Université Mohammed V

Faculté des Science

Département informatique

Licence Professionnelle- Cybersécurité et Ingénierie des systèmes

Rapport de projet

Détection d'Intrusions avec les techniques d'apprentissage automatique

Submitted by

Alaa BAHHOU

Sommaire

Table des figures

I. Introduction

Des attaques informatiques de plus en plus sophistiquées font face à des défis de plus en plus compliqués dans le domaine de la cybersécurité. La détection et la prévention des incursions sont essentielles pour garantir la sécurité des systèmes et des réseaux.

Dans le domaine de la cybersécurité, ce projet vise à utiliser des techniques d'apprentissage automatique pour la détection d'intrusions. Il se divise en deux composants différents mais complémentaires.

La première section comprend l'analyse et le pré-traitement du fichier CICIDS_2017 concernant la cybersécurité.

La seconde partie concerne la classification des attaques en utilisant les modèles d'apprentissage automatique étudiés.

Pour ce projet, le corpus CICIDS 2017 a été choisi car il contient un large nombre d'enregistrements répartis sur plusieurs jours. Nous nous concentrons sur la partie du tableau contenant les attaques onlines d'un jeudi matin afin de simplifier la tâche. Avant de commencer la classification, il était crucial de mener une analyse approfondie du fichier et de mener le pré-traitement nécessaire, en utilisant les méthodes vues en cours. De plus, selon nos besoins, les méthodes de prétraitement supplémentaires seront utilisées.

Les algorithmes d'apprentissage automatique suivants seront utilisés pour classer les attaques :

Les réseaux de neurones, les arbres de décision, la régression logistique, KNN, les SVM, Naïve-Bayes et la forêt d'arbres de décision. Les bibliothèques de scikit-learn, ainsi que de nombreuses autres, seront utilisées pour mettre en œuvre ces algorithmes. De plus, des valeurs métriques d'évaluation seront calculées pour évaluer les performances respectives de chaque algorithme.

Enfin, il sera effectué une comparaison des performances de chaque modèle implémenté. Cette analyse comparative mettra en évidence les forces et les faiblesses de chaque algorithme en termes de détection des attaques.

A. Contexte et problématique

Notre faculté nous a confié un projet pour explorer l'application des techniques d'apprentissage automatique dans le domaine de la détection d'intrusions dans le cadre de notre programme de cybersécurité. Ce projet nous donne la chance d'approfondir nos connaissances en cybersécurité et de mettre en pratique les idées et les méthodes que nous avons étudiées.

Pour ce projet, le dataset CICIDS 2017 a été choisi en raison de la richesse en enregistrements d'attaques online. Cependant, il convient de souligner qu'une problématique supplémentaire se pose: la présence potentielle de données dupliquées dans la collection, ce qui pourrait avoir un impact sur les résultats de l'analyse et de la classification. De plus, nous devrons gérer la diversité et la complexité des attaques présentes dans le dataset.

B. Objectif

La détection d'Intrusions (attaques webs) en utilisant des techniques d'apprentissage automatique mentionnées précédemment.

II. Analyse et prétraitement des données

A. Description du dataset

Le dataset CICIDS2017_thursday_Morning_WebAttacks contient des enregistrements de données collectées à partir de diverses attaques Web contre un réseau informatique simulé. Les données ont été collectées lors d'un test de cyberdéfense qui a eu lieu un jeudi matin, l'année 2017. Le réseau simulé a été construit pour ressembler à un réseau typique utilisé par une organisation, avec des machines virtuelles, des serveurs Web, des serveurs de messagerie et des serveurs de base de données. Les attaques ont été lancées à partir d'une variété de sources et ont été conçues pour imiter les types d'attaques que les réseaux peuvent rencontrer dans la vie réelle.
Le dataset contient des informations sur diverses caractéristiques de réseau, telles que les ports de destination, la durée des flux, le nombre de paquets, la longueur des paquets, les flags TCP, quelques attaques détectées, etc. Il y a un total de 170,366 enregistrements dans le dataset, avec une répartition entre les attaques et le trafic normal.

Le dataset CICIDS2017_thursday_Morning_WebAttacks est un sous-ensemble de la base de données CICIDS2017, qui contient des enregistrements de trafic réseau générés lors d'attaques Web simulées.

Voici une brève exploration du dataset :

- Nombre de lignes : 170.366
- Nombre de colonnes : 79
- Type de données : les données sont principalement numériques, mais il y a aussi une colonne catégorielle.
- Valeurs manquantes : certaines colonnes contiennent des valeurs manquantes, qui devront être gérées avant toute analyse.

- Classes : la variable cible est "Label", qui indique si un enregistrement est lié à une attaque ou non. Les deux classes sont "BENIGN" (trafic normal) et "Web Attack" (une variété d'attaques Web telles que XSS, SQL injection, etc.).
- Distribution des classes : il y a un fort déséquilibre entre les classes, avec la majorité des enregistrements étant "BENIGN" (96,7%) et seulement une petite proportion d'entre eux étant des attaques Web (3,3%).
- Caractéristiques : les autres colonnes sont des caractéristiques du trafic réseau, telles que la durée de la communication, le nombre de paquets envoyés et reçus, la taille des paquets, les statistiques des délais entre les paquets, etc.

Pour effectuer une analyse ou une classification sur ce dataset, il faudra donc prendre en compte les données inutiles des classes et gérer les valeurs manquantes avant de pouvoir utiliser les données.

B. Exploration des données

Au cours de ce projet, nous avons utilisé des techniques d'apprentissage automatique pour préparer les données pour la classification des attaques en traitant le fichier CICIDS 2017. Pour garantir la qualité et la fiabilité des résultats, le prétraitement des données est une étape cruciale.

Pour commencer, nous avons effectué une analyse approfondie du groupe afin de comprendre la structure et les différentes variables qu'il contient. Nous avons pu réduire la dimensionnalité du dataset et se concentrer sur les données les plus importantes après avoir identifié les caractéristiques pertinentes pour la détection des attaques online.

Après cela, nous avons géré les données dupliquées inbound et outbound, qui pourraient être un problème avec la collection. Nous avons identifié et supprimé les enregistrements en double à l'aide de méthodes spécifiques, garantissant ainsi l'intégrité des données et éliminant tout biais induit par ces duplications.

La gestion des valeurs manquantes a été une autre étape du prétraitement. Nous avons analysé les variables du dataset et utilisé des stratégies appropriées pour remplir ou éliminer les valeurs manquantes et les valeurs infinies, en veillant à préserver l'intégrité des données et à éviter toute distorsion dans l'analyse ultérieure.

De plus, afin de mettre toutes les variables à la même échelle dans certains algorithmes, nous avons effectué une normalisation des données. Cela permet d'éviter que certaines variables dominantes prennent le dessus dans l'analyse.

Enfin, nous avons pris en compte la gestion de la mémoire en optimisant notre dataset afin que le calcul se fasse plus rapidement.

En effectuant ces étapes de prétraitement, nous avons veillé à créer un dataset propre et optimisé, prêt à être utilisé pour la classification des attaques en utilisant les techniques d'apprentissage automatique étudiées. Les étapes de prétraitement ont amélioré la qualité des données et réduit les sources potentielles d'erreur, garantissant des résultats plus fiables et pertinents pour renforcer la sécurité des systèmes informatiques.

C. Traitement des données manquantes et aberrantes

Il existe une variété de techniques courantes utilisées pour traiter les données manquantes et aberrantes. La première étape implique l'utilisation de la méthode "dropna()" en python pour supprimer les lignes contenant des valeurs manquantes. Lorsque le nombre de lignes avec des valeurs manquantes est très faible et que leur suppression n'affecte pas significativement les résultats de l'analyse, cette méthode est utile.

```
df = df.dropna()# drop rows with missing values
```

La deuxième méthode consiste à utiliser la méthode python "drop_duplicates()" pour supprimer les lignes en double. Lorsque les données ont été collectées à partir de plusieurs sources ou lorsqu'il y a des doublons, cette méthode est utile.

```
df = df.drop_duplicates()#drop duplicate rows
```

La troisième méthode consiste à utiliser la classe "LabelEncoder()" de la bibliothèque "scikit-learn" pour effectuer une "label encoding" pour les variables catégorielles. Pour convertir les variables catégorielles en variables numériques afin qu'elles puissent être utilisées dans les algorithmes d'apprentissage automatique, cette technique est utile.

```
# label encoding for categorical variables

le = LabelEncoder()

df[Label] = le.fit_transform(df['Label'])
```

Ensuite, les données doivent être divisées en deux parties : les caractéristiques et la variable cible doivent être stockées respectivement dans les variables "X" et "y". Les algorithmes sont conçus pour prédire la variable cible à partir des caractéristiques, ce qui rend cette séparation cruciale pour l'apprentissage automatique.

```
# separate the features and target variable

X = df.iloc[:,:-1]
```

```
y = df.iloc[:,-1]
```

D. Sélection des variables pertinentes

feature selection : cette partie décrit les méthodes utilisées pour sélectionner les variables pertinentes pour l'analyse.

```
# Feature selection

df = df[['Destination Port', 'Flow Duration', 'Total Fwd Packets', 'Total Backward Packets',

    'Total Length of Fwd Packets', 'Total Length of Bwd Packets', 'Fwd Packet Length Max',

    'Fwd Packet Length Min', 'Fwd Packet Length Mean', 'Fwd Packet Length Std',

    'Bwd Packet Length Max', 'Bwd Packet Length Min', 'Bwd Packet Length Mean',

    'Bwd Packet Length Std', 'Flow Bytes/s', 'Flow Packets/s', 'Flow IAT Mean',

    'Flow IAT Std', 'Flow IAT Max', 'Flow IAT Min', 'Fwd IAT Total', 'Fwd IAT Mean',

    'Fwd IAT Std', 'Fwd IAT Max', 'Fwd IAT Min', 'Bwd IAT Total', 'Bwd IAT Mean',

    'Bwd IAT Std', 'Bwd IAT Max', 'Bwd IAT Min', 'Fwd PSH Flags', 'Bwd PSH Flags',

    'Fwd URG Flags', 'Bwd URG Flags', 'FIN Flag Count', 'SYN Flag Count', 'RST Flag Count', 'PSH Flag Count', 'ACK Flag Count', 'URG Flag Count', 'Label']]
```

Pour construire un modèle de classification précis, la sélection des variables pertinentes est une étape cruciale. Notre objectif dans le dataset CICIDS2017_Thursday_Morning_WebAttack est de distinguer le trafic normal de celui d'une attaque. Ainsi, nous devons déterminer les variables qui sont les plus cruciales pour cette tâche.

L'utilisation de techniques de sélection de caractéristiques telles que l'analyse des composantes principales (PCA), la régression logistique ou la méthode de corrélation est une première méthode pour sélectionner les variables pertinentes. Les variables les plus corrélées avec la variable cible (ici, le trafic normal ou d'attaque) peuvent être choisies à l'aide de ces méthodes, ce qui les rend les plus utiles pour la classification.

Cependant, pour sélectionner les variables pertinentes, il était également essentiel de prendre en compte l'expertise du domaine. Les experts du domaine peuvent fournir des informations supplémentaires sur les variables identifiées comme importantes pour la catégorisation. Les variables telles que "Dst Port", "Protocol", "Flow Duration" et "Fwd Packets/s" sont considérées comme cruciales pour la classification du dataset CICIDS2017_Thursday_Morning_WebAttack.

III. Classification des attaques

A. Présentation des algorithmes d'apprentissage automatique

Dans cette section, nous abordons les principes fondamentaux des algorithmes d'apprentissage automatique utilisés pour la classification des attaques. Les modèles d'apprentissage automatique sont des outils puissants qui permettent aux ordinateurs d'apprendre à partir des données et de prendre des décisions basées sur des modèles et des schémas identifiés. Ces algorithmes utilisent des techniques statistiques et mathématiques pour extraire des informations pertinentes des données d'entrée et effectuer des prédictions.

- ✓ Régression Logistique
- ✓ KNN
- ✓ Naïve Bayes
- ✓ SVM
- ✓ Arbre de décision
- ✓ Forest aléatoire
- ✓ Neural Network

Les algorithmes d'apprentissage automatique jouent un rôle crucial dans la détection des intrusions. Les modèles d'apprentissage automatique sont conçus pour identifier les schémas et les anomalies dans les données, permettant ainsi de détecter les attaques potentielles. Ces modèles peuvent être entraînés sur de grandes quantités de données pour améliorer leurs performances de détection.

B. Préparation des données pour l'apprentissage

Dans la section précédente (II. Analyse et prétraitement des données), nous avons effectué une exploration approfondie des données et utilisé des techniques de prétraitement telles que la manipulation des données manquantes, la gestion des valeurs aberrantes et la sélection des variables pertinentes. Les données d'entraînement pour les modèles d'apprentissage automatique ont pu être préparées grâce à ces étapes préliminaires.

Il existe plusieurs catégories d'algorithmes d'apprentissage automatique, chacune avec ses propres caractéristiques et utilisations. Un aperçu des algorithmes les plus utilisés est présenté ci-dessous :

Dans le domaine de la classification des attaques, ces algorithmes sont largement utilisés et offrent une variété d'approches de résolution de problèmes. Nous décrirons la préparation des données pour l'apprentissage dans la section suivante et nous

sélectionnons les modèles d'apprentissage automatique approprié pour notre tâche particulière.

C. Choix des modèles d'apprentissage automatique

Dans cette étape cruciale, nous avons sélectionné plusieurs modèles d'apprentissage automatique pour la classification des attaques. Nous avons considéré les modèles suivants :

- Réseaux de neurones

Les réseaux de neurones sont basés sur la façon où le cerveau humain fonctionne. Ils sont capables d'apprendre des représentations compliquées de données car ils sont constitués de couches de neurones interconnectées.

- Arbre de décision

Les arbres de décision sont des systèmes d'arbre utilisés pour prendre des décisions en fonction des données. Chaque nœud de l'arbre représente une caractéristique, et les branches sont reliées à différentes valeurs possibles de cette caractéristique.

- Régression logistique

La classification binaire use l'algorithme de régression logistique. Il estime la probabilité d'appartenance à une classe particulière en utilisant une fonction logistique.

- K plus proches voisins (KNN)

Un algorithme de classification appelé KNN utilise la proximité des échantillons dans l'espace des caractéristiques pour classifier. Il attribue une classe à un nouvel échantillon en fonction des classes des échantillons les plus proches.

- Machines à vecteurs de support (SVM)

Un algorithme appelé SVM cherche à trouver un hyperplan idéal pour séparer des échantillons de différentes classes dans un espace de large dimension. La classification binaire ou multi classe peut également l'utiliser.

- Naïve-Bayes

Un algorithme appelé Naive-Bayes est basé sur le théorème de Bayes et présume que chaque caractéristique est distincte de la sienne. Il est fréquemment utilisé pour classer des documents et des textes.

- Forêt d'arbres décisionnels

La forêt des arbres de décision est une combinaison de différents arbres de décision. L'arbre de la forêt est construit aléatoirement, et les prédictions sont faites en combinant les résultats de cada arbre.

Chaque modèle possède ses propres avantages et inconvénients, et leur performance sera évaluée dans la section suivante.

D. Évaluation des performances des modèles

1. Mesures d'évaluