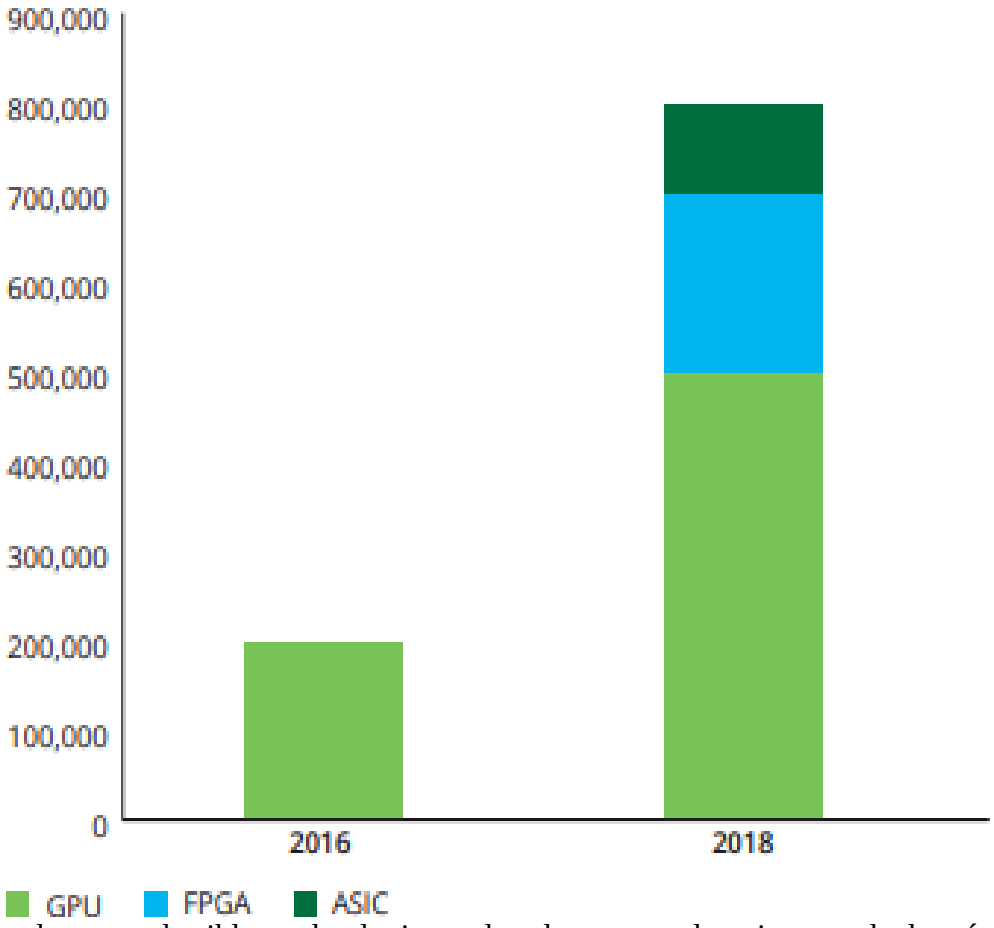
commencera à subir divers degrés de retard de traitement.

Les tâches qui prennent quelques minutes avec de plus petits ensembles de formation peuvent prendre plus de temps - dans certains cas, des semaines - lorsque les ensembles de données deviennent plus grands. Un meilleur matériel de traitement peut être nécessaire s'appuyant sur des cibles technologiques variées ou une combinaison de ces dernières.

Une étude portée par [14] montre l'immersion de plus en plus avérée aussi bien de GPU que des circuits FPGA dans le déploiement de solution dédiées à l'implémentation des modèles d'apprentissage. Par leur coût plus onéreux, les GPU, même s'ils peuvent se présenter efficaces pour le traitement de données massives, peuvent être de plus en plus distancés par le recours à des accélérateurs dédiées. Aidés par des progrès aussi bien dans la technologie des circuits que dans les méthodes et des outils d'implémentation, leurs combinaisons à des architectures systèmes sur puce nous paraissent plus qu'opportune.

Dans un article récent, Deloitte Global prédit qu'à la fin de 2018, plus de 25% de toutes les puces destinées à accélérer l'apprentissage automatique seront des ASIC (et des FPGA). Lorsque cela se produit, les applications activées par l'apprentissage automatique risquent de provoquer de grands changements dans l'industrie tout en se développant dans de nouveaux domaines.

Figure I.6



Evolution des types de cibles technologiques dans les centres de traitement de données [14]

1.7 L'apport de la méthodologie dans la conception d'un système sur puce dédié

Une combinaison de la méthodologie décrite par les deux figures ci-haut (KDD) avec celle dédiées au développement de circuits peut être pertinente. Couplé à des solutions open sources bien choisies sur des critères d'évaluation objectifs, on est certain que le prototypage d'application appliqué au ML sera d'une efficacité avérée. Ce sera le cœur de la question que nous proposons de résoudre dans ce présent mémoire.

1.8 Conclusion

Nous avons vu dans ce chapitre les types de diabète, les différents traitements et tests ainsi que les complications dues à cette maladie. Même s'il existe des méthodes de prévention qui permettent de réduire le risque d'avoir le diabète, parfois il est impossible de l'éviter comme pour le diabète de type 1. Dans ces cas-là, la seule solution est de

pouvoir le diagnostiquer très tôt et faire tout son possible pour combattre les complications qui peuvent être mortelles.

Pour notre cas d'étude nous allons utiliser le ML pour la prévention du diabète, dans le chapitre qui suit, l'objectif principal est d'appliquer les différents algorithmes de classification de le ML sur des données concernant les risques de développer un diabète, et choisir l'algorithme le plus approprié pour notre étude.

Chapitre II

Prévention du diabète avec Machine Learning

2.1 Introduction

Nous avons évoqué dans le chapitre précèdent quelques méthodes de prévention du diabète. Nous nous intéressons dans ce chapitre sur la prévention du diabète considéré comme problème de classification en utilisant de Machine Learning.

En Machine Learning, un problème de classification se présente sous la forme d'un ensemble de données, contenant des exemples issus de l'observation d'un phénomène. Chaque exemple est constitué d'une description et d'une étiquète. Un algorithme d'apprentissage analyse ses données afin de construire un classificateur. C'est à ce classificateur que revient ensuite la tâche d'étiqueter de nouveaux exemples à partir de leur description.

Afin d'étudier le problème de classification auquel nous nous intéresserons, dans un premier temps nous définirons ce qu'est le Machine Learning, après nous présenterons l'état de l'art où nous effectuerons une comparaison entre certains travaux passés en revue concernant l'application des méthodes d'apprentissage dans le domaine médical, notamment dans le diabète. Pour finir nous présenterons la base de données utilisée, sur laquelle nous ferons une comparaison de performances de plusieurs algorithmes de ML à l'aide de plusieurs outils, afin de choisir l'algorithme le plus approprié pour notre cas d'étude.

2.2 Machine Learning

Le ML ou **«** apprentissage automatique **»** [15] en français est un concept qui fait de plus en plus parler de lui, et qui se rapporte au domaine de l'intelligence artificielle. Encore appelé « apprentissage statistique », ce terme renvoie à un processus de développement, d'analyse et d'implémentation conduisant à la mise en place de procédés systématiques. Pour faire simple, il s'agit d'une sorte de programme permettant à un ordinateur ou à une machine un apprentissage automatisé, de façon à pouvoir réaliser un certain nombre d'opérations très complexes.
L'objectif visé est de rendre la machine ou l'ordinateur capable d'apporter des solutions à des problèmes compliqués, par le traitement d'une quantité astronomique d'informations.
Cela offre ainsi une possibilité d'analyser et de mettre en évidence les corrélations qui existent entre deux ou plusieurs situations données, et de prédire leurs différentes implications.

Le ML présente plusieurs types d'apprentissage, selon les données fournis et le problème posé, le schéma de la figure II.1 résume quelque algorithme de ML

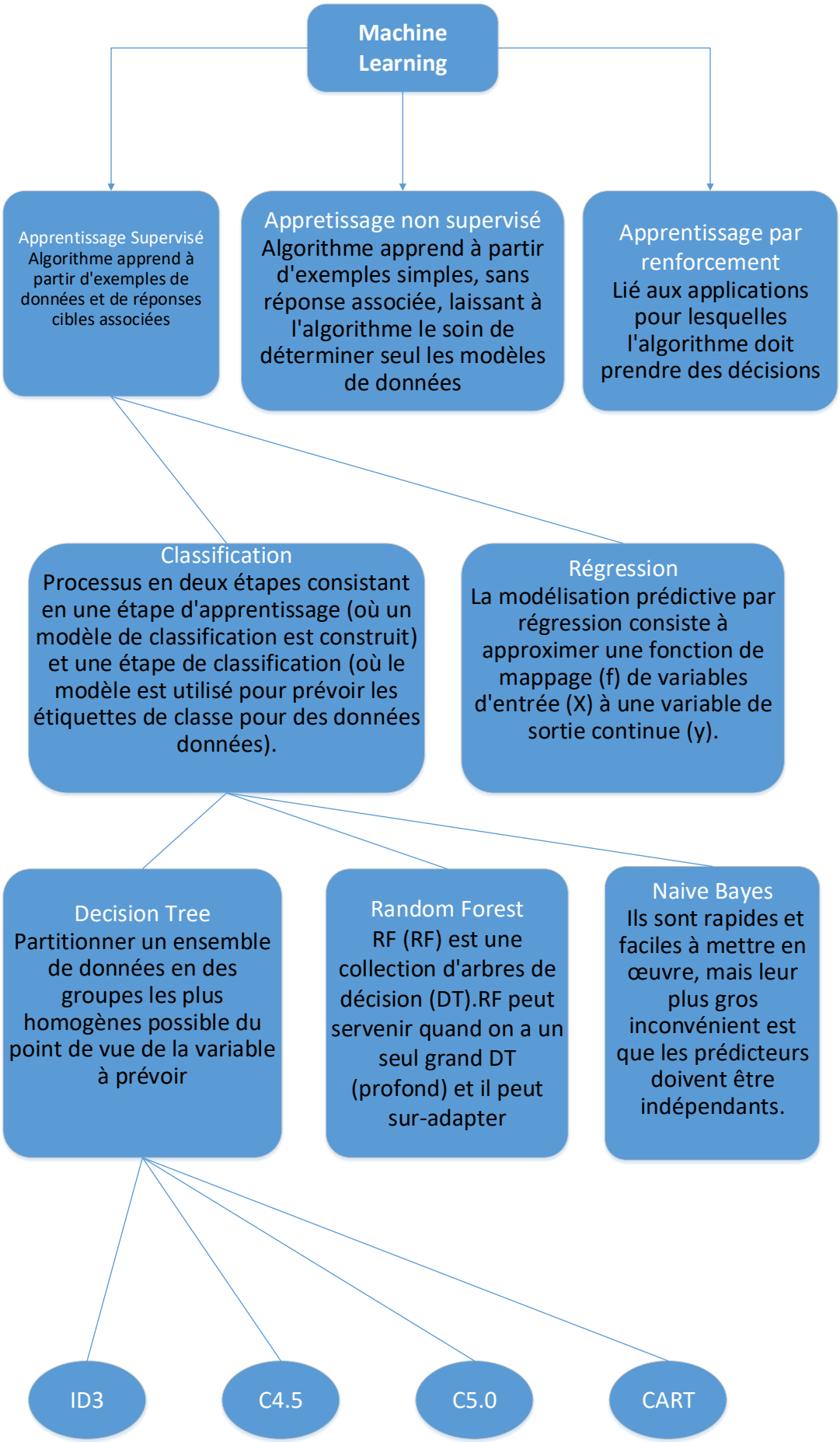


Figure II.1. Types d'apprentissage pour la Machine Learning

Il existe d'autres algorithmes d'apprentissages qui sont très utilisé, nous citons quelque uns :

Définition de l'algorithme SVM : L'algorithme Machine à vecteurs de support (SVM) est un ensemble de méthodes d'apprentissage supervisé utilisées pour la classification, la régression et la détection des valeurs aberrantes. Il est capable d'effectuer une classification multi-classes sur un ensemble de données qui peut être utilisé pour la classification, la régression ou d'autres tâches. Intuitivement, une bonne séparation est obtenue par l'hyper-plan qui a la plus grande distance des points de formation les plus proches de n'importe quelle classe, car plus la marge est grande, plus l'erreur de généralisation du classificateur est faible [16].

Le plus proche voisin : En reconnaissance de forme, l'algorithme des k plus proches voisins (k-NN) est une méthode non paramétrique utilisée pour la classification et la régression. Dans les deux cas, il s'agit de classer l'entrée dans la catégorie à laquelle appartiennent les k plus proches voisins dans l'espace des caractéristiques identifiées par apprentissage. Le résultat dépend si l'algorithme est utilisé à des fins de classification ou de régression [17].

Méthodes d'ensemble : Le but des méthodes d'ensemble est de combiner les prédictions de plusieurs estimateurs de base construits avec un algorithme d'apprentissage donné, afin d'améliorer la prédiction et rendre un estimateur plus robuste.

