Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

AIDE A LA PREVENTION DE L'ACCIDENT VASCULAIRE CEREBRAL PAR LES TECHNIQUES D'ANALYSE PREDICTIVE ET D'INTELLIGENCE ARTIFICIELLE

[1]Assistant Sylvain-Mozart NGANDU KANUMAYI, [2]Assistant Hippolite KABANGO MBUILA, [3]Professeur Boni KIBAMBE MUTAMBA, [4]Benie PATIENCE MUNKANDI

[1]Faculté des Sciences économique et des gestion, département des sciences commerciales administrative et de Gestion Informatique, Université pédagogique nationale (UPN), République démocratique du Congo

[2]Faculté des sciences et technologies, Département de mathématiques, de statistique et d'informatique, Université pédagogique nationale (UPN), République démocratique du Congo

[3]Faculté des Sciences économique et des gestion, département des sciences commerciales administrative et de Gestion Informatique, Université pédagogique nationale (UPN), République démocratique du Congo

[4]Faculté des sciences et technologies, Département de mathématiques, de statistique et d'informatique, Université pédagogique nationale (UPN), République démocratique du Congo

DOI: https://doi.org/10.5281/zenodo.18379420

Published Date: 28-January-2026

Résumé: **Cet article traite de l'application des méthodes d'intelligence artificielle, en particulier l'apprentissage automatique, pour prédire le risque d'AVC, dans le but de prévenir ce problème avant qu'il ne survienne.**

L'AVC reste un problème sérieux pour la santé publique, qui cause beaucoup de morts et des handicaps permanents, même si beaucoup de cas pourraient être évités grâce à une détection tôt des risques. L'étude utilise un ensemble de données déjà existant provenant de la plateforme Kaggle, qui inclut 5229 personnes adultes. Ces données comportent des informations sur divers aspects de leur vie, notamment leur âge, leur sexe, s'ils souffrent d'hypertension ou de diabète, leur poids corporel, et s'ils fument. Après avoir préparé et étudié les données, différents algorithmes de classification ont été essayés, comme la Random Forest, l'arbre de décision, le Support Vector Machine et la régression logistique. Pour corriger l'incidence inégale des classes dans les données, la méthode SMOTE a été utilisée. Les résultats indiquent que les modèles utilisant les arbres de décision, notamment la Random Forest, donnent les meilleurs résultats en ce qui concerne la précision et leur capacité à prédire le risque d'AVC. En résumé, cette recherche montre que les méthodes d'apprentissage automatique sont un bon outil pour aider les professionnels de santé dans leurs décisions et peuvent jouer un rôle important dans la prévention des accidents vasculaires cérébraux.

Mots-clés: **Accident vasculaire cérébral, intelligence artificielle, apprentissage automatique, prévention primaire, analyse prédictive.**

1. INTRODUCTION

L'évolution rapide des technologies numériques a entraîné une production massive de données dans presque tous les domaines d'activité (Mayer-Schönberger, 2013).[1] Cette explosion des données a favorisé l'émergence de nouvelles disciplines capables d'exploiter efficacement ces informations, notamment l'intelligence artificielle (IA) (Haiech, 2020). [2]

Aujourd'hui, l'IA occupe une place centrale dans plusieurs secteurs, dont celui de la santé, où elle contribue à l'amélioration du diagnostic, du suivi des patients et surtout de la prévention des maladies.

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

Parmi les pathologies constituant un enjeu majeur de santé publique figure l'Accident Vasculaire Cérébral (AVC) (Bejot, 2007). [3] Malgré les avancées médicales, le nombre de personnes victimes d'AVC continue d'augmenter (Feigin, 2021), [4] alors même qu'une grande proportion de ces accidents pourrait être évitée grâce à une prévention efficace (O'Donnell, 2016). [5]

Dans ce contexte, l'utilisation des techniques d'analyse prédictive et d'intelligence artificielle apparaît comme une piste prometteuse pour anticiper les risques et renforcer la prévention primaire (Obermeyer, 2016). [6]

1.1 Théories

Les théories de la médecine préventive mettent en évidence l'importance de l'identification précoce des facteurs de risque afin d'éviter l'apparition des maladies. (OMS, 2005)[7]

Parallèlement, les théories de l'intelligence artificielle, notamment celles liées à l'apprentissage automatique, reposent sur la capacité des machines à apprendre à partir de données pour produire des prédictions fiables.

En santé, l'analyse prédictive permet d'exploiter des données cliniques et comportementales afin d'anticiper la survenue de pathologies, ouvrant ainsi la voie à une prévention personnalisée et proactive.

1.2 Situations désirées

La situation idéale serait de disposer d'un outil intelligent capable :

- d'identifier précocement les individus à risque d'AVC ;
- d'évaluer de manière fiable le niveau de risque à partir des facteurs individuels ;
- d'aider les professionnels de santé à mettre en place des mesures préventives adaptées.

Une telle solution contribuerait à réduire l'incidence des AVC, la mortalité associée ainsi que les handicaps permanents qui en découlent.

1.3 Situation vécue

Dans la réalité, les AVC continuent de représenter une cause majeure de mortalité et de handicap à travers le monde (Feigin, Global burden of stroke, 2021). [8]

La prévention reste souvent insuffisante, en grande partie à cause (Beaglehole, 2011) [9]:

- d'un dépistage tardif des facteurs de risque ;
- d'une mauvaise hygiène de vie ;
- d'un manque d'outils prédictifs accessibles et performants.

Malgré l'existence d'outils d'aide au diagnostic, la prévention primaire demeure encore peu exploitée de manière systématique.

1.4 Problématique (questions de recherche)

Face à cette situation, les questions suivantes se posent :

- Les techniques d'intelligence artificielle peuvent-elles contribuer efficacement à la prévention primaire de l'Accident Vasculaire Cérébral ?
- Dans quelle mesure un modèle basé sur l'apprentissage automatique peut-il prédire le risque d'AVC à partir des facteurs de risque individuels ?

1.5 Hypothèse

Les hypothèses formulées dans ce travail sont les suivantes :

- Les techniques d'intelligence artificielle peuvent contribuer à la prévention primaire de l'AVC grâce à un modèle capable d'évaluer le risque individuel.
- Les performances des algorithmes d'apprentissage automatique permettront une évaluation fiable et pertinente du risque d'Accident Vasculaire Cérébral.

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

1.6 Objectif

1.6.1 Objectif général

L'objectif de la recherche est de développer un modèle d'intelligence artificielle (*apprentissage automatique)* capable de prédire le risque d'accident vasculaire cérébral pour une personne sur base des facteurs de risques individuels.

La solution permettra ainsi aux professionnels de santé d'opter pour des mesures préventives adéquates et de ce fait, de réduire l'incidence de l'accident vasculaire cérébral.

2. METHODOLOGIE

Cette partie présente l'approche méthodologique adoptée pour mener à bien la présente étude. Elle décrit les méthodes et techniques utilisées pour la collecte, l'analyse et l'interprétation des données, ainsi que la population concernée et l'échantillon retenu.

2.1. Méthodes de recherche

Dans le cadre de cette étude, la méthode analytique a été privilégiée. Elle a permis d'examiner en profondeur les données disponibles afin d'identifier les relations existantes entre les facteurs de risque et la survenue de l'Accident Vasculaire Cérébral.

Cette méthode s'est avérée appropriée pour comprendre les mécanismes sous-jacents aux phénomènes étudiés et pour construire un modèle prédictif fiable.

2.2. Techniques de collecte des données

La technique documentaire a été utilisée pour la collecte des informations théoriques. Elle a consisté à consulter des ouvrages scientifiques, des articles de recherche, des mémoires académiques ainsi que des ressources en ligne spécialisées, permettant de consolider les bases conceptuelles du travail.

Par ailleurs, les données utilisées pour l'analyse empirique ont été extraites d'un jeu de données secondaire disponible sur la plateforme Kaggle. Ce jeu de données regroupe des informations relatives aux caractéristiques sociodémographiques, cliniques et comportementales des individus.

2.3. Techniques d'analyse des données

L'analyse des données a reposé sur les techniques de l'apprentissage automatique. Après une phase de préparation des données (nettoyage, encodage et normalisation), plusieurs algorithmes de classification ont été mis en œuvre afin de prédire le risque d'Accident Vasculaire Cérébral.

Les données ont été divisées en ensembles d'entraînement et de test afin d'évaluer la capacité de généralisation des modèles. Les performances des algorithmes ont été appréciées à l'aide d'indicateurs statistiques tels que la précision, la sensibilité, la spécificité et la matrice de confusion.

2.4. Techniques d'interprétation des résultats

L'interprétation des résultats a consisté à comparer les performances obtenues par les différents modèles développés. Cette analyse comparative a permis d'identifier l'algorithme le plus performant pour l'évaluation du risque d'AVC.

Les résultats ont ensuite été interprétés à la lumière des connaissances théoriques issues de la littérature scientifique, afin d'en dégager la portée et les limites dans un contexte de prévention primaire.

2.5. Population

La population cible de cette étude est constituée d'individus adultes, hommes et femmes, susceptibles de présenter des facteurs de risque associés à l'Accident Vasculaire Cérébral.

Cette population est représentative de personnes exposées à des risques cardiovasculaires variés, incluant des facteurs modifiables et non modifiables.

2.6. Échantillon

L'échantillon utilisé dans cette étude est composé de 5 229 individus, issus du jeu de données exploité. Ces données proviennent d'une base publique mise à disposition sur la plateforme Kaggle.

L'échantillon comprend des hommes et des femmes présentant différentes caractéristiques telles que l'âge, le sexe, l'hypertension, le diabète, le statut tabagique, l'indice de masse corporelle et d'autres variables pertinentes pour l'évaluation du risque d'AVC.

Le choix de cet échantillon se justifie par sa taille et sa diversité, permettant une analyse statistique et prédictive fiable.

3. RESULTATS (PRESENTATION DES RESULTATS)

La présente étude vise à développer un modèle de prédiction de l'accident vasculaire cérébral. Pour ce faire et parvenir aux résultats escomptés, il importe d'employer une chaine de traitement adéquate.

Nous proposons dans ce travail l'implémentation de la solution sur le navigateur Anaconda, avec le langage de programmation python. Le modèle de prédiction sera développé avec trois algorithmes : la forêt d'arbres aléatoires, les machines à vecteurs de support et les arbres de décision. On retiendra à la fin du processus l'algorithme offrant les meilleures performances.

3.1. Outils de développement

3.1.1. Anaconda

Anaconda est un outil en distribution libre et open source destiné à la programmation Python et R. Il est beaucoup utilisé en science de données, machine learning et l'intelligence artificielle car il contient plusieurs packages nécessaires dans ce domaine notamment *Python, Numpy, Panda, Jupyter, etc*. Et comme le langage Python, il est multiplateforme.

Ce logiciel est primordial et incontournable pour tous les développeurs dans le domaine de la data science. Grâce à sa capacité de collecter et transformer des données à grande échelle ainsi qu'aux outils qu'il propose.

Il existe plusieurs applications dans Anaconda Navigator telles que :

- JupyterNotebook
- Spyder
- Pycharm
- VSCode
- RStudio
- Anaconda powerShell

3.1.2. Jupyter Notebook

Jupyter Notebook est une application web qui vous permet de stocker des lignes de code Python, les résultats de l'exécution de ces dernières (graphiques, tableaux, etc.) et du texte formaté. Cela peut être comparé à une page web contenant du code Python.

3.1.3. Python

Python est un langage de programmation interprété, multiparadigme et multiplateformes. Il favorise la programmation impérative structurée, fonctionnelle et orientée objet. Il est doté d'un typage dynamique fort, d'une gestion automatique de la mémoire par ramassemiettes et d'un système de gestion d'exceptions ; il est ainsi similaire à Perl, Ruby, Scheme, Smalltalk et Tcl.

Le langage Python est placé sous une licence libre proche de la licence BSD et fonctionne sur la plupart des plateformes informatiques, des smartphones aux ordinateurs centraux, de Windows à Unix avec notamment GNU/Linux en passant par

macOS, ou encore Android, iOS, et peut aussi être traduit en Java ou .NET. Il est conçu pour optimiser la productivité des programmeurs en offrant des outils de haut niveau et une syntaxe simple à utiliser[10].

Nous l'utiliserons dans ce travail car c'est un langage de référence en intelligence artificielle. En raison notamment de la simplicité de sa syntaxe ainsi que ses nombreuses bibliothèques adaptées aux projets de science de données.

3.2. Construction du modèle

Un modèle d'apprentissage automatique est une représentation mathématique d'un processus réel. La construction d'un tel modèle exige que certaines étapes soient respectées. Les étapes pour développer un modèle d'apprentissage automatique sont :

- Acquisition des données
- Préparation ou exploration des données
- Sélection de l'algorithme
- Entraînement du modèle
- Evaluation du modèle
- Déploiement
- Test
- Evaluation

Les étapes d'acquisition et de préparation de données correspondent à la phase « ***Acquisition et compréhension de données*** » du processus TDSP vu dans le chapitre précédent. La sélection de l'algorithme, l'entraînement du modèle et l'évaluation du modèle correspondent à la phase « ***Modélisation*** » du processus TDSP. Enfin l'étape déploiement correspond à la phase « ***Déploiement*** » du processus TDSP.

L'acquisition de données ayant été réalisée au chapitre précédent, nous procéderons aux étapes d'exploration, de modélisation, de déploiement et de test.

3.2.1. Importation des bibliothèques

Avant de procéder aux différentes étapes de développement du modèle, il est primordial d'importer toutes les bibliothèques nécessaires. Les bibliothèques à importer sont les suivantes :

- **Numpy** : permet d'effectuer des opérations sur des tableaux et des matrices en Python. Elle permet de traiter des tableaux qui stockent des valeurs de même type et facilite l'exécution d'opérations mathématiques sur les tableaux.
- **Pandas** : permet de convertir des structures de données en objets dataframe. De gérer les données manquantes, d'ajouter/supprimer des colonnes de dataframe, d'imputer les fichiers manquants et de tracer les données avec un histogramme.
- **Matplotlib** : aide à générer des visualisations de données telles que des diagrammes et des graphiques bidimensionnels (histogrammes, diagrammes de dispersion, graphiques de coordonnées non cartésiennes).
- **Scikit-learn** : utilisé pour traiter les tâches standards de Machine Learning telles que le regroupement, la régression, la sélection de modèles, la réduction de la dimensionnalité et la classification.
- **Seaborn** : basé sur Matplotlib, elle sert d'outil de Machine Learning Python utile pour la visualisation de modèles statistiques, cartes thermiques et autres types de visualisations qui résument les données et dépeignent les distributions globales.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

L'image ci-dessous montre l'importation de ces différentes bibliothèques :

```
# IMPORTATION DE TOUTES LES LIBRAIRIES

import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import make_pipeline
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.preprocessing import PolynomialFeatures, StandardScaler
from sklearn.metrics import f1_score, confusion_matrix, classification_report
from sklearn.model_selection import learning_curve
```

Figure 1: Importation de bibliothèques

3.2.2. Importation de l'ensemble de données

Avant de procéder à l'étape d'exploration, il est important de charger l'ensemble de données (dataset) dans l'environnement d'analyse. Cette tâche a été accomplie dans le précédent chapitre. Notre dataset se présente sous forme d'un fichier Excel et porte le nom de dat.xlsx[1]. Il contient 5229 exemples et 8 colonnes.

Le code ci-dessous montre le chargement du dataset dans Jupyter Notebook :

```
# IMPORTATION DU DATASET

data = pd.read_excel('dat.xlsx')
data.head()
```

	Genre	Age	Hypertension	Maladie_cardiaque	Glycémie_moyenne	Imc	statut_tabagique	AVC
0	Male	67.0	0	1	228.69	36.6	formerly smoked	1
1	Male	80.0	0	1	105.92	32.5	never smoked	1
2	Female	49.0	0	0	171.23	34.4	smokes	1
3	Female	79.0	1	0	174.12	24.0	never smoked	1
4	Male	81.0	0	0	186.21	29.0	formerly smoked	1

Figure 2: Importation du dataset

Ce dataset comprend les variables ci-après:

- Genre
- Age
- Hypertension
- Maladies cardiaques
- Glycémie moyenne
- Indice de masse corporelle (imc)
- Statut tabagique
- AVC (variable cible)

[1] .xlsx est une extension de nom de fichier pour tableur au format Office Open XML utilisé par Microsoft Office à partir de la version 2007

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

La variable AVC est la cible du modèle. Autrement, c'est la valeur qu'on cherche à prédire. Toutes les autres variables sont les caractéristiques du modèle.

3.3. Exploration de données

Elle consiste à explorer un large ensemble de données pour y découvrir des tendances, caractéristiques et corrélations à examiner plus en profondeur par la suite. On utilise diverses techniques statistiques pour définir les caractéristiques de l'ensemble de données : taille, quantité, qualité, nature.

En observant le dataset, on constate qu'il comprend 5229 lignes et 8 colonnes. Il n'y a aucune valeur manquante.

```
# EXPLORATION ET PRETRAITEMENT DE DONNEES
# Taille du dataset

df.shape
```

```
(5229, 8)
```

```
# Vérification des valeurs manquantes

df.isna().sum()/df.shape[0]
```

```
Genre                0.0
Age                  0.0
Hypertension         0.0
Maladie_cardiaque    0.0
Glycémie_moyenne     0.0
Imc                  0.0
statut_tabagique     0.0
AVC                  0.0
```

Figure 3: Taille de données et observation de valeurs manquantes

En examinant de plus près la variable cible **AVC,** on découvre qu'il y a dans le dataset 4733 cas négatifs et 496 cas positifs soit 90% de cas négatifs et 0.9% de cas positifs. Ce qui nous permet de comprendre que nous avons à faire à un problème de **déséquilibre de classes**.

Les classes déséquilibrées font référence à un problème de classification où les classes ne sont pas représentées de façon égale. Ce déséquilibre de classe augmente nettement la difficulté de l'apprentissage par l'algorithme de classification. En effet, l'algorithme n'a que peu d'exemples de la classe minoritaire sur lesquels apprendre. Il est donc biaisé vers la population des négatifs et produit des prédictions potentiellement moins robustes qu'en l'absence de déséquilibre. Dans l'image ci-dessous la répartition inégale de cas négatifs et positifs :

```
# Analyse de la target

df['AVC'].value_counts()
```

```
0    4733
1     496
Name: AVC, dtype: int64
```

```
df['AVC'].value_counts(normalize = True)
```

```
0    0.905144
1    0.094856
Name: AVC, dtype: float64
```

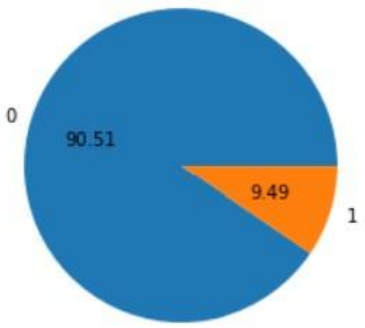

Figure 4: Analyse de la variable cible

Il existe une relation entre la variable cible (AVC) et toutes les autres variables du dataset. Cette corrélation est illustrée par le graphique suivant :

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

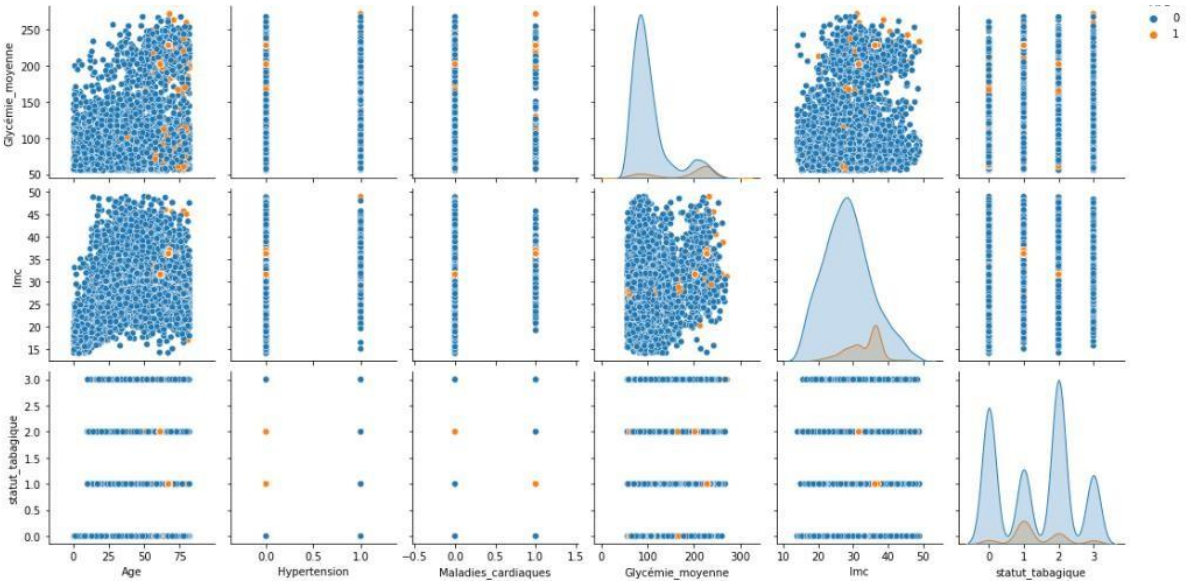

Figure 5: Relation entre la variable cible et quelques variables du dataset

Par ailleurs, il existe aussi un lien entre les différentes variables du dataset entre elles. Ce lien est donné dans une matrice de corrélation.

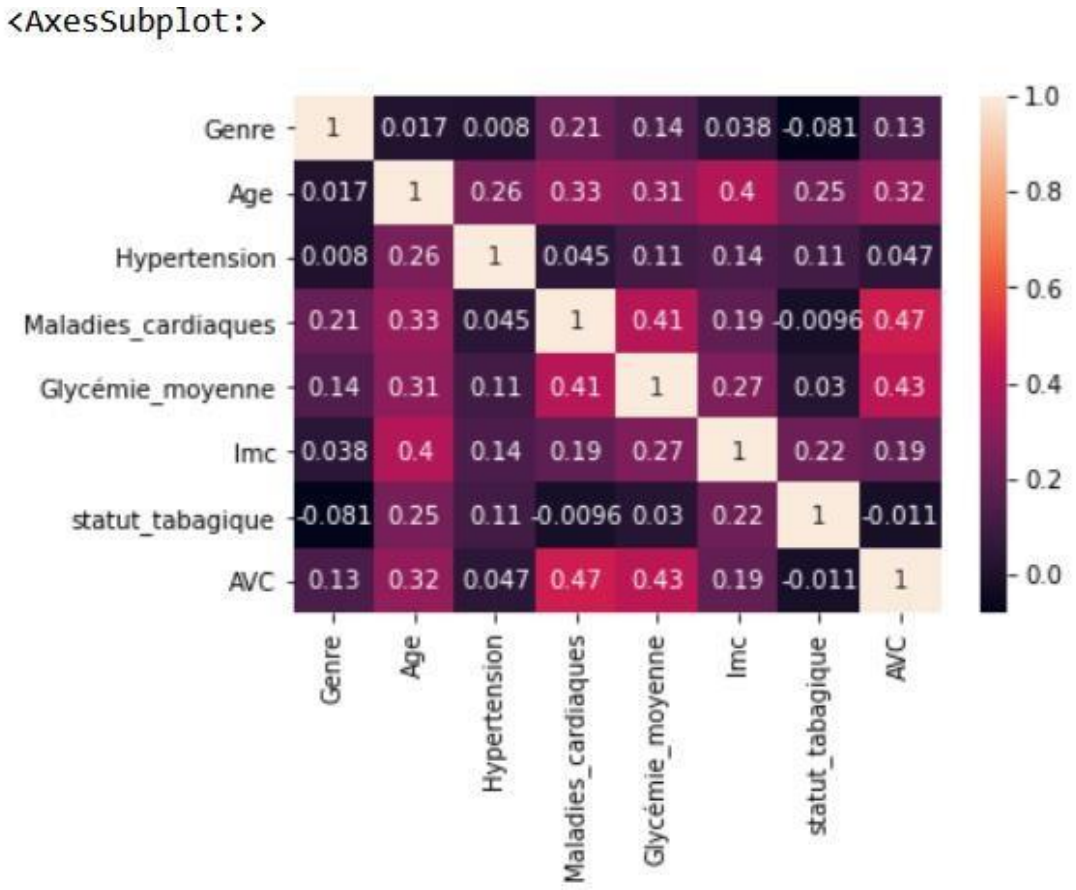

Figure 6: Matrice de corrélations

Nous remarquons aussi que les colonnes statut tabagique et genre sont des variables qualitatives qu'il faut convertir en variables quantitatives. Pour ce faire nous procédons à l'encodage de ces variables.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

```
from sklearn.preprocessing import LabelEncoder
encoder = LabelEncoder()
```

```
# Encodage

df['statut_tabagique'] = encoder.fit_transform(df['statut_tabagique'])
df['Genre'] = encoder.fit_transform(df['Genre'])
```

Figure 7: Encodage de variables genre et statut tabagique

```
df.head()
```

	Genre	Age	Hypertension	Maladies_cardiaques	Glycémie_moyenne	Imc	statut_tabagique	AVC
0	1	67.0	0	1	228.69	36.6	1	1
1	1	80.0	0	1	105.92	32.5	2	1
2	0	49.0	0	0	171.23	34.4	3	1
3	0	79.0	1	0	174.12	24.0	2	1
4	1	81.0	0	0	186.21	29.0	1	1

Figure 8: Dataset après encodage

Analyse de la distribution de quelques variables :

En étudiant la distribution de la variable âge on comprend que l'âge des individus représentés dans le dataset varie de 0 à 80 ans et qu'entre 60 et 80 ans nous atteignons un pic.

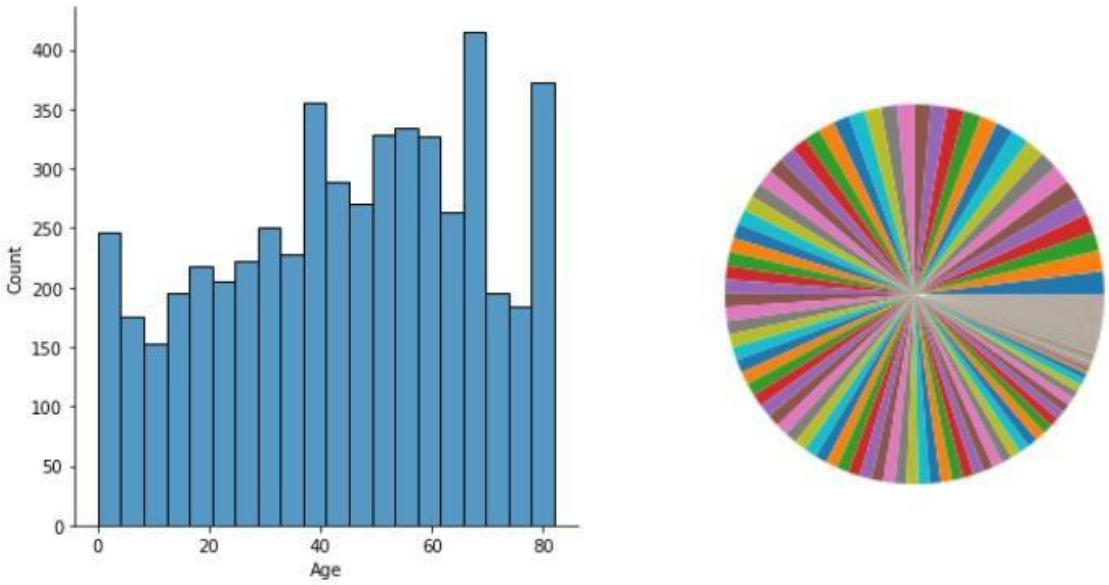

Figure 9: Distribution de la variable âge

Pour ce qui est de l'indice de masse corporelle, nous remarquons que nous atteignons un pic lorsque cet indice est de 30. Ce qui signifie qu'il y a un grand nombre d'individus obèses dans le dataset.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

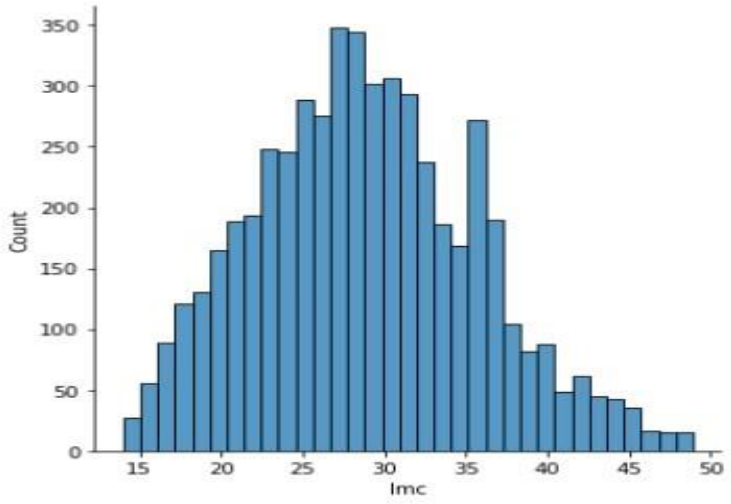

Figure 10: Distribution de l'indice de masse corporelle

Quant à l'hypertension artérielle, seulement 9.16% des individus sont testés positifs et 90.84% sont négatifs.

```
plt.pie(df['Hypertension'].value_counts(), labels=[0, 1], autopct="%0.2f")
plt.show()
```

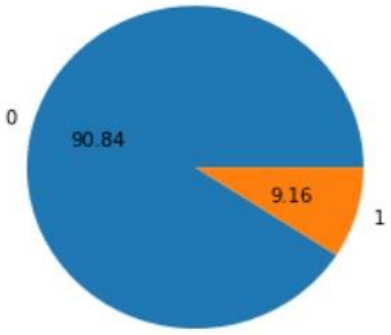

Figure 11: Analyse de la variable hypertension

Après l'étude des différentes variables, le dataset est divisé en deux parties : une première partie appelée **trainset** qui contient les données qui serviront à l'entraînement du modèle et une deuxième partie appelée **testset** qui comprend les données qui serviront au test. Après répartition, le trainset comprend 3789 observations négatives et 394 observations positives. Et le testset à son tour comprend 944 observations négatives et 102 observations positives.

```
# PRETRAITEMENT DE DONNEES

# Répartition de données d'entraînement et de test

trainset, testset = train_test_split(dataf, test_size=0.2, random_state=0)
```

```
trainset['AVC'].value_counts()
```

```
0    3789
1     394
Name: AVC, dtype: int64
```

```
testset['AVC'].value_counts()
```

```
0    944
1    102
Name: AVC, dtype: int64
```

Figure 12: Répartition de données d'entraînement et de test

3.4. Modélisation

3.4.1. Sélection de l'algorithme

Dans cette phase, différents algorithmes sont entraînés sur les données afin de retenir celui qui offre les meilleures performances. Puisqu'il s'agit d'un mode d'apprentissage supervisé, les algorithmes sélectionnés sont :

- Arbre de décision
- Forêt d'arbres aléatoires (*Random Forest*)
- Machines à vecteur de support (*Support Vector Machines*)
- Régression logistique

Mais avant de procéder à l'entraînement du modèle, il importe de remédier au problème de déséquilibre de classes afin d'avoir des prédictions exactes.

3.4.2. Rééquilibrage de classes

En cas de déséquilibre de classe, la difficulté d'apprentissage de l'algorithme est fortement augmentée car celui-ci n'a que peu d'exemples de la classe minoritaire sur lesquels apprendre. Plusieurs approches sont possibles lorsqu'on désire gérer le problème de classes déséquilibrées.

Face à un dataset déséquilibré, plusieurs options se présentent :

- **Collecter davantage de données**

Cela peut paraître simpliste, mais la collecte de données supplémentaires est presque toujours négligée et peut parfois s'avérer efficace.

Il est important de réfléchir à la possibilité de collecter davantage de données pour le problème. Cela pourrait éventuellement rééquilibrer les classes à un degré variable.

- **Utiliser des méthodes de ré-échantillonnage**

Le ré-échantillonnage est une stratégie qui permet de modifier l'ensemble de données utilisé afin d'avoir des données plus équilibrées. Cette stratégie comporte deux méthodes qui permettent d'égaliser les classes : ***l'oversampling*** et ***l'undersampling*.**

L'oversampling ou le sur-échantillonnage en français, fonctionne en augmentant le nombre d'observations de la classe minoritaire afin d'arriver à un ratio classe minoritaire/classe majoritaire satisfaisant.

L'undersampling ou le sous-échantillonnage en français, fonctionne en diminuant le nombre d'observations de la classe majoritaire afin d'arriver à un ratio classe minoritaire/classe majoritaire satisfaisant.

- **Générer des échantillons synthétiques**

Il existe des algorithmes pour générer des échantillons synthétiques de manière automatique. Le plus populaire de ces algorithmes est SMOTE (*Synthetic Minority Oversampling Technique*). Comme son nom l'indique, SMOTE est une méthode de suréchantillonnage. Elle fonctionne en créant des échantillons synthétiques à partir de la classe minoritaire au lieu de créer de simples copies.

3.4.2.1. Présentation de la méthode SMOTE

SMOTE (Synthetic Minority Oversampling Technique) est une technique statistique permettant d'augmenter le nombre de cas d'un jeu de données de façon équilibrée. Le composant fonctionne par génération de nouvelles instances à partir de cas minoritaires fournis en entrée. Cette implémentation de SMOTE ne modifie pas le nombre de cas majoritaires.

Les nouvelles instances ne sont pas seulement des copies des cas minoritaires existants. En réalité, l'algorithme prend des échantillons de l'espace de caractéristiques pour chaque classe cible et ses plus proches voisins. L'algorithme génère ensuite de nouveaux échantillons qui combinent les caractéristiques du cas cible avec des caractéristiques de ses voisins. Cette

approche augmente le nombre de caractéristiques disponibles pour chaque classe et rend les échantillons plus généraux. SMOTE prend en entrée l'ensemble du jeu de données, mais augmente le pourcentage des cas minoritaires uniquement. Supposons, par exemple, un jeu de données déséquilibré dans lequel seulement 1 % des cas ont la valeur cible A (classe minoritaire) et 99 % des cas la valeur B. Pour multiplier par deux le pourcentage de cas minoritaires, entrez 200 comme Pourcentage SMOTE dans les propriétés du composant.

3.4.2.2. Application de la méthode SMOTE à l'ensemble de données

En appliquant la méthode SMOTE sur notre dataset déséquilibré, nous obtenons un nombre égal d'observations positives et négatives. Soit 3789 cas positifs et négatifs.

```
from imblearn.over_sampling import SMOTE

print("Avant OverSampling, comptage des labels '1': {}".format(sum(y_train==1)))
print("Avant OverSampling, comptage des labels '0': {} \n".format(sum(y_train==0)))

sm = SMOTE(random_state=2)
X_train_smote, y_train_smote = sm.fit_resample(X_train, y_train.ravel())

print('Après OverSampling, la taille de X_train: {}'.format(X_train_smote.shape))
print('Après OverSampling, la taille de y_train: {} \n'.format(y_train_smote.shape))

print("Après OverSampling, comptage des labels '1': {}".format(sum(y_train_smote==1)))
print("Après OverSampling, comptage des labels '0': {}".format(sum(y_train_smote==0)))
```

Figure 13: Application de la méthode Smote sur l'ensemble de données

3.4.3. Entraînement et évaluation du modèle

3.4.3.1. Métriques de performance

Avant d'entraîner les algorithmes, il est important de poser les bases théoriques sur les métriques de performances. En effet, la connaissance de ces métriques permet de faire un choix judicieux de métriques à utiliser. Une métrique de performance ou d'évaluation quantifie la performance d'un modèle prédictif. La qualité d'un modèle de classification dépend de la métrique utilisée pour l'évaluer. Les métriques les plus utilisées en Machine Learning sont : l'accuracy, la précision, la sensibilité et le F1 score.

- **L'accuracy**

L'accuracy est une métrique utilisée pour évaluer la performance des modèles de classification à 2 ou plusieurs classes. Elle est traduite en français par « *précision* » mais ne doit pas être confondue avec la métrique précision. Elle mesure le taux de prédictions correctes sur l'ensemble des individus. Elle est calculée par la formule ci-dessous :

$$\textbf{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

Avec:

TP: True positives (Vrais positifs)

TN: True Negatives (Vrais négatifs)

FP: False Positives (Faux positifs)

FN: False Negatives (Faux Negatifs)

- **La précision**

C'est le pourcentage d'instances positives par rapport au nombre total d'instances positives prévues. Ici le dénominateur est la prédiction du modèle effectuée comme positive à partir de l'ensemble de données. Autrement, elle indique combien du nombre total de prédictions spécifiées comme positives sont correctement affectées. Elle est calculée par :

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

$$\textbf{Precision} = \frac{TP}{TP+FP}$$

- **Le rappel ou la sensibilité**

C'est le pourcentage d'instances positives par rapport au nombre total d'instances positives réelles. Le dénominateur est le nombre réel d'instances positives présentes dans l'ensemble de données.

$$\textbf{Rappel} = \frac{TP}{TP+FN}$$

- **Le F1 score**

Le f1-score est la moyenne harmonique de la précision et du rappel. Cela prend la contribution des deux, donc plus le score f1 est élevé mieux c'est.

$$\textbf{F1 score} = \frac{2*TP}{2*TP+FP+FN}$$

3.4.3.2. Matrice de confusion

La matrice de confusion appelée aussi tableau de contingence, permet d'examiner l'apprentissage d'un modèle de classification. Elle montre donc à quel point un certain modèle peut être confus lorsqu'il fait des prédictions. Il existe dans une matrice de confusion quatre terminologies principales à comprendre : TP, TN, FP et FN.

		Classe réelle	
		Positif	Négatif
Classe prédite	positif	TP	FP
	Négatif	FN	TN

Tableau 1: Représentation d'une matrice de confusion

Avec :

Termes	Significations	Descriptions
TP	True positive	Cas positifs prédits comme positifs
TN	True negative	Cas négatifs prédits comme négatifs
FN	False negative	Cas positifs prédits comme négatifs
FP	False positif	Cas négatifs prédits comme positifs

Tableau 2: Termes associés à la matrice de confusion

3.4.3.3. Entraînement de la Random Forest

Le dataset étant déjà divisé en trainset et en testset, nous procédons directement à l'entraînement du modèle avec la RandomForestClassifier. L'entraînement du modèle se fait grâce à la méthode fit () et l'évaluation avec la méthode score () comme le montre l'image ci-dessous :

```
# Entraînement de la RandomForest

rf = RandomForestClassifier(n_estimators = 50, random_state = 100)
model =  rf.fit(X_train_smote, y_train_smote)
model_predict = model.predict(X_test)

print("La performance sur le trainset est : ", rf.score(X_train, y_train))
print("La performance sur le testset est : ", rf.score(X_test, y_test))
print(confusion_matrix(y_test, model_predict))
print(classification_report(y_test, model_predict))
```

Figure 14: Entraînement de la Random Forest

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

```
La performance sur le trainset est :  0.9998087588449034
La performance sur le testset est :  0.9998087588449034
[[4733    0]
 [   1  495]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      4733
           1       1.00      1.00      1.00       496

    accuracy                           1.00      5229
   macro avg       1.00      1.00      1.00      5229
weighted avg       1.00      1.00      1.00      5229
```

Figure 15: Evaluation de la Random Forest

Comme indiqué sur l'image ci-dessus, la Random Forest obtient un score de 99% sur les données d'entraînement et 99% sur les données de test. La matrice de confusion obtenue avec la Random Forest se présente comme suit :

		Classe réelle	
		Positif	Negatif
Classe prédite	positif	4733	0
	Negatif	1	495

Tableau 3: Matrice de confusion obtenue avec la Random Forest

On peut voir clairement que le modèle obtient 4733 vrais positifs, 1 faux négatif, 495 vrais négatifs et 0 faux positif.

3.4.3.4. Entraînement de l'arbre de décision

Un meilleur score est obtenu avec l'arbre de décision qui réalise 100% sur les données d'entraînement et 100 % sur les données de test.

```
# Entraînement de l'arbre de décision

dt = DecisionTreeClassifier()
decision =  dt.fit(X_train_smote, y_train_smote)
predict_model = decision.predict(X_test)

print("La performance sur le trainset est : ", dt.score(X_train, y_train))
print("La performance sur le testset est : ", dt.score(X_test, y_test))
print(confusion_matrix(y_test, predict_model))
print(classification_report(y_test, predict_model))
```

Figure 16: Entraînement de l'arbre de décision

```
La performance sur le trainset est :  1.0
La performance sur le testset est :  1.0
[[4733    0]
 [   0  496]]
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      4733
           1       1.00      1.00      1.00       496

    accuracy                           1.00      5229
   macro avg       1.00      1.00      1.00      5229
weighted avg       1.00      1.00      1.00      5229
```

Figure 17: Evaluation de l'arbre de décision

La matrice de confusion obtenue avec l'arbre de décision :

		Classe réelle	
		Positif	Negatif
Classe prédite	positif	4733	0
	Negatif	0	496

Tableau 4: Matrice de confusion avec l'arbre de décision

Le modèle obtient 4733 vrais positifs, 0 faux positifs, 0 faux négatifs et 496 vrais négatifs.

3.4.3.5. Entraînement du Support Vector Machines

Le Support Vector Machine réalise un score de 79% sur les données d'entraînement et 79% sur les données de test.

```
#  Entrainement du SVC

sv = SVC(random_state = 0)
model2 =  sv.fit(X_train_smote, y_train_smote)
prediction = model2.predict(X_test)

print("La performance sur le trainset est : ", sv.score(X_train, y_train))
print("La performance sur le testset est : ", sv.score(X_test, y_test))
print(confusion_matrix(y_test, prediction))
print(classification_report(y_test, prediction))
```

Figure 18: Entraînement du support vector machine

```
La performance sur le trainset est :  0.7980493402180149
La performance sur le testset est :  0.7980493402180149
[[3746  987]
 [  69  427]]
              precision    recall  f1-score   support

           0       0.98      0.79      0.88      4733
           1       0.30      0.86      0.45       496

    accuracy                           0.80      5229
   macro avg       0.64      0.83      0.66      5229
weighted avg       0.92      0.80      0.84      5229
```

Figure 19: Evaluation du support vector machine

Ci-dessous, la matrice de confusion obtenue avec le SVM :

		Classe réelle	
		Positif	Negatif
Classe prédite	positif	3746	987
	Negatif	69	427

Tableau 5: Matrice de confusion avec le support vector machine

Le support vector machine obtient 3746 vrais positifs, 987 faux positifs, 69 faux négatifs et 427 vrais négatifs.

3.4.3.6. Entraînement de la régression logistique

Après entraînement, la régression logistique donne un f1 score égale à 46%, une précision de 32%, un rappel de 80% et un accuracy de 82%.

Novelty Journals

ISSN 2394-7314

International Journal of Novel Research in Computer Science and Software Engineering
Vol. 13, Issue 1, pp: (1-19), Month: January - April 2026, Available at: www.noveltyjournals.com

```
#  Entrainement de la régression logistique

lr = LogisticRegression(random_state = 0)
mod =  lr.fit(X_train_smote, y_train_smote)
predire = mod.predict(X_test)

print("La performance sur le trainset est : ", lr.score(X_train, y_train))
print("La performance sur le testset est : ", lr.score(X_test, y_test))
print(confusion_matrix(y_test, predire))
print(classification_report(y_test, predire))
```

Tableau 6: Entrainement de la régression logistique

```
La performance sur le trainset est :  0.8225282080703767
La performance sur le testset est :  0.8225282080703767
[[3902  831]
 [  97  399]]
              precision    recall  f1-score   support

           0       0.98      0.82      0.89      4733
           1       0.32      0.80      0.46       496

    accuracy                           0.82      5229
   macro avg       0.65      0.81      0.68      5229
weighted avg       0.91      0.82      0.85      5229
```

Figure 20: Evaluation de la régression logistique

3.5. Nouvelles prédictions

La Random Forest étant l'un des algorithmes qui offre les meilleures performances, c'est celui qui sera retenu pour le test afin de faire des nouvelles prédictions. Pour ce faire, nous prendrons deux cas de figure.

Dans le premier exemple, nous entrons les informations suivantes :

- Genre : masculin (1),
- Age : 30,
- Hypertendu : non (0),
- Maladies cardiaques : non (0),
- Glycémie moyenne : 103, □ Indice de masse corporelle : 25, □ Statut tabagique : 2(non-fumeur).

La prédiction se révèle être négative c'est-à-dire que l'individu ne présente aucun risque de survenue d'AVC.

```
# Estimation du risque

risque = [[1, 30, 0, 0, 103, 25, 2]]

# Prediction du risque en utilisant la Random Forest

prédiction = model.predict(risque)
score = model.predict_proba(risque)
print(prédiction)

if prédiction == 1:
    print("Vous êtes une potentielle victime de l'AVC.\n Probabilité : ",score)
else:
    print ('Tout va bien. Vous ne risquez rien.')
```

```
[0]
Tout va bien. Vous ne risquez rien.
```

Figure 21: Premier test du modèle

Dans le deuxième exemple, nous entrons les informations suivantes :

- Genre : Féminin(0),
- Age : 65,
- Hypertendu : oui(1),
- Maladies cardiaques : non(0),
- Glycémie moyenne : 223, □ Indice de masse corporelle : 26, □ Statut tabagique : fumeur (3).

La prédiction se révèle être positive c'est-à-dire que l'individu présente un risque de survenue d'AVC. La probabilité est de 92%.

```
# Estimation du risque

risque = [[1, 65, 1, 0, 223, 26, 3]]

# Prediction du risque en utilisant la Random Forest

prédiction = model.predict(risque)
score = model.predict_proba(risque)
print(prédiction)

if prédiction == 1:
    print("Vous êtes une potentielle victime de l'AVC.\n Probabilité : ",score)
else:
    print ('Tout va bien. Vous ne risquez rien.')
```

```
[1]
Vous êtes une potentielle victime de l'AVC.
 Probabilité :  [[0.08 0.92]]
```

Figure 22: Deuxième test du modèle

Nous avons procédé au développement du modèle d'évaluation du risque d'accident vasculaire cérébral. Avant toute chose, nous avons présenté quelques outils permettant de développer un modèle d'apprentissage automatique. Notamment le navigateur Anaconda, Jupyter notebook, le langage de programmation python et le tableur Excel.

Durant la construction du modèle, nous avons rencontré un problème de déséquilibre de classe de notre dataset. Problème qui a été résolu grâce à la méthode de ré-échantillonnage SMOTE. Ensuite, quatre algorithmes d'apprentissages ont été entraînés : l'arbre de décision, la forêt aléatoire, les séparateurs à vaste marge et la régression logistique. La forêt aléatoire ou Random Forest est celui qui a été retenu en raison de ses performances.

4. DISCUSSION

La présente section est consacrée à l'analyse critique et à l'interprétation des résultats obtenus au cours de cette étude. Elle vise également à confronter ces résultats aux théories et aux travaux scientifiques existants, afin d'en apprécier la pertinence et la portée dans le contexte de la prévention primaire de l'Accident Vasculaire Cérébral.

4.1. Analyse et interprétation des résultats

Les résultats obtenus à l'issue de l'expérimentation montrent que les techniques d'apprentissage automatique sont capables de prédire le risque d'Accident Vasculaire Cérébral avec un niveau de performance satisfaisant. Les différents algorithmes utilisés ont donné des résultats différents, ce qui montre que le choix du modèle est très important pour la qualité des prédictions. Les modèles utilisant des arbres de décision et des méthodes d'ensemble, comme la Random Forest, ont donné des résultats plus précis, plus sensibles et plus spécifiques. Ces résultats peuvent être expliqués par la possibilité que ces algorithmes ont de traiter bien des données différentes et de représenter des liens complexes entre les variables qui influencent le résultat principal. Ces variables apparaissent comme des éléments déterminants dans l'évaluation du risque d'AVC, ce qui renforce la cohérence clinique des modèles développés.

4.2. Vérification des hypothèses

Les résultats obtenus permettent de confirmer les hypothèses formulées au début de cette recherche.

En effet, l'hypothèse selon laquelle l'intelligence artificielle peut contribuer à la prévention primaire de l'Accident Vasculaire Cérébral est validée par les performances des modèles prédictifs développés.

De plus, l'hypothèse postulant que les algorithmes d'apprentissage automatique sont capables de fournir une évaluation fiable du risque d'AVC est également confirmée. Les indicateurs de performance observés démontrent que ces outils peuvent constituer un appui pertinent à la prise de décision médicale.

4.3. Confrontation des résultats aux théories existantes

Les résultats de ce mémoire sont en adéquation avec les théories de la médecine préventive, qui soulignent l'importance de l'identification précoce des facteurs de risque afin de réduire l'incidence des maladies cardiovasculaires. L'approche prédictive adoptée dans cette étude s'inscrit pleinement dans cette logique de prévention primaire.

Sur le plan théorique, les principes de l'apprentissage automatique reposent sur l'exploitation des données historiques pour anticiper des événements futurs. Les performances obtenues confirment que ces fondements théoriques sont applicables au domaine de la santé, et plus particulièrement à la prévention de l'AVC.

4.4. Comparaison avec les travaux antérieurs

Les résultats de cette étude rejoignent ceux de plusieurs travaux antérieurs portant sur l'utilisation de l'intelligence artificielle dans la prédiction des risques cardiovasculaires. Des études précédentes ont montré que les algorithmes de classification, tels que les arbres de décision, les machines à vecteurs de support et les méthodes d'ensemble, offrent de bonnes performances dans la prédiction des maladies cardiovasculaires.

Toutefois, certaines différences observées entre les performances obtenues dans ce mémoire et celles rapportées dans la littérature peuvent s'expliquer par la nature des données utilisées, la taille de l'échantillon, les variables sélectionnées ainsi que les techniques de prétraitement appliquées.

Malgré ces variations, les résultats actuels confirment la pertinence de l'intelligence artificielle comme outil complémentaire aux méthodes traditionnelles de prévention et de diagnostic.

4.5. Limites de l'étude et perspectives

Bien que les résultats obtenus soient encourageants, cette étude présente certaines limites. Les données utilisées proviennent d'une base secondaire, ce qui peut introduire des biais liés à la qualité ou à la représentativité des informations collectées. De plus, certains facteurs cliniques ou génétiques pertinents n'ont pas été pris en compte faute de disponibilité des données.

Ces limites ouvrent des perspectives pour des travaux futurs, notamment l'intégration de données cliniques plus complètes, l'utilisation de jeux de données locaux et l'exploration de modèles d'apprentissage profond afin d'améliorer davantage les performances prédictives.

5. CONCLUSION

Le présent travail a porté sur l'aide à la prévention de l'accident vasculaire cérébral par les techniques d'analyse prédictive et d'intelligence artificielle. L'objectif poursuivi consistait à prévenir la survenue de l'accident vasculaire cérébral grâce à un modèle d'intelligence artificielle (apprentissage automatique) capable de prédire le risque de cette pathologie sur base de facteurs de risques individuels.

Pour ce faire, nous avons notamment posé les bases théoriques sur l'intelligence artificielle, sa définition, son évolution au fil des ans, sa situation actuelle ainsi que ses domaines. Hormis cela, nous avons évoqué la médecine préventive, révélant les deux types de conception de celle-ci.

En effet, nous avons dû opérer un choix judicieux d'une méthodologie de gestion de projet à utiliser. Après avoir mis en évidence les avantages et inconvénients de plusieurs méthodologies, notre choix s'est porté sur la méthodologie Microsoft TDSP en raison de ses nombreux avantages. Une présentation de la méthodologie a été faite par la suite, révélant son cycle de vie. Cycle de vie qui nous a permis de mener à bien ce projet.

En fin, plusieurs nouveaux concepts ont été abordés. Notamment, les métriques de performances, la matrice de confusion et le rééchantillonnage. Le problème de déséquilibre de classe de l'ensemble de données nous a conduits à avoir recours à la méthode SMOTE pour le rééquilibrer. Après entraînement des quatre algorithmes sélectionnés, la forêt aléatoire était celui qui offrait les meilleures performances et était de ce fait celui qui a été retenu.

REFERENCES

[1] Beaglehole, R. e. (2011). Priority Actions for the Non-Communicable Disease Crisis. *The Lancet*, 1438–1447.

[2] Bejot, Y. e. (2007). Épidémiologie des accidents vasculaires cérébraux. *La Presse Médicale*, 117–127.

[3] Feigin, V. L. (2021). Étude d'observation issue de la collaboration Global Burden of Diseases, Injuries, and Risk Factors Study . *The Lancet Neurology*, 795–820.

[4] Feigin, V. L. (2021). Global burden of stroke. *The Lancet Neurology*, 795–820.

[5] Haiech, J. (2020). Parcourir l'histoire de l'intelligence artificielle, pour mieux la définir et la comprendre. *médecine/sciences*, 919 - 923.

[6] Mayer-Schönberger, V. &. (2013). *A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt.* Boston: Houghton Mifflin Harcourt .

[7] O'Donnell, M. J. (2016). Global and regional effects of potentially modifiable risk factors associated with stroke. *The Lancet*, 761–775.

[8] Obermeyer, Z. &. (2016). Predicting the future Big data, machine learning, and clinical medicine. *The New England Journal of Medicine*, 1216–1219.

[9] OMS. (2005). *Prévention des maladies chroniques : un investissement vital.* Kinshasa: World Health Organization.

[10] Turing, A. (1950). *Computing machinery and intelligence* (Vol. 1236). mind.

[11] Wikipédia, "Python." https://fr.wikipedia.org/wiki/python_langage