- **Bagging Trees** : une des premières méthodes historiquement, selon laquelle on construit plusieurs arbres de décision en ré-échantillonnant l'ensemble d'apprentissage, puis en construisant les arbres par une procédure de consensus [18].

2.3 Evaluation des performances pour différents algorithmes de classification et choix de l'algorithme approprié

Le plus difficile pour résoudre un problème de ML consiste souvent à trouver l'estimateur approprié pour une application ou un problème donné.

Différents estimateurs conviennent mieux à différents types de données et à différents problèmes.

Pour choisir l'algorithme le plus approprié, nous avons d'abord étudié les travaux antérieurs rapportés dans la littérature sur la prévention du diabète avec le ML, ensuite nous avons renforcé notre choix en faisant une comparaison entre différents algorithmes de classification, en utilisant notre base de données sur différentes plateformes.

2.3.1 Etat de l'art

Le ML est une technologie de plus en plus utilisée dans toutes les industries. En industrie des transports pour le développement d'un système de navigation sans conducteur, en domaine bancaire où l'on cherche à estimer la capacité d'une personne à rembourser un prêt ou en secteur médical où les machines Learning aident à diagnostiquer le cancer, le diabète, dont la prédiction de ce dernier constitue le point principal dans ce travail .

Dans le cadre de la prévision du diabète plusieurs travaux ont été réalisés dans ce domaine.

Dans [19], Song et al. ont décrit et ont expliqué les différents algorithmes de classification utilisant différents paramètres tels que le glucose, la pression artérielle, l'épaisseur de peau, l'insuline, l'IMC, le diabète, et l'âge. Dans cette recherche, les chercheurs n'utilisaient que de petits échantillons de données pour prédire le diabète. Dans cet article, les algorithmes utilisés étaient cinq algorithmes différents : GMM, ANN, SVM, EM et régression logistique. Finalement. Les chercheurs ont conclu que le réseau de neurones artificiels (ANN) fournissait une précision élevée pour la prédiction du diabète.

Dans L'étude de Pradeep et Naveen [20], les performances des techniques d'apprentissage automatique ont été comparées et mesurées en fonction de leur précision. La précision de la technique varie d'avant le prétraitement et après le prétraitement, tels qu'ils ont été identifiés dans cette étude. Cela indique que, dans la prévision des maladies, le prétraitement de l'ensemble de données a son propre impact sur la performance et la précision de la prévision. Les chercheurs ont démontré que la technique de l'arbre décisionnel offre une meilleure précision dans cette étude avant le prétraitement pour prédire les maladies du diabète. Alors que la technique de Random forest et la machine à vecteurs de support fournissent une meilleure prédiction après le prétraitement dans cette étude en utilisant le jeu de données sur le diabète.

Xue-Hui Meng et al. [21] utilisaient différentes techniques d'exploration de données pour prédire les maladies du diabète en utilisant des ensembles de données du monde réel en collectant des informations auprès d'un interrogateur distribué. Les chercheurs ont comparé trois techniques : ANN, régression logistique et j48. Enfin, il a été conclu que la technique d'apprentissage automatique j48 fournit une précision efficace et meilleure.

Une comparaison a été établie entre les algorithmes J48, CART, SVM, et kNN pour la prévention du diabète [22] , les auteurs ont utilisé une base de données contenant 10 attributs

pour 545 patients, les résultats avec l'outil Weka montre que l'algorithme J48 donne de meilleurs résultats avec 67.15% d'Accuray.

2.3.2 Evaluation des performances avec différentes plateformes de Machine Learning

2.3.2.1 Description de la base de données utilisée

Cet ensemble de données est à l'origine de l'Institut national du diabète et des maladies digestives et rénales. L'objectif de l'ensemble de données est de prédire de manière diagnostique si un patient est diabétique ou non, en fonction de certaines mesures de diagnostic incluses dans l'ensemble de données. Plusieurs contraintes ont été placées sur la sélection de ces instances dans une base de données plus grande. En particulier, tous les patients dans cette base sont des femmes d'au moins 21 ans et d'origine indienne.

Les jeux de données comprennent 768 instances, avec neuf attributs qui présentent huit variables prédictives médicales et une variable cible, qui indique si le patient est diabétique ou non. Les variables prédictives incluent :

- Nombre de fois enceinte
- La concentration de glucose plasmatique à 2 heures dans un test de tolérance au glucose par voie orale
- Tension artérielle diastolique (mm Hg)
- Épaisseur du pli cutané des triceps (mm)
- Insuline sérique 2 heures (mu U / ml)
- Indice de masse corporelle (poids en kg / (taille en m) ^ 2)
- Fonction pedigree du diabète : une fonction qui évalue la probabilité de diabète en fonction des antécédents familiaux
- Age (Années)
- Variable de classe (0 ou 1)

2.3.2.2 Mesures de performance dans le domaine du ML

- **Accuracy**

C'est le rapport entre le nombre de prévisions correctes et le nombre total d'échantillons d'entrée.

$$Accuracy = \frac{Nombre\ de\ prévisions\ correctes}{Nombre\ total\ d'échantillons} \qquad (1)$$

- **AUC**

L'AUC est l'un des paramètres d'évaluation les plus utilisés. Il est utilisé pour le problème de classification binaire. Pour définir l'AUC, il faut comprendre les deux termes suivant :

True Positive Rate (Sensibilité)

$$True\ Positive\ Rate\ (TPR) = \frac{TRue\ Positive\ (TP)}{False\ Negative\ (FN) + True\ Positive\ (TP)} \quad (2)$$

Le TPR correspond à la proportion de points de données positifs qui sont correctement considérés comme positifs par rapport à tous les points de données positifs.

False Positive Rate (spécificité)

Le FPR correspond à la proportion de points de données négatifs considérés à tort comme positifs, par rapport à tous les points de données négatifs.

$$True\ Positive\ Rate\ (FPR) = \frac{False\ Positive\ (FP)}{False\ Positive\ (FP) + True\ Negative\ (TN)} \quad (3)$$

Le False Positive Rate et le True Positive Rate ont tous les deux des valeurs comprises entre 0 et 1. L'AUC est l'aire sous la courbe du graphique FPR vs TPR à différents points entre 0 et 1. Plus la valeur d'AUC est élevée, meilleure est la performance du modèle.

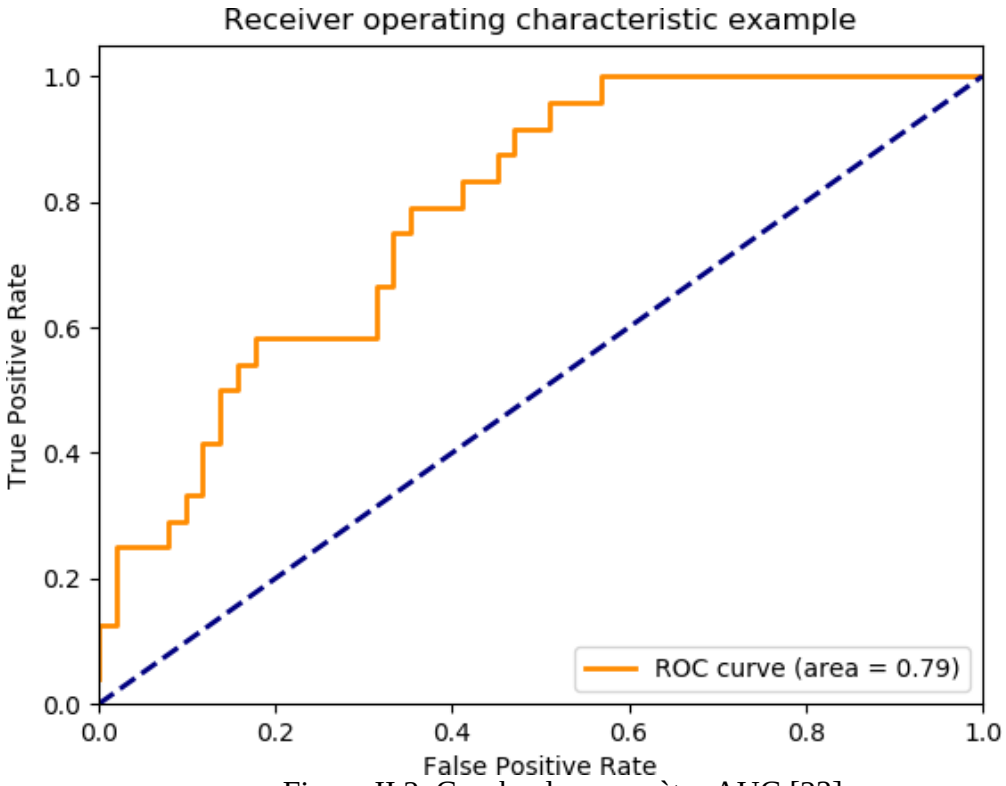


Figure II.2. Courbe du paramètre AUC [23]

➢ Précision

La précision est une bonne mesure à déterminer lorsque les coûts du FP sont élevés.

$$True\ Positive\ Rate\ (TPR) = \frac{True\ Positive\ (TP)}{False\ Positive\ (FN) + False\ Positive\ (TP)} \quad (4)$$

Dans notre étude nous avons pris, les deux paramètres Accuracy et AUC, comme mesures d'évaluation des performances des algorithmes de ML, car le paramètre Accuracy il mesure la capacité de l'algorithme à prédire l'occurrence ''oui'', et le paramètre AUC, il prend en plus en considération le taux de fausses alarmes.

2.3.2.3 Evaluation des performances avec MATLAB

a- Classification Learner

Cette application permet d'explorer l'apprentissage automatique supervisé à l'aide de divers classificateurs, d'explorer les données, former des modèles et évaluer les

résultats. Les classificateurs incluent dans cette application sont: DT, SVM, la régression logistique, KNN et la classification d'ensemble. [24]

L'apprentissage est effectué en fournissant un ensemble connu de données d'entrée et des réponses connues pour ces entrées (des classes dans notre étude). Ces données sont utilisées pour former un modèle. Pour utiliser le modèle avec de nouvelles données l'application permet d'exporter le modèle sur l'espace travail et de tester de nouvelles données.

Nous avons pris 80% de données pour l'apprentissage et 20% pour le test.

b- Résultats et discussion

Le tableau II.2. Regroupe les résultats d'évaluation des performances en termes d'accuracy et de recall pour plusieurs classificateurs sous Matlab. Nous avons considéré des classificateurs de type DT et un de ses ensembles (bagging), le SVM, le KNN.

Tableau II.1. Comparaison entre les algorithmes de classification sous Matlab

Algorithme	Accuracy	AUC
DT (Profondeur = 2, CART)	73.3%	0.79
DT (Profondeur = 5, CART)	**76.0%**	**0.81**
DT (Profondeur = 12, CART)	71.5%	0.73
Bagged Trees	75.5%	0.80
SVM	75.9%	0.81
KNN	70.4%	0.72

Les résultats de comparaison sous l'application classification Learner, montrent que tous les algorithmes présentent de bonnes performances en terme des deux paramètres accuracy et AUC. Les performances marquées par l'utilisation des arbres de décision avec une profondeur de cinq sont légèrement supérieurs aux autres algorithmes.

2.3.2.4 Evaluation des performances avec Weka

a- Description

Weka [25] est une suite de logiciels d'apprentissage automatique écrite en Java et développée à l'université de Waikato en Nouvelle-Zélande et qui sont disponibles sous la Licence publique générale GNU.

Nous avons pris 80% de données pour l'apprentissage et 20% pour le test.

b- Résultats et discussion

Le tableau II.3 regroupe les résultats d'évaluation des performances en termes d'accuracy, de recall et de précision pour plusieurs classificateurs sous Weka. Nous avons considéré des classificateurs de type DT et ses ensembles (bagging), le SVM, le KNN et le Naive Bayes.

Tableau II.3. Comparaison entre les algorithmes de classification sous WEKA

Algorithme	Accuracy	AUC
Naive Bayes	76.30%	0.819
Bagging (j48)	75.2%	0.810
Decision Tree (j48, C4.5)	74.21%	0.764
KNN	70.18%	0.650
SVM	69.27%	0.570

Nous remarquons qu'avec l'utilisation de la base de données sur le diabète sous Weka, tous les algorithmes présentent de bonnes performances en termes de d'accuracy et de AUC.

2.3.2.5 Evaluation des performances avec Python (Scikit Learn)

a- Bibliothèque de Scikit Learn

Scikit-learn [26] est une bibliothèque en Python qui fournit de nombreux algorithmes d'apprentissage non supervisés et supervisés. Python est un langage de programmation de haut niveau. Créé par Guido van Rossum et publié pour la première fois en 1991. Python propose un système de type dynamique et une gestion automatique de la mémoire. Il prend en charge plusieurs paradigmes de programmation, notamment orienté objet, impératif, fonctionnel et procédural, et dispose d'une bibliothèque standard étendue et complète.

Nous avons pris 80% de données pour l'apprentissage et 20% pour le test.

a- Résultats et discussion

Le tableau II.4 regroupe les résultats d'évaluation des performances en termes d'accuracy, et de recall pour plusieurs classificateurs avec la bibliothèque Scikit Learn.

Tableau II.4. Comparaison entre les algorithmes de classification sous ScikitLearn

Algorithme	Accuracy	AUC
DT (Profondeur = 3, CART)	86.36%	0.852
DT (Profondeur = 5, CART)	**88.31%**	**0.873**
DT (Profondeur = 12, CART)	78.5%	0.74
Bagged Trees	83.76%	0.767
SVM	83.76%	0.742
KNN	81.16%	0.714

Les résultats de comparaison avec l'utilisation de la bibliothèque de Scikit Learn, montrent que tous les algorithmes présentent de bonnes performances en terme des deux paramètres accuracy et AUC. Les performances marquées par l'utilisation des arbres de décision avec une profondeur de cinq sont légèrement supérieurs aux autres algorithmes.

2.3.2.6 Comparaison des résultats obtenus pour DT

Le tableau II.5 regroupe les résultats d'évaluation des performances en termes d'accuracy, et d'AUC pour le classificateur de type DT sous les trois plateformes. Les résultats montrent que l'évaluation sous ScikitLearn donne de meilleurs résultats en termes d'AUC et d'accuracy.

Tableau II.5. Comparaison des performances pour l'DT dur les trois plateformes

Plateformes	Matlab (DT-CART)	Weka (J48-C4.5)	ScikitLearn (DT-CART)
Accuracy	75.4%	74.21%	88.31%
AUC	0.80	0.764	0.873

2.4 Conclusion

Le Machine Learning ou l'apprentissage automatique est le sous domaine de l'informatique qui offre aux ordinateurs la possibilité d'apprendre le comportement d'une base de données, et tirer un modèle qui illustre son mécanisme de fonctionnement, ce qui montre la corrélation entre le ML et la prévention du diabète. Le ML présente plusieurs types d'algorithmes, le choix de l'algorithme approprié est une étape importante pour l'obtention de meilleurs résultats. Pour cela nous avons consulté les études faites dans le domaine de la prévention du diabète avec le ML, et nous avons aussi fait des comparaisons entre ces différents algorithmes. Nous concluons que les DT donnent une meilleure prévision par rapport aux autres algorithmes, d'où l'utilisation de cet algorithme pour l'obtention du modèle de prévention du diabète.

Chapitre III :

Decision Tree

3.1 Introduction

L'apprentissage par l'utilisation des arbres de décision est une méthode couramment utilisée dans l'exploration de données, l'objectif est de créer un modèle qui prévoit la valeur d'une variable cible en fonction de plusieurs variables d'entrée (attributs).

Nous présenterons dans ce chapitre quelques notions théoriques des arbres de décision ainsi que les avantages de leurs utilisations, nous allons par la suite justifier la plateforme par la laquelle nous obtiendrons notre modèle.

3.2 Les arbres de décision

3.2.1 Description

Les arbres de décision représentent une méthode très efficace d'apprentissage supervisé. Il s'agit de partitionner un ensemble de données en des groupes les plus homogènes possible du point de vue de la variable à prédire. On prend en entrée un ensemble de données classées, et on fournit en sortie un arbre qui ressemble beaucoup à un diagramme d'orientation où chaque nœud final (feuille) représente une décision (une classe) et chaque nœud non final (interne) représente un test. Chaque feuille représente la décision d'appartenance à une classe des données vérifiant tous les tests du chemin menant de la racine à cette feuille.

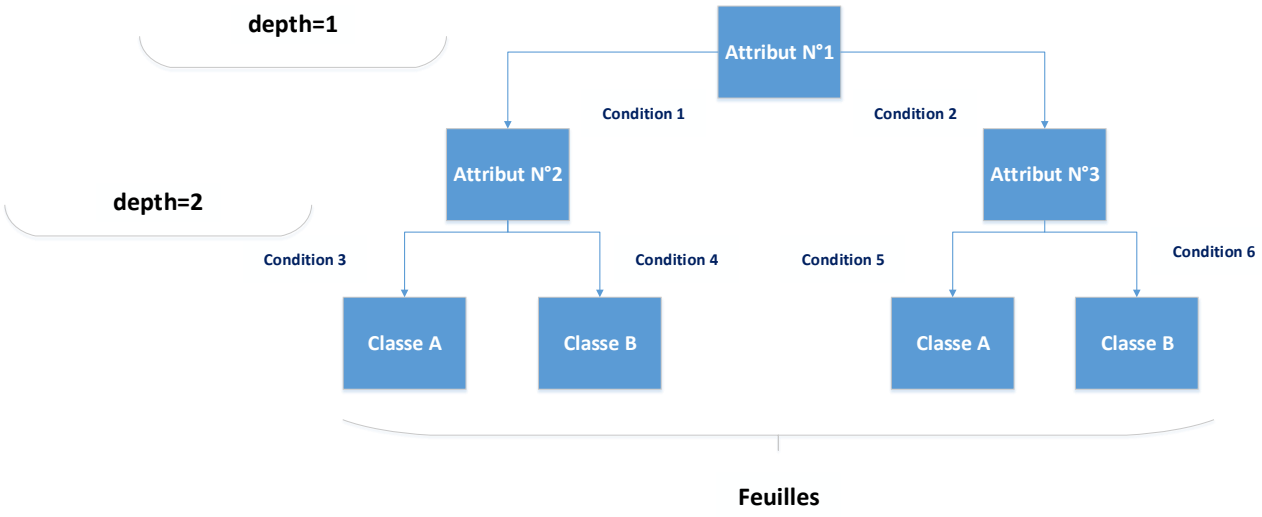


Figure III.1. Schéma de la structure d'un DT (profondeur=2)

3.2.2 Avantages du DT

- La simplicité de compréhension et l'interprétation ;
- Nombre de tests limités par le nombre d'attributs ;
- Construction efficace à l'aide d'apprentissage par optimisation ;
- Performant sur de grands jeux de données : la méthode est relativement économique en termes de ressources de calcul ;
- Le modèle peut gérer à la fois des valeurs numériques et des catégories.

3.2.3 Algorithmes d'apprentissage de DT

Les différentes façons de répartition des attributs sur le DT (critère de fractionnement) et choix du nœud racine sont l'objet de nombreux algorithmes de DT.

- **Algorithme ID3** : Cet algorithme est basé sur le calcul de l'entropie de Shannon et le gain d'information.

- **Algorithme C4.5/C5** : C4.5, successeur d'ID3, utilise une extension au gain d'information appelée ''Gain Ratio'', C4.5 a été remplacé en 1997 par un C5.0/See5 (C5.0 pour Unix / Linux, See5 pour Windows). Le C5.0 est l'algorithme de classification qui s'applique dans les bases données volumineuses. Le C5.0 est meilleure que C4.5 sur l'efficacité, la mémoire et la vitesse.

- **Algorithme CART** : cet algorithme construit un DT strictement binaire avec exactement deux branches pour chaque nœud de décision, CART utilise l'indice de Gini [27] comme critère de division des attribues et leurs répartition sur l'arbre.

3.3 Méthodologies de mise en œuvre

3.3.1 Etat de l'art

Bien qu'il n'existe aucun historique de comparaison de plateformes de ML, dans une étude récente [28], les chercheurs de L'Université Arizona State, ont trouvé que la comparaison sur la base de la précision et du F-score, montre que toutes les plates-formes fonctionnent aussi bien.

La statistique de test Iman – Davenport [29], calculée à partir des rangs moyens des valeurs de l'AUC, rejette l'hypothèse selon laquelle toutes les plateformes fonctionnent de manière égale, lorsqu'on considère leur meilleure précision de classification pour tous les algorithmes.

Le test de Holm montre qu'Azure ML [30], Python (Scikit Learn) et SAS sont les plus performantes des cinq plateformes lorsque le meilleur score d'AUC sur un ensemble d'algorithmes est utilisé comme métrique de performance.

3.3.2 Discussion

À partir des résultats obtenus dans l'article sur la comparaison des performances des plateformes, et la grande capacité d'ajustement que présente l'utilisation du Python avec les bibliothèques de Scikit Learn, nous choisissons ce dernier pour obtenir le modèle DT pour notre base de données.

3.4 Obtention du modèle

Tel que l'étude précédente a justifié le choix de bibliothèque de Scikit Learn pour entraîner nos données avec les DT, dans cette étape nous avons exploité encore cette bibliothèque pour obtenir le modèle. [Annexe] présente notre code écrit en python pour l'obtention du modèle du DT.

3.4.1 Paramètre choisis pour l'obtention du DT

Pour obtenir le modèle DT, la bibliothèque de Scikit Learn nous donne la possibilité de faire plusieurs ajustements qui concernent les caractéristiques de l'arbre, nous citons quelque uns :

- **Profondeur de l'arbre** : précise combien de niveau notre arbre doit contenir. Nous avons fixé ce paramètre à 5.
- **Critère de division** : c'est le critère qui décide dans quel sens l'arbre doit se propager, et les attributs dans chaque nœud. Dans notre cas nous avons choisi le Gini Index comme critère de division, qui représente l'algorithme CART.
- **Précision** : cette précision concerne les valeurs d'attributs, dans notre cas nous l'avons fixé à 1 (un chiffre après la virgule).

3.4.2 Le modèle obtenu

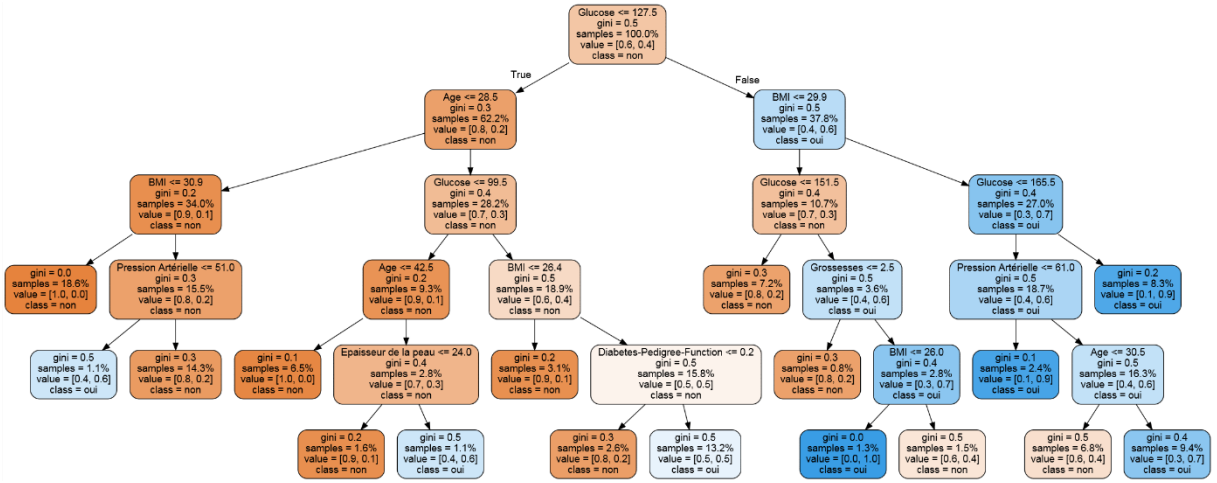


Figure III.2. Le DT obtenu en utilisant Scikit Learn

Le paramètre ‘’value ‘’ indique combien d'échantillons du nœud donné entrent dans chaque catégorie.

Le paramètre ‘’samples’’ indique le nombre d’échantillons traités au niveau du nœud donné.

3.5 Conclusion

Dans ce chapitre nous avons évoqué quelques aspects théoriques concernant les DT,pour l’obtention de notre modèle, nous avons choisis l’utilisation de la bibliothèques de Scikit Learn, pour la large documentation que cette dernière présente, sa performance assurée par une étude très récente mais aussi sa grande capacité d’ajustement comme la profondeur du DT que nous avons fixé à 5 (la profondeur donnant les meilleurs performance) et l’algorithme CART comme algorithme de construction du DT qui utilise le Gini Index comme critère de division.

L’étape suivante consiste à concevoir un accélérateur matériel pour le modèle obtenu.

Chapitre IV

Implémentation du modèle

4.1 Introduction

Dans le précédent chapitre, nous avons obtenu notre modèle DT, correspondant au système de prévention du diabète, nous sommes actuellement confrontés à une autre tâche qui est l'implémentation hardware de ce modèle. En effet ce système de prévision peut être utilisé dans d'autres travaux complémentaires comme unité de traitement dans un cloud, pour pouvoir gérer une grande quantité de requêtes. La solution envisagée est l'implémentation hardware sur une carte FPGA.

Nous allons d'abord suivre une méthodologie pour le choix du processus de conception de notre modèle, par lequel nous obtiendrons l'IP core qui va être implémenté sur une cible SoC, afin de le tester en utilisant un environnement de développement software.

4.2 Méthodologies d'implémentation

Plusieurs études ont été faites pour analyser l'utilisation du HLS par rapport aux autres méthodes d'implémentation. Nous citons les conclusions obtenues de l'article [31] :

La notion de DP a été définie, ainsi qu'une méthode permettant d'évaluer les gains de la productivité d'une méthode HLS par rapport à une description manuelle du format HDL. En utilisant cette méthode, un compilateur HLS a été comparé au VHDL manuel. Cette étude a montré les résultats suivant :

- La méthode HLS est efficace en termes de fréquence, elle obtient une période minimale légèrement meilleure que le VHDL manuel.
- Cependant, HLS présente une surcharge importante en termes de LUT et de registres.
- En moyenne, la conception avec la méthode HLS a pris * 4,42 moins de temps que l'écriture et le test VHDL à la main.
- La perte de qualité reflète la perte due à l'augmentation du niveau d'abstraction. Il y a une perte de qualité d'environ *2 due à l'utilisation de la méthode HLS par rapport à l'écriture manuelle VHDL.
- Globalement, la notion DP qui évalue le compromis entre qualité de conception et efficacité de la conception, montre que HLS atteint une productivité de conception supérieure à celle du VHDL.

4.3 Environnements d'implémentation HLS

Plusieurs études ont été faites pour analyser les différents outils HLS, plusieurs outils ne sont plus utilisés de nos jours ou ont été amélioré et ont été commercialisé sous un autre nom.

Nous allons citer dans ce qui suit les outils les plus utilisés pour la synthèse haut niveau [32] :

1. **Vivado HLS**

Vivado HLS a été développé par Xilinx. L'outil reçoit en entrée des fichiers sources en : C, C ++ et SystemC. Le pilote automatique peut optimiser la latence et la consommation d'énergie.

Lors de la génération RTL, le code vhdl est souvent très long par rapport au code C, le code généré est lisible mais les noms des variables générées sont très difficiles à comprendre.

L'outil donne la possibilité d'optimiser le modèle, évaluer ses performances et prend aussi en charge plusieurs types d'interfaces.

2. **BlueSpec**

BlueSpec, un outil de BlueSpec Inc, l'outil reçoit en entrée des fichiers en BSV (BSV).
BlueSpec analyse le code source ligne par ligne et génère le code RTL à partir de là. Les noms de signaux de la conception BSV ont été conservés dans le code RTL. Cela a un impact extrêmement positif sur la lisibilité du code généré.
L'environnement BSV n'offre aucune capacité d'exploration de la conception, car l'entrée de conception se fait au niveau matériel.
Cet outil présente plusieurs inconvénients tels que la difficulté d'implémentation, optimisation du modèle très limitée,…

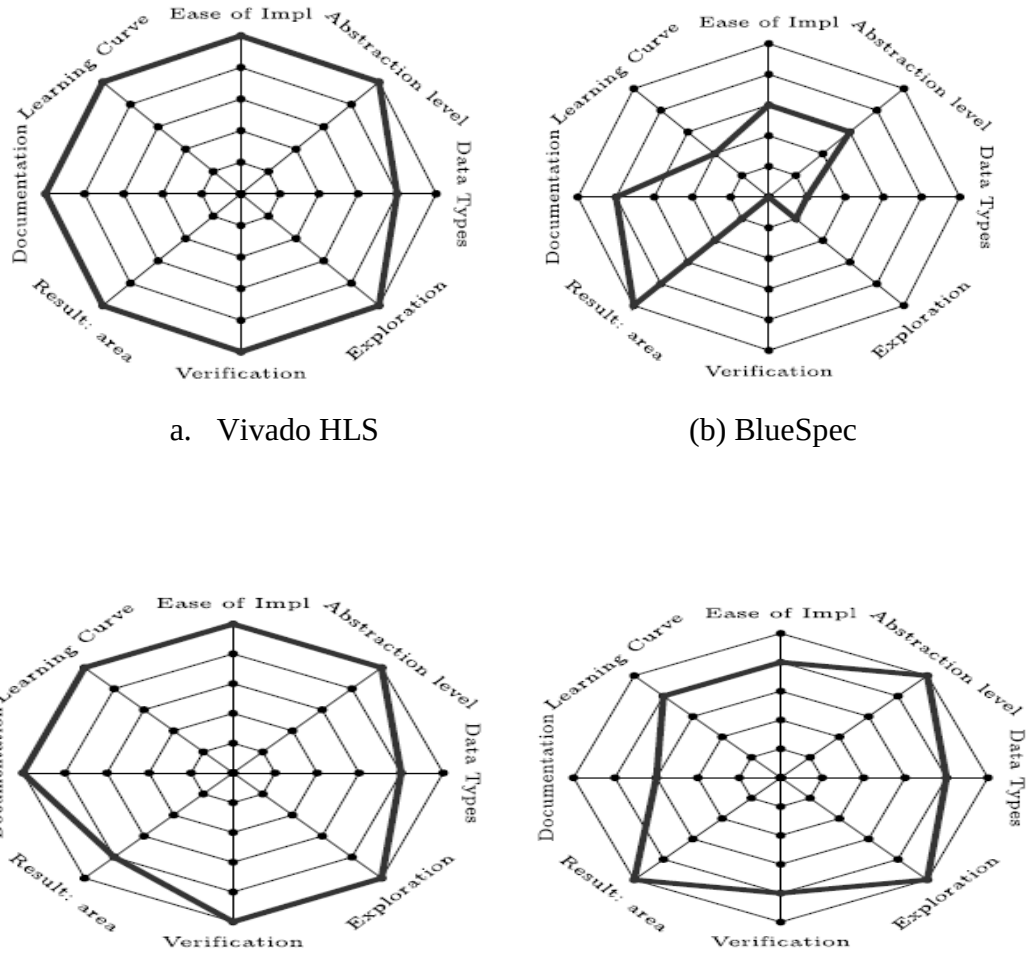
3. Catapult C

Catapult C appartient à Calypto Design Systems, acquis de Mentor Graphics en août 2011.
CatapultC accepte les fichiers sources C, C++ et SystemC et offre une excellente interface qui guide le concepteur tout au long du processus HLS.
Les étapes comprennent la sélection des fichiers source (design et test bench), la configuration (technologie cible, fréquence d'horloge, etc.), les contraintes de conception (zone et latence, mappage E / S et optimisations de boucle), la planification et la génération de RTL.

4. CyberWorkBench

CyberWorkBench (CWB) est un outil HLS de NEC. C, SystemC et BDL (BDL) sont pris en charge. Cet outil permet au concepteur de limiter le nombre de ressources (multiplicateurs, additionneurs, etc.) et en fonction de la licence, la conception résultante peut être écrite en VHDL ou Verilog.

Pour le choix de l'environnement d'implémentation, nous nous sommes basées sur les digrammes « Radar chart » [15], le choix se fait sur plusieurs critères : facilité d'implémentation, disponibilité de la documentation, optimisation du modèle, vérification avec le testBench, niveau d'abstraction,…
La qualité d'un outil se mesure par la largeur de la toile d'araignée, puisque plus cette dernière est étendue plus l'outil HLS est meilleur, en se basant sur la figure ci-dessous il est clair que notre choix pour l'outil de synthèse haut niveau est **« vivado HLS»**



a. Vivado HLS (b) BlueSpec

(c) CatapultC (d) CyberWorkBench

Figure IV.1. Diagramme d'araignée de quelques outils HLS [32]

4.4 Synthèse haut niveau HLS

Le flot de conception de Vivado HLS est présenté dans la figure ci-dessous. Les étapes du flot de conception sont les suivantes :

1er étape : Description comportemental du modèle

Cette étape exige de suivre une méthodologie bien précise pour obtenir un modèle optimal et synthétisable. En effet il faut décrire le modèle comportementale en C, C++, ou systemC, lui associé un test bench, un programme qui appelle la fonction C à synthétiser, lui fournit des données de test et teste l'exactitude de ses sorties.

Vivado HLS prend en charge la simulation C avant la synthèse pour valider l'algorithme C et la co-simulation C / RTL après synthèse pour vérifier la mise en œuvre du RTL. Le test bech sera utilisé dans la simulation et la co- Simulation RTL.

L'une des principales techniques permettant d'assurer une bonne qualité de résultats dans le dispositif FPGA consiste à utiliser des types de données de précision arbitraire. Les types de données de précision arbitraire permettent aux variables d'être définies comme n'importe quelle largeur de bits arbitraire (6 bits, 12 bits, 143 bits, etc.). Cela permet au code C de modéliser avec précision et de synthétiser les largeurs de bits exactes requises dans le matériel.

Il est aussi important de mentionner que les appels systèmes tels que : « printf () », « exit () », « puts () » ne sont pas supportés par la synthèse.

Les objets ne peuvent pas être créés et détruits dynamiquement donc les allocations dynamiques comme « alloc () » ou « malloc () » et la récurrence ne sont pas aussi des constructions supportées par la synthèse.

Vivado HLS a fourni un certain nombre de bibliothèques C pour s'assurer que les fonctions courantes sont implémentées en tant que matériel FPGA de haute qualité. Si les bibliothèques fournies par le C/C++ sont synthétisées directement, la qualité du résultat dans le matériel FPGA sera médiocre.

2ème étape : Simulation du code C

Cette étape consiste à compiler, exécuter (simulation) et debugger le code C

3éme étape : Synthèse du modèle

Synthétiser le code C en une implémentation RTL, des rapports sont générés à cette étape et qui comportent des informations concernant l'estimation des ressources utilisées, l'estimation des performances et les interfaces de notre modèle.

Vivado HLS créera automatiquement les modules de chemin de données et de chemin de contrôle nécessaires à la mise en œuvre matérielle de notre algorithme.

L'analyse des rapports dans cette étape nous aidera à optimiser le modèle dans les prochaines étapes.

4éme étape : Optimisation du modèle

Un élément crucial de la création de conceptions RTL de haute qualité à l'aide de la synthèse de haut niveau, est la possibilité d'appliquer des optimisations au code C. La synthèse de haut niveau essaie toujours de minimiser la latence des boucles et des fonctions. Pour ce faire, dans les boucles et les fonctions, elle tente d'exécuter autant d'opérations que possible en parallèle. En plus de ces optimisations automatiques, les directives permettent :

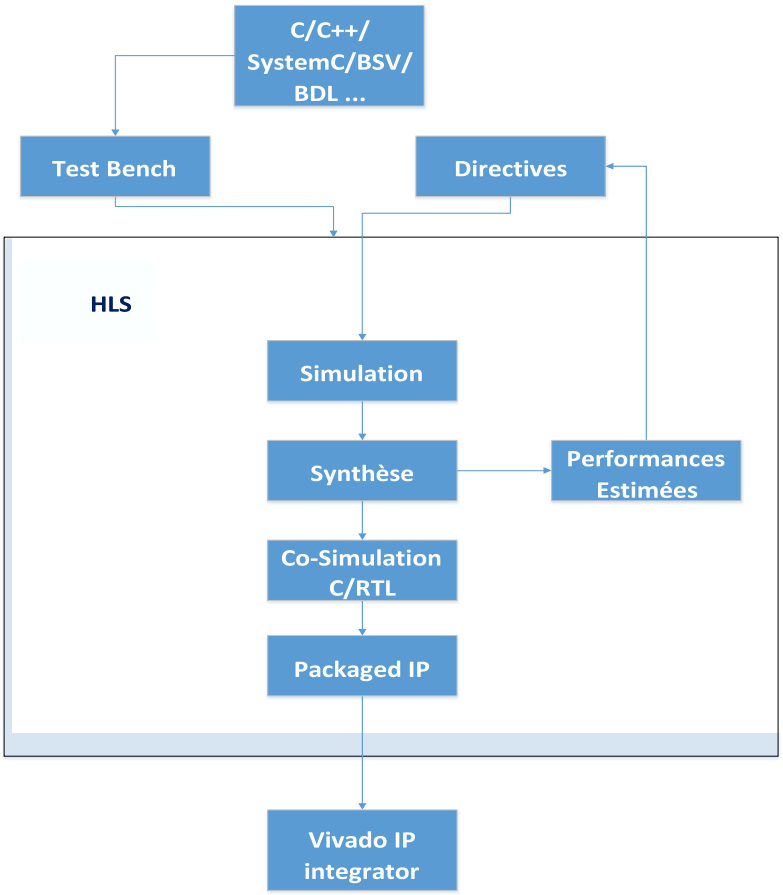
- Exécuter plusieurs tâches en parallèle, par exemple plusieurs exécutions de la même fonction ou plusieurs itérations de la même boucle. C'est le pipelining.
- Restructurer la mise en œuvre physique des tableaux (blocs RAM), des fonctions, des boucles et des ports afin d'améliorer la disponibilité des données et d'aider les données à circuler plus rapidement dans la conception.
- Fournir des informations sur les dépendances de données, ou leur absence, permettant davantage d'optimisations.

La technique d'optimisation finale consiste à modifier le code source C pour supprimer les dépendances inattendues dans le code susceptibles de limiter les performances du matériel.

5éme étape : Implémentation RTL et génération de l'IP

Lancer la co-simulation C/RTL, et empaqueter l'implémentation RTL dans une sélection de formats IP block pour pouvoir l'utiliser avec les autres outils de Vivado Design suite.

Figure IV.2. Flot de conception de Vivado HLS



4.5 Mise en œuvre du DT

4.5.1 Description comportementale de DT en C

Tout d'abord, nous avons écrit le code source C correspondant au modèle DT de la prévention du diabète. Le code consiste en une fonction appelée "DT" avec cinq variables de type « double » ,2 variables de type entier, et la variable «diabète» de type

entier, qui prend un « 0 » indiquant que le test de diabète est négatif ou un « 1 » dans le cas contraire.Le DT se présente sous la forme d'une succession d'instructions If et Else aboutissant au résultat "1" ou "0".

Le test commence à partir de la racine et continue tout au long de l'arbre, en suivant des directions précises dans les branches de l'arbre selon les valeurs des attributs, jusqu'à en arriver au niveau des feuilles ou la décision sera prise.

Un deuxième fichier C est nécessaire pour les tests dans Vivado HLS qui est le test bench également décrit en C. Le test bench prend la forme de la fonction C principale qui exécute la fonction «DT» et auto-contrôle les résultats.

4.5.2 Performances obtenues de la synthèse et l'implémentation avec et sans optimisation

- **Horloge et ressources estimées**

Tableau IV.1. Ressources Matérielles et performances estimées sur Vivado HLS

Ressources Matérielles et performances	Sans directives	Directive ''Pipline''
BRAM 18K	0	0
DSP48E	0	0
FF	1582	1582
LUT	6043	6032
Clock (ns)	8.371	7.884
Latence (cycle d'horloge)	1	1

Le récapitulatif de synchronisation indique la fréquence d'horloge cible et estimée. Si la fréquence d'horloge estimée est supérieure à la cible, le matériel ne fonctionnera pas à cette fréquence d'horloge.

- La latence est le nombre de cycles d'horloge requis pour que la fonction calcule toutes les valeurs de sortie. Il s'agit simplement du décalage entre l'application des données et le moment où elles sont prêtes. Pour la plupart des applications, cela n'est guère préoccupant, en particulier lorsque la latence de la fonction matérielle dépasse largement celle des fonctions logicielles ou système telles que DMA.

- ## Interfaces

Tableau IV.2. Ports et protocoles d'E / S créés par la synthèse d'interface

RTL Ports	Dir	Bits	Protocols	Source Object	C Type
ap_clk	In	1	ap_ctrl_hs	DT	return value
ap_rst	In	1	ap_ctrl_hs	DT	return value
ap_start	In	1	ap_ctrl_hs	DT	return value
ap_done	Out	1	ap_ctrl_hs	DT	return value
ap_idle	Out	1	ap_ctrl_hs	DT	return value
ap_ready	Out	1	ap_ctrl_hs	DT	return value
glycemie	In	64	ap_none	glycemie	scalar
BMI	In	64	ap_none	BMI	scalar
PA	In	64	ap_none	PA	scalar
DPF	In	64	ap_vld	DPF	scalar
insulin	In	64	ap_none	insulin	scalar
EP	In	64	ap_none	EP	scalar
age	In	32	ap_none	age	scalar
grossesses	In	32	ap_none	grossesses	scalar
diabete	Out	32	ap_vld	diabete	pointer
diabete_ap_vld	Out	1	ap_vld	feu	pointer

La section Interface affiche les ports et les protocoles d'E / S créés par la synthèse d'interface :

- La conception nécessite un cycle d'horloge, une horloge et une réinitialisation ont

donc été ajoutées à la conception : ap_clk et ap_rst. Les deux sont des entrées mono- bit.

- Un protocole d'E /S au niveau des blocs a été ajouté pour contrôler la conception RTL : ports ap_start, ap_done, ap_idle et ap_ready.

Tableau IV.3. Tableau explicatif de quelques signaux générés par Vivado HLS

Signal	**Description**
ap_start	Ce signal contrôle l'exécution du bloc et doit être réaffecté à la logique 1 pour que la conception puisse commencer à fonctionner. Il doit être maintenu à la logique 1 jusqu'à ce que la négociation en sortie associée ap_ready soit validée. Lorsque ap_ready devient haut, on peut décider de maintenir ap_start affirmé et d'effectuer une autre transaction ou de définir ap_start sur logique 0 et de permettre à la conception de s'arrêter à la fin de la transaction en cours. Si ap_start est déclaré bas avant qu'ap_ready soit à l'état haut, la conception n'a peut-être pas lu tous les ports d'entrée et pourrait bloquer l'opération lors de la lecture suivante de l'entrée.
ap_done	Ce signal de sortie indique que la conception est prête pour de nouvelles entrées. Le signal ap_ready est défini sur 1 lorsque la conception est prête à accepter de nouvelles entrées, indiquant que toutes les lectures d'entrées pour cette transaction sont terminées. Si la conception ne comporte aucune opération en pipeline, les nouvelles lectures ne sont pas effectuées avant le début de la transaction suivante. Ce signal est utilisé pour décider quand appliquer de nouvelles valeurs aux ports d'entrée et s'il faut commencer une nouvelle transaction en utilisant le signal d'entrée ap_start. Si le signal ap_start n'est pas déclaré haut, ce signal diminue lorsque la conception termine toutes les opérations de la transaction en cours.
ap_idl	Un 1 logique sur cette sortie indique que la conception a terminé toutes les opérations de cette transaction. Comme il s'agit de la fin de la transaction, un 1 logique sur ce signal indique également que les données du port ap_return sont valides. Toutes les fonctions n'ont pas un argument de retour de fonction et, par conséquent, toutes les conceptions RTL n'ont pas de port ap_return.

ap_ready	Ce signal indique si la conception fonctionne ou est inactive (aucune opération). L'état inactif est indiqué par la logique 1 sur ce port de sortie. Ce signal est affirmé bas lorsque la conception commence à fonctionner. Ce signal est élevé lorsque la conception est terminée et qu'aucune autre opération n'est effectuée.

La conception a quatre ports de données :

- Les ports d'entrée glycémie, BMI, EP, PA et DPF sont des entrées de 64 bits dont le protocole E / S est ap_none, et les entrées âge et grossesses sont de 32 bits, avec le même protocole.
- La conception comporte également un port de sortie 32 bits diabète.

4.6 Implémentation physique du système sur une cible SoC

4.6.1 Plateforme cible

La ZedBoard est une carte d'évaluation de chez Xilinx permettant d'utiliser la puce Zynq-7000 (composée d'un processeur ARM et d'un FPGA).

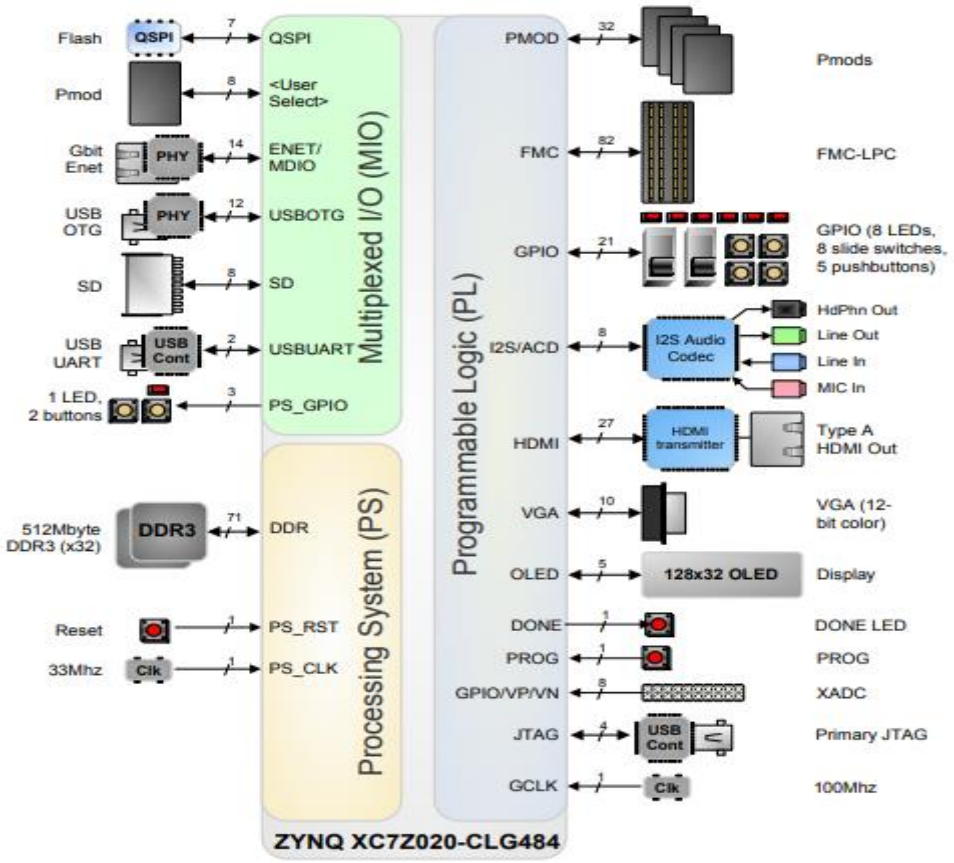


Figure IV.3. Diagramme en bloc de la ZedBoard [33]

La Zedboard est constitué de deux parties principales [33] :

- **Un système de traitement PS (processing system) :** basé sur un processeur double cœur ARM cortex A9 capable d'accueillir un système d'exploitation comme LINUX. On appelle PS la partie processeur et les périphériques associés cela inclus : les deux cœurs ARM cortex A9, les bus AMBA et AXI, le DMA, les GPIOs, I2C, UART, CAN, SPI, le contrôleur des mémoires QuadSPI, NAND et NOR et le contrôleur de mémoire vive.

- **Un système de programmation logique PL (programmable logic) :** Cette partie contient les éléments logiques de base, la RAM, des DSP et les entrées sorties standard.

Au niveau de la connexion, la ZedBoard est assez complète et permet l'interfaçage de nombreux éléments :

- HDMI (Audio et vidéo)
- VGA (Vidéo)
- Ethernet (10/100/1000 Mbps)
- Port de débogage ARM Debug Access Port (DAP)
- Port USB 2.0 OTG (Device, Host et OTG)
- Port de programmation USB-JTAG - Port USB-UART
- Connecteur FMC.

Les composants IHM sont également assez nombreux :

- 9 LEDs
- 8 interruptrices glissières
- 7 boutons poussoirs + 2 boutons Reset

Les mémoires intégrées, les entrées et sorties vidéo et audio, l'interface USB double rôle et Ethernet, ainsi que le slot SD simplifient au maximum la phase de conception et ceci sans matériel supplémentaire.

4.6.2 Flot de conception de notre solution

Le schéma ci-dessous englobe la méthodologie suivie pour la conception du système sur puce pour la prévention du diabète.

Dans les chapitres précédents, nous avons obtenu notre modèle DT à implémenter, pour cela nous allons suivre les étapes restantes dans le schéma ci-dessous

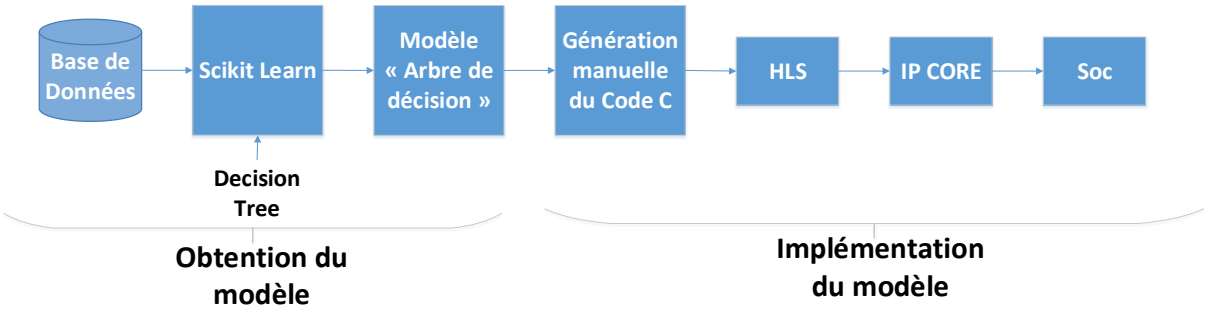


Figure IV.4. Schéma explicatif du flot de conception notre solution

4.6.2.1 L'interface AXI

Pour l'implémentation de notre DT sur une cible SoC, nous devons établir une connexion entre l'IP conçu avec HLS et le Processing System de la carte ZedBoard. Pour cela, Xilinx standardise les interconnexions IP avec le protocole AXI4. Pour créer cette interface, nous avons spécifié une interface de type AXILite pour les sept paramètres '' glycémie, âge, BMI, EP, PA, DPF et grossesses'' dans la description comportementale du DT en C.

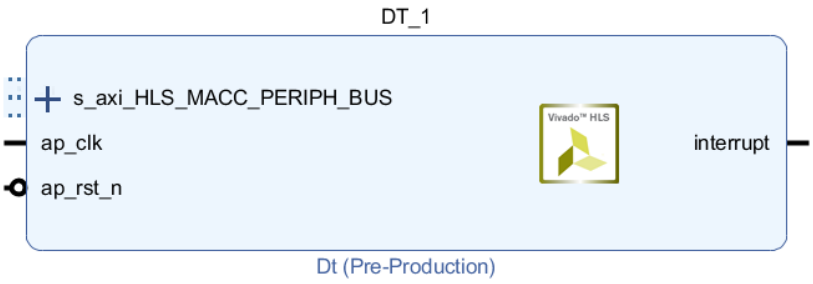


Figure IV.5. IP du DT conçu sur HLS

Les performances estimées avec Vivado HLS après l'ajout de l'interface AXI ont changé, le tableau suivant résume ces derniers.

Tableau IV.4.Ressources Matérielles et performances estimées avec l'interface AXI Lite sur Vivado HLS

Ressources Matérielles et performances	**Interface AXI Lite**
BRAM 18K	0
DSP48E	0
FF	2090
LUT	6904
Clock (ns)	7.884
Latence (cycle d'horloge)	2

AXI est une interconnexion point à point conçue pour les systèmes de microcontrôleurs hautes performances et haute vitesse et fait partie des spécifications AMBA d'ARM. Le protocole AXI est basé sur une interconnexion point à point pour éviter le partage de bus et donc permettre une bande passante supérieure et une latence plus faible. AXI est sans doute l'interconnexion d'interface AMBA la plus répandue.

L'essence du protocole AXI réside dans le fait qu'il fournit un cadre pour la communication des différents blocs à l'intérieur de chaque puce. Il propose une procédure avant toute transmission, afin que la communication soit claire et ininterrompue. [34]

La procédure pour le protocole AXI est la suivante :

- Le maître et l'esclave doivent «établir une liaison» pour confirmer les signaux valides
- La transmission du signal de contrôle doit être dans des phases séparées
- Canaux séparés pour la transmission des signaux
- Le transfert continu peut être réalisé par une communication de type rafale

Le diagramme suivant montre une interconnexion de bus AXI typique. Le processeur est connecté à la matrice d'interconnexion AXI via le bus AXI. La matrice peut prendre en charge plusieurs maîtres et plusieurs esclaves.

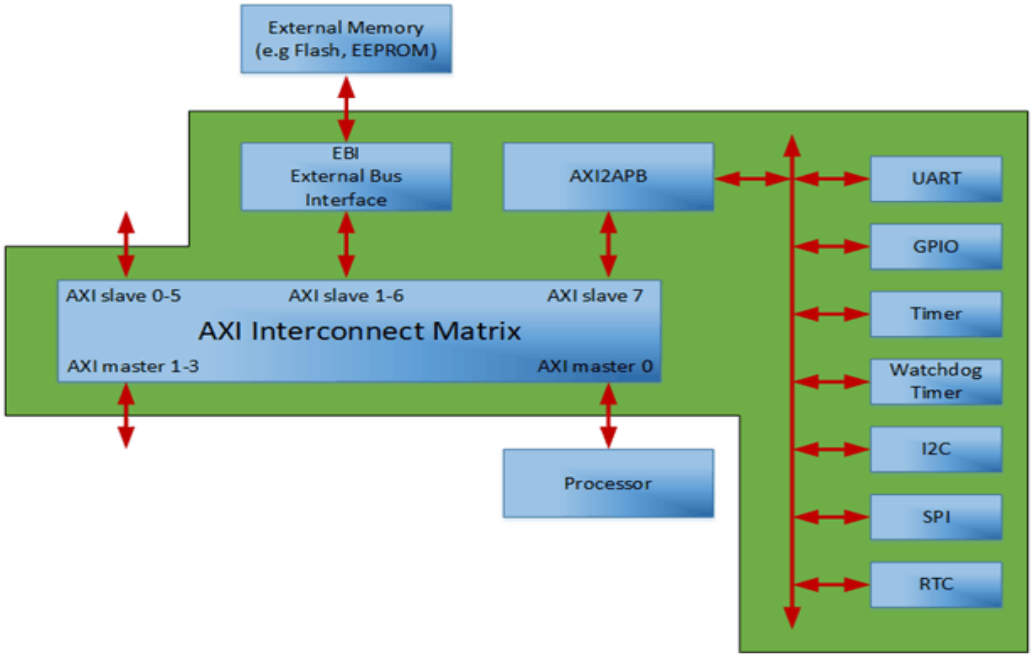


Figure IV.6. Connexion de canal entre les interfaces maître et esclave [34]

Une interconnexion AXI possède une architecture très développée, qui permet :

- D'utiliser un nombre différent de ports esclave et maître : sur VIVADO, on peut avoir jusqu'à 16 maitres et 16 esclaves sur une seul interconnexion AXI
- Conversion de largeur : Par exemple un maitre qui a une largeur de bus de données de 64 bits peut communiquer avec un esclave qui a une largeur de bus de données de 32 bits via une interconnexion AXI.
- AXI3 à AXI4 : Une interconnexion AXI peut établir une communication entre deux modules utilisant des versions d'interface AXI différentes
- Transformation de domaine d'horloge : Les deux parties d'une interconnexion AXI peuvent travailler sur des fréquences d'horloge différentes.

4.6.2.2 Conception matérielle avec Vivado IP Integrator

Une utilisation courante de la conception de synthèse de haut niveau est de créer un accélérateur pour un processeur – pour déplacer le code qui s'exécute sur le CPU dans la logique programmable FPGA afin d'améliorer les performances.

a-Mise en œuvre du système sur le Zynq SoC à l'aide de Vivado

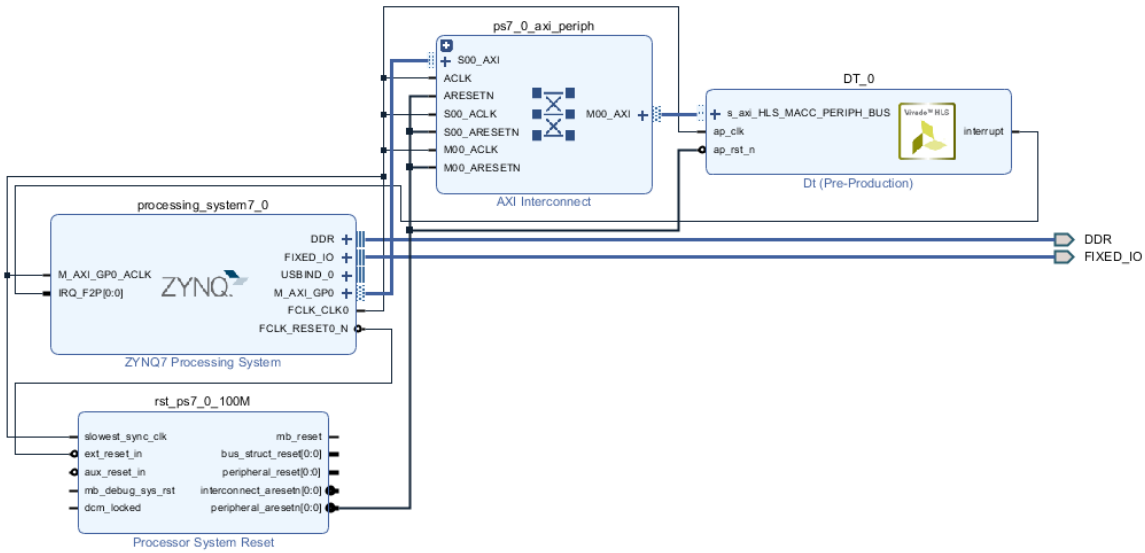


Figure IV.7. Schéma en bloc de l'implémentation sur une cible SoC

La figure ci-dessus illustre la conception de bloc matériel lors de l'utilisation de Vivado IP Integrator. Il convient de noter que lors de la création du bloc dans IP Integrator, les IP Zynq PS et HLS sont ajoutées manuellement, tandis que les deux IP supplémentaires sont automatiquement ajoutés à la conception ; (Processor System Reset et AXI Interconnect) lors de l'exécution du bloc et de l'automatisation de la connexion. Toutes les interconnexions entre les différents blocs sont également réalisées.

- Processing System ZYNQ7 (Processing_system7_0)

  Le processing system est ajouté manuellement, il sert uniquement à la configuration et ne sera pas implémenté sur le PL, il correspond en fait au PS fixe sur la puce Zynq, tout ce qui concerne le PS doit être spécifié dans cet IP.

- DT_0

  C'est l'IP qui a été conçu, simulé et exporté au format HLS.

- AXI Interconnect (ps7_0_axi_periph)

  Il est utilisé pour interconnecter un périphérique maître AXI mappé en mémoire, dont le PS, avec un périphérique esclave mappé en mémoire, qui est le cœur HLS compatible AXI-Lite appelé «DT» dans cette solution.

- Processor System reset (rst_ps7_0_100M)

Il permet d'adapter la conception à son application en définissant certains paramètres pour activer / désactiver les fonctions.

b- Synthèse, implémentation du Soc et génération du Bitstream

La génération de flux binaire crée l'image matérielle, qui peut être exportée vers un environnement logiciel dans lequel nous allons créer une application logicielle utilisant le custom hardware.

4.6.2.3 Conception logicielle avec SDK

Le kit de développement logiciel (SDK) fournit un environnement permettant de créer des plates-formes logicielles et des applications destinées aux processeurs intégrés Xilinx. Le SDK fonctionne avec les conceptions matérielles créées avec Vivado.

Dans cet environnement nous avons suivis les étapes suivantes pour vérifier le bon fonctionnement de notre IP généré par HLS :

1. Nous avons créé une application software écrite en code C
- Utiliser les fonctions disponibles sur le driver de notre IP (xdt.c) fournit par HLS, principalement les fonctions dédiées à l'insertion des trois variables (glycémie, âge, BMI, EP, PA, DPF et grossesses), et la fonction pour récupérer la variable (diabète) indiquant le résultat fournie par notre IP.
    - Définir un modèle logiciel de la fonctionnalité matérielle HLS (code C de DT) avec lequel nous pouvons comparer les résultats de référence.
    - Ecrire une fonction main, qui fait appel à la fonction hardware (IP fournit par HLS) et la fonction software (code C de DT), et qui fait une comparaison entre les résultats des deux fonctions et envoi une réponse ''Results match'', si les deux fonctions donnent le même résultat et ''Results Mismatch'' dans le cas contraire.

2. Nous avons fait le branchement nécessaire entre la ZedBoard et l'ordinateur.

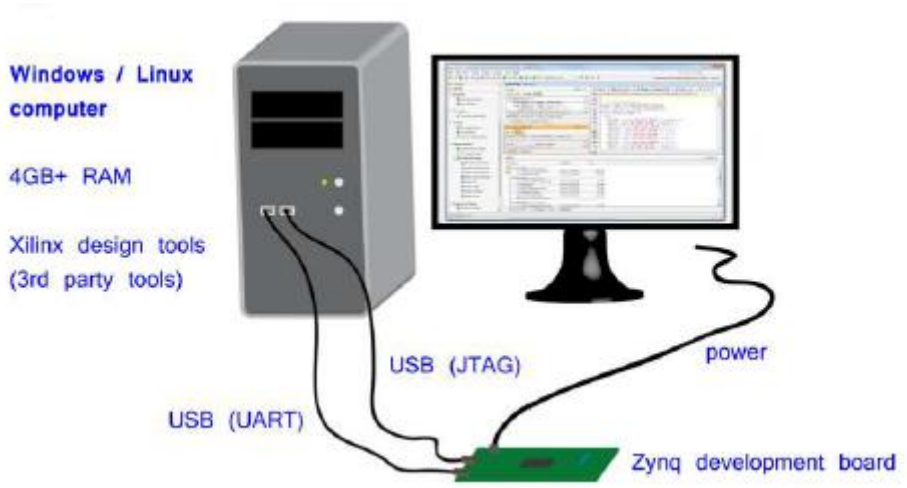


Figure IV.8. Dispositif du test

3. Nous avons utilisé le BitStream générer par notre system SoC à partir de l'outil IP Integrator comme plateforme Hardware.
4. Enfin, nous avons chargé le code sur la carte ZedBoard, et observer le résultat sur le terminal de SDK.

4.7 Conclusion

Nous avons abordé dans ce chapitre le flot de conception de la synthèse haut niveau de notre DT ou nous avons obtenue l'IP core à partir de la description en C de notre arbre de que nous avons implémenté sur une cible SoC, nous avons par la suite générer le flux binaire que nous avons importé sur l'environnement logiciel SDK et sur lequel nous avons vérifié le bon fonctionnement de notre IP.

Conclusion générale

Dans cette présente étude, nous nous sommes intéressés à l'implémentation d'un système sur puce dédié à la prévention du diabète.

Cette implémentation nous l'avions réfléchie à travers une méthodologie de développement facilitant le prototypage rapide. C'est ainsi qu'une étude comparative des diverses plateformes mettant en œuvre des algorithmes de ML nous a conduit à déterminer et la meilleure plateforme, et le meilleur algorithme qui devra être appliqué pour la prévention du diabète.

Comme on l'avait démontré, le couple Scikit Learn et le DT était le choix pertinent quant à l'obtention du modèle ML. La meilleure précision (88.31%) a été obtenue pour l'algorithme DT (profondeur de 5) utilisant CART comme critère de division, comparativement à d'autres DT avec différentes profondeurs et à d'autres algorithmes de ML tel que Bagged Tree, SVM et KNN.

Pour l'implémentation de notre modèle, nous avons opté pour l'approche HLS. Ce choix a été déduit de travaux comparatifs sur les approches d'implémentation ; le critère a été d''abord le temps de prototypage. D'autres études ont montré que Vivado HLS est le meilleur outil de synthèse haut niveau tenant compte de plusieurs critères tels que la facilité d'implémentation, la possibilité d'optimisation du modèle et disponibilité des ressources documentaire.

Le DT obtenu a été implémenté en hardware en utilisant une plateforme FPGA de type ZedBoard. Nous avons vérifié le bon fonctionnement de l'IP obtenu par la comparaison des résultats obtenus avec ceux obtenus par une implémentation purement soft.

Perspectives

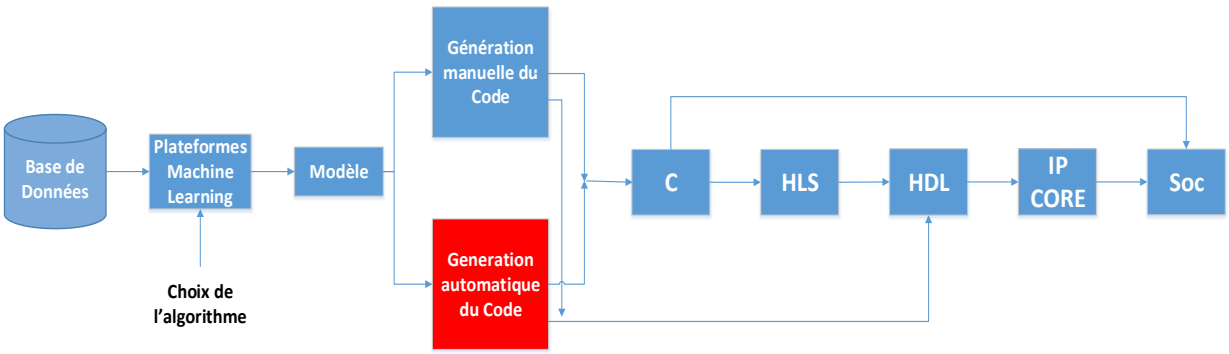
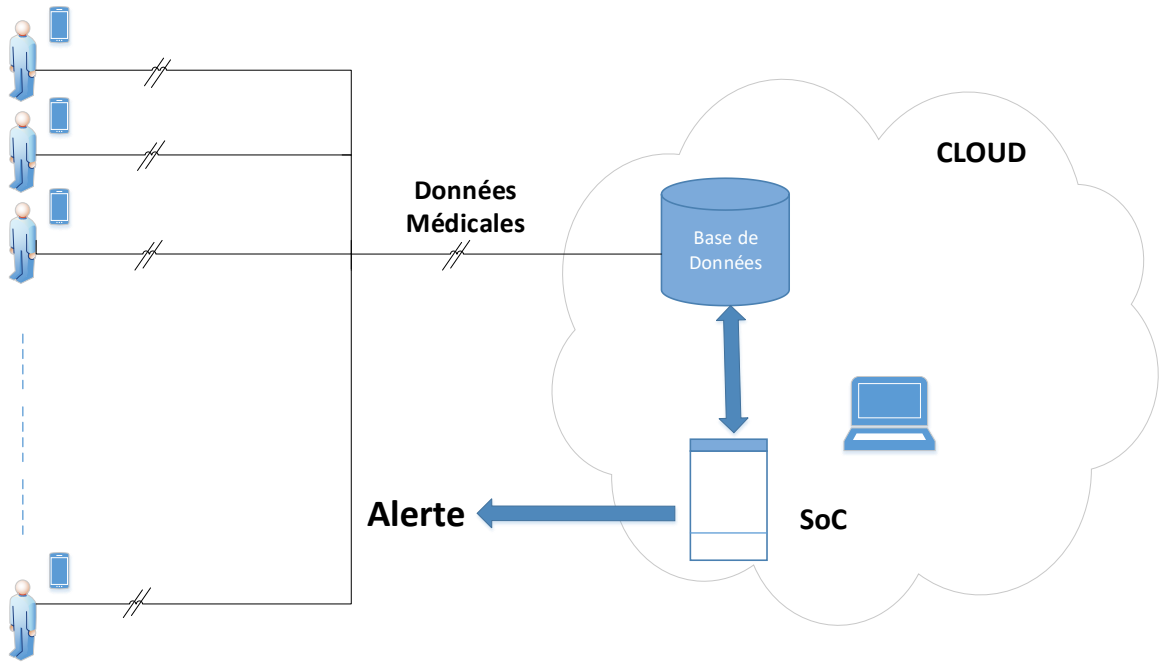


Figure IV.6. Amélioration de notre méthodologie d'implémentation

Dans notre approche de développement, le passage du modèle d'apprentissage vers son implémentation est subordonné à une traduction manuelle. Cette limitation est toute naturelle car les plateformes ML sont dédiées aux « Data Scientiste ». Nous proposons, comme perspective, le développement d'un traducteur automatique vers des langages de description ou (et) vers des langages impératifs. Ces derniers peuvent exploiter le HLS pour une approche prototypage rapide.



Comme aujourd'hui, de nombreux types de systèmes embarqués peuvent s'interfacer sans effort raisonnable avec les systèmes Cloud pour l'échange de données, le stockage de données et le

traitement de ces dernières, nous proposons comme autre perspectives pour notre projet, une connexion du SoC que nous avons développé avec le Cloud. L'utilisation des accélérateurs a niveau du Cloud est un potentiel qui peut dépasser les limites technique du Cloud, comme le transfert des fonctions d'un système temps réel vers le Cloud qui impose un certain nombre de contraintes en matière de synchronisation, telles que la communication en temps réel, la durée d'exécution des boucles fermées et la gigue, qui pourraient être difficiles à atteindre aujourd'hui.

Le schéma suivant illustre notre proposition qui s'inscrit dans la continuité de notre projet, on peut imaginer des milliers de personnes qui communiquent leurs données médicales via une application mobile au Cloud. Notre solution SoC constitue l'unité de traitement de ce système de prévention du diabète, ou les données collectées sur le Cloud sont envoyées au SoC, pour décider si une personne avec ses données prédictives médicales, est à risque d'avoir un diabète ou non. Cette information est ensuite envoyée à la personne.

L'intérêt d'un tel système est le traitement des données en temps réels, ainsi que le nombre de personnes pouvant bénéficier de cette application qui se chiffre en million.

BIBLIOGRAPHIE

[1] World Health Statistics 2014. Genève, Organisation Mondiale de la Santé, 2014.

[2] Équipe de professionnels de la santé de Diabète Québec Mai 2014.

[3] De Kerdanet M. Éditorial. Diabète de type 1 de l'enfant : des chiffres et des pistes. Bull Epidémiol Hebd. 2017;(27- 28):570-1. http://invs.santepubliquefrance.fr/beh/2017/27-28/2017_27-28_0.html.

[4] Communauté de top santé.Top santé.. [en ligne]. [consulté le 19 juin 2019]. Disponible sur <https://www.topsante.com/themes/diabete-gestationnel>

[5] Dr Marc Adasy, le diabète. CHU de limoges, France. 2016.

[6] 4 ELEVES DE 1ERE S DU LYCEE MARCELIN BERTHELOT. Le diabète. [en ligne]. [consulté le 19 juin 2019]. Disponible sur <http://tpe-lediabete.e-monsite.com/pages/la-place-du-diabete-de-nos-jours/ >

[7] INTERNATIONAL DIABETE FEDIRATION. IDF DIABETES ATLAS - 8TH EDITION. [en ligne]. [consulté le 12 juin 2019]. Disponible sur <https://www.diabete.qc.ca/>

[8] Diabetes Atlas IDF 8e Edition 2017.

[9]AGENCE ARCHIPEL. DIABETE, Québec. [en ligne]. [consulté le 12 juin 2019]. Disponible sur < https://diabetesatlas.org/>

[10] Projet TANINA, INSP.

[11] Willi et al. 2007: Willi C. Bodenmann P. Ghali WA. Faris PD. Cornuz J. (2007). Active smoking and the risk of type 2 diabetes; a systematic review and meta-analysis. Journal of the American Medical Association Volume 298, pages 2654-2664

[12] American Diabetes Association

[13] Contreras, I., & Vehi, J. (2018). Artificial intelligence for diabetes management and decision support: literature review. *Journal of medical Internet research*, *20*(5), e10775.

[14] DELIOTTE. Hitting the accelerator: the next generation machine-leraning chips. [en ligne]. [consulté le 12 juin 2019]. Disponible sur <https://www2.deloitte.com/content/dam/Deloitte/global/Images/infographics/technologymediatelecommunications/gx-deloitte-tmt-2018-nextgen-machine-learning-report.pdf>

[15] WIKIPEDIA. Apprentissage Automatique. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://fr.wikipedia.org/wiki/Apprentissage_automatique >

[16] WIKIPEDIA. Machine à vecteurs support. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://fr.wikipedia.org/wiki/Machine_%C3%A0_vecteurs_de_support>

[17] WIKIPEDIA. Méthode des K plus proches voisins. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://fr.wikipedia.org/wiki/M%C3%A9thode_des_k_plus_proches_voisins>

[18] WIKIPEDIA. Arbre de décision. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://fr.wikipedia.org/wiki/Arbre_de_d%C3%A9cision_(apprentissage)#M%C3%A9thodes_d'ensembles>

[19] Song, Y., Liang, J., Lu, J., & Zhao, X. (2017). An efficient instance selection algorithm for k nearest neighbor regression. *Neurocomputing*, *251*, 26-34

20] Pradeep, K. R., & Naveen, N. C. (2016, December). Predictive analysis of diabetes using J48 algorithm of classification techniques. In *2016 2nd International Conference on Contemporary Computing and Informatics (IC3I)* (pp. 347-352). IEEE.

[21] Meng, X. H., Huang, Y. X., Rao, D. P., Zhang, Q., & Liu, Q. (2013). Comparison of three data mining models for predicting diabetes or prediabetes by risk factors. *The Kaohsiung journal of medical sciences*, *29*(2), 93-99.

[22] Saravananathan, K., & Velmurugan, T. (2016). Analyzing Diabetic Data using Classification Algorithms in Data Mining. *Indian Journal of Science and Technology*, *9*(43), 1-6.

[23] SCIKIT LEARN DEVELOPERS. Scikit Learn. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html#sphx-glr-auto-examples-model-selection-plot-roc-py>

[24] THE MATHWORKS, INC. MATLAB for Artificial Intelligent. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur< https://www.mathworks.com/help/stats/classificationlearner-app.html>

[25] Wikipédia. Weka(informatique). [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://fr.wikipedia.org/wiki/Weka_(informatique)>

[26] WIKIPEDIA. Scikit Learn. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur< https://fr.wikipedia.org/wiki/Scikit-learn>

[27] WIKIPEDIA. Decision tree learning. [en ligne]. [consulté le 26 janvier 2019]. Disponible sur <https://en.wikipedia.org/wiki/Decision_tree_learning#Gini_impurity>

[28] Roy, A., Qureshi, S., Pande, K., Nair, D., Gairola, K., Jain, P., ... & Sharma, S. (2019). Performance Comparison of MLPlatforms. *INFORMS Journal on Computing*.

[29] Roy, A., Qureshi, S., Pande, K., Nair, D., Gairola, K., Jain, P., ... & Sharma, S. (2019). Performance Comparison of MLPlatforms. *INFORMS Journal on Computing*, pages 12,13.

[30] Roy, A., Qureshi, S., Pande, K., Nair, D., Gairola, K., Jain, P., ... & Sharma, S. (2019). Performance Comparison of MLPlatforms. *INFORMS Journal on Computing*, pages 13,14.

[31] Maxime Pelcat, Cédric Bourrasset, Luca Maggiani, François Berry ‘’ Design Productivity of a High Level Synthesis Compiler versus HDL’’, 2016

[32] Wim Meeus, Kristof Van Beeck, Toon Goedemé, Jan Meel, Dirk Stroobandt “An overview of today’s high-level synthesis tools”, 2012.

[33] AVENET. ZEDBoard. [en ligne]. [consulté le 02 mars 2019]. Disponible sur < http://zedboard.org/>

[34] ANYSILICON. The Google of the semiconductor industry. [en ligne]. [consulté le 29 mars 2019]. Disponible sur <https://anysilicon.com/understanding-axi-protocol-quick-introduction/>

Annexes

Code sur Scikit Learn

```
#import the necessary packages
import numpy as np
import pandas as pd
import csv as df
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score
from sklearn import tree
from subprocess import call
from sklearn.tree import export_graphviz
from IPython.display import Image
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeClassifier
from sklearn import tree
import graphviz
from IPython.display import SVG
from graphviz import Source
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import AdaBoostClassifier
from sklearn.metrics import recall_score
from sklearn.tree._tree import TREE_LEAF
#Loading data file
dataset =pd.read_csv("C:/Users/7 Pro/Desktop/PFE/Base-de-données-
Diabète/Final/diabetes_8.csv",engine='python')
print("Dataset Lenght:: ",len(dataset))
print("Dataset Shape:: ",len(dataset.shape))
print("Dataser:: ")
#afficher les 5 première lignes des données
dataset.head()
#Separating the Target variable
```

```
X=dataset.values[:,0:8]

Y=dataset.values[:,8]

#Spliting dataset into test and train

X_train, X_test, Y_train, Y_test=train_test_split( X, Y,
test_size=0.2,random_state=15923)

#Function to perform training with entrpy

model= DecisionTreeClassifier(criterion = "gini", random_state=
14,max_depth=5,min_samples_leaf=5)

model.fit(X_train, Y_train)

#Function to perform training with entrpy

model= DecisionTreeClassifier(criterion = "gini", random_state=
14,max_depth=5,min_samples_leaf=5)

model.fit(X_train, Y_train)

#cheking Accuracy

print( "Accuracy is ",accuracy_score(Y_test,Y_pred_en)*100)

print ("Recall",recall_score(Y_test,Y_pred_en, average='macro'))

from sklearn.tree._tree import TREE_LEAF


def is_leaf(inner_tree, index):

    # Check whether node is leaf node

    return (inner_tree.children_left[index] == TREE_LEAF and

            inner_tree.children_right[index] == TREE_LEAF)


def prune_index(inner_tree, decisions, index=0):

    if not is_leaf(inner_tree, inner_tree.children_left[index]):

        prune_index(inner_tree, decisions,
inner_tree.children_left[index])

    if not is_leaf(inner_tree, inner_tree.children_right[index]):

        prune_index(inner_tree, decisions,
inner_tree.children_right[index])


    # Prune children if both children are leaves now and make the same
decision:

    if (is_leaf(inner_tree, inner_tree.children_left[index]) and

        is_leaf(inner_tree, inner_tree.children_right[index]) and

        (decisions[index] == decisions[inner_tree.children_left[index]])
and

        (decisions[index] ==
decisions[inner_tree.children_right[index]])):

        # turn node into a leaf by "unlinking" its children
```

```
        inner_tree.children_left[index] = TREE_LEAF
        inner_tree.children_right[index] = TREE_LEAF
        ##print("Pruned {}".format(index))

 def prune_duplicate_leaves(mdl):
 decisions = mdl.tree_.value.argmax(axis=2).flatten().tolist() # Decision
for each node
    prune_index(mdl.tree_, decisions)
#visulize a tree
prune_duplicate_leaves(model)
dot_file='C:/Users/7 Pro/Desktop/PFE/TEST-notebook/Best-algo-algerie/tree3.dot'
png_file='C:/Users/7 Pro/Desktop/PFE/TEST-notebook/best-algorithm/tree3.png'
dot_data = tree.export_graphviz(model, out_file=None,feature_names
=['Grossesses','Glucose','Pression Artérielle','Epaisseur de la
peau','Insulin','BMI','Diabetes-Pedigree-Function','Age'],
 class_names= ['non','oui'], rounded = True, proportion = True, precision
=1, filled = True)
graph=graphviz.Source(dot_data)
graph
```

Code C du DT

```
#include "DT.h"



void DT(double glycemie, int age ,double BMI,double EP,double PA, double DP
F, int grossesses, int *diabete)
{

#pragma HLS INTERFACE s_axilite port=return bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=glycemie bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=age bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=BMI bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=EP bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=DPF bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=PA bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=grossesses bundle=HLS_MACC_PERIPH_BUS
#pragma HLS INTERFACE s_axilite port=diabete bundle=HLS_MACC_PERIPH_BUS


   static int acc_reg = 0;

  if (glycemie<= 127.5)
    {if (age <= 28)
      {if (BMI <=30.9) acc_reg=0;
        else
```

```
          {if (PA <= 51) acc_reg=1;
            else acc_reg=0;
          }
        }
        else
          {if (glycemie <=99.5)
            {if (age <=42.5) acc_reg=0;
              else
                {if (EP <=24) acc_reg=0;
                  else acc_reg=1;
                }
            }
            else
              {if (BMI <=26.4) acc_reg=0;
                else
                  {if (DPF <=0.2) acc_reg=0;
                    else acc_reg=1;
                  }
                }
              }
            else
              {if (BMI <=29.9)
                {if (glycemie <=151.5) acc_reg=0;
                  else
                    {if (grossesses<= 2.5) acc_reg=0;
                      else
                        {if (BMI <=25) acc_reg=1;
                          else acc_reg=0;
                        }
                      }
                    else
                      {if (glycemie <=165.5)
                        {if (PA<=61) acc_reg=1;
                          else
                            {if (age <=30.5) acc_reg=0;
                              else acc_reg=1;
                            }
                          }
                          else acc_reg=1;
                        }
                      }


   *diabete = acc_reg;
}
```

Test Bench

```
#include "DT.h"

int main()
{

  double glycemie[5]={100,130,160,160,160};
  int age[5]={25,25,25,25,25};
  double BMI[5]={21,20,20,20,28};
  double EP[5]={1,1,1,1,1};
  double PA[5]={142,142,142,142,142};
```

```
  double DPF[5]={0.2,0.2,0.2,0.2,0.2};
  int grossesses[5]={0,0,1,4,4};
  int expect[5]={0,0,0,1,0};
  int rslt[5];
  int sum[5];
  int k=0;

  for(int i=0;i<5;i++)

    {
    DT(glycemie[i], age[i] , BMI[i] , EP[i] , PA[i] , DPF[i] , grossesses[i
] , &rslt[i]);

    if(rslt[i]==expect[i]) sum[i]=0; else sum[i]=1;

    k=k+sum[i];
    }
  if(k==0) return 0;

  else return 1;
}
```

Code sur le SDK

```
#include <stdio.h>
#include "XDt.h"   // Device driver for HLS HW block
// Add BSP header files
#include <stdlib.h>  // Standard C functions, e.g. exit()
#include <stdbool.h> // Provides a Boolean data type for ANSI/ISO-C
#include "xparameters.h" // Parameter definitions for processor periperals
#include "xscugic.h"     // Processor interrupt controller device driver
#include "inttypes.h"
// HLS macc HW instance
XDt DtMacc;
//Interrupt Controller Instance
XScuGic ScuGic;
// Global variable definitions - used by ISR
volatile static int RunHlsMacc = 0;
volatile static int ResultAvailHlsMacc = 0;
// Setup and helper functions
int setup_interrupt();
int XDt_init(XDt *hls_maccPtr);
```

```
void hls_macc_start(void *InstancePtr);

// The ISR prototype

void hls_macc_isr(void *InstancePtr);

// Software model of HLS hardware

void sw_macc(double glycemie, int age ,double BMI,double EP,double PA,
double DPF, int grossesses, int *diabete);

int main()

{

   print("Program to test communication with HLS MACC block in PL\n\r");

   double glycemie= 121, BMI=28.4, EP= 26, PA= 50,  DPF=0.5 ;

   int age = 35,grossesses = 0;

   int res_hw;

   int res_sw;

   int status;

   //Setup the matrix mult

   status = XDt_init(&DtMacc);

   if(status != XST_SUCCESS){

      print("HLS peripheral setup failed\n\r");

      exit(-1);


   //Setup the interrupt

   status = setup_interrupt();

   if(status != XST_SUCCESS){

      print("Interrupt setup failed\n\r");

      exit(-1);


   //set the input parameters of the HLS block

   XDt_Set_glycemie(&DtMacc,*((u64*)&glycemie));

   XDt_Set_BMI(&DtMacc, *((u64*)&BMI));

   XDt_Set_EP(&DtMacc,*((u64*)& EP));

   XDt_Set_PA(&DtMacc,*((u64*)&PA));

   XDt_Set_DPF(&DtMacc, *((u64*)&DPF));

   XDt_Set_age(&DtMacc, age);

   XDt_Set_grossesses(&DtMacc,grossesses);

   if (XDt_IsReady(&DtMacc))

      print("HLS peripheral is ready.  Starting... ");

   else {
```

```
        print("!!! HLS peripheral is not ready! Exiting...\n\r");
        exit(-1);

    if (0) { // use interrupt
        XDt_Start(&DtMacc);
        while(!ResultAvailHlsMacc)
            ; // spin
        res_hw = XDt_Get_diabete(&DtMacc);
        print("Interrupt received from HLS HW.\n\r");
    } else { // Simple non-interrupt driven test
        XDt_Start(&DtMacc);
        do {
            res_hw = XDt_Get_diabete(&DtMacc);
        } while (!XDt_IsReady(&DtMacc));
        print("Detected HLS peripheral complete. Result received.\n\r");

    //call the software version of the function
    sw_macc(glycemie, age , BMI, EP, PA, DPF, grossesses, &res_sw);
    printf("Result from HW: %d; Result from SW: %d\n\r", res_hw, res_sw);
    if (res_hw == res_sw) {
        print("*** Results match ***\n\r");
        status = 0;

    else {
        print("!!! MISMATCH !!!\n\r");
        status = -1;

   // cleanup_platform();
    return status;
}
void sw_macc(double glycemie, int age ,double BMI,double EP,double PA,
double DPF, int grossesses, int *diabete)
{
    static int acc_reg = 0;
    if (glycemie<= 127.5)
        {if (age <= 28)
            {if (BMI <=30.9) acc_reg=0;
```

```
else
   {if (PA <= 51) acc_reg=1;
      else acc_reg=0;


else
   {if (glycemie <=99.5)
      {if (age <=42.5) acc_reg=0;
         else
            {if (EP <=24) acc_reg=0;
               else acc_reg=1;


      else
         {if (BMI <=26.4) acc_reg=0;
            else
               {if (DPF <=0.2) acc_reg=0;
                  else acc_reg=1;




      else
         {if (BMI <=29.9)
            {if (glycemie <=151.5) acc_reg=0;
               else
                  {if (grossesses<= 2.5) acc_reg=0;
                     else
                        {if (BMI <=25) acc_reg=1;
                           else acc_reg=0;



                  else
                     {if (glycemie <=165.5)
                        {if (PA<=61) acc_reg=1;
```

```
                                        else
                                           {if (age <=30.5) acc_reg=0;
                                              else acc_reg=1;


                                        else acc_reg=1;


   *diabete = acc_reg;
}
int XDt_init(XDt *hls_maccPtr)
{
   XDt_Config *cfgPtr;
   int status;
   cfgPtr = XDt_LookupConfig(XPAR_XDT_0_DEVICE_ID);
   if (!cfgPtr) {
      print("ERROR: Lookup of acclerator configuration failed.\n\r");
      return XST_FAILURE;

   status = XDt_CfgInitialize(hls_maccPtr, cfgPtr);
   if (status != XST_SUCCESS) {
      print("ERROR: Could not initialize accelerator.\n\r");
      return XST_FAILURE;

   return status;
}
void hls_macc_start(void *InstancePtr){
   XDt *pAccelerator = (XDt *)InstancePtr;
   XDt_InterruptEnable(pAccelerator,1);
   XDt_InterruptGlobalEnable(pAccelerator);
   XDt_Start(pAccelerator);
}
void hls_macc_isr(void *InstancePtr){
   XDt *pAccelerator = (XDt *)InstancePtr;
   //Disable the global interrupt
   XDt_InterruptGlobalDisable(pAccelerator);
```

```
    //Disable the local interrupt
    XDt_InterruptDisable(pAccelerator,0xffffffff);
    // clear the local interrupt
    XDt_InterruptClear(pAccelerator,1);
    ResultAvailHlsMacc = 1;
    // restart the core if it should run again
    if(RunHlsMacc){
       hls_macc_start(pAccelerator);


}
int setup_interrupt()
{
    //This functions sets up the interrupt on the ARM
    int result;
    XScuGic_Config *pCfg =
XScuGic_LookupConfig(XPAR_SCUGIC_SINGLE_DEVICE_ID);
    if (pCfg == NULL){
       print("Interrupt Configuration Lookup Failed\n\r");
       return XST_FAILURE;


    result = XScuGic_CfgInitialize(&ScuGic,pCfg,pCfg->CpuBaseAddress);
    if(result != XST_SUCCESS){
       return result;


    // self test
    result = XScuGic_SelfTest(&ScuGic);
    if(result != XST_SUCCESS){
       return result;


    // Initialize the exception handler
    Xil_ExceptionInit();
    // Register the exception handler
    //print("Register the exception handler\n\r");
    Xil_ExceptionRegisterHandler(XIL_EXCEPTION_ID_INT,(Xil_ExceptionHandler
)XScuGic_InterruptHandler,&ScuGic);
    //Enable the exception handler
    Xil_ExceptionEnable();
```

```
    // Connect the Adder ISR to the exception table
    //print("Connect the Adder ISR to the Exception handler table\n\r");
    result =
XScuGic_Connect(&ScuGic,XPAR_FABRIC_DT_0_INTERRUPT_INTR,(Xil_InterruptHand
ler)hls_macc_isr,&DtMacc);
       if(result != XST_SUCCESS){
          return result;

       //print("Enable the Adder ISR\n\r");
          XScuGic_Enable(&ScuGic,XPAR_FABRIC_DT_0_INTERRUPT_INTR);
                   return XST_SUCCESS;
}
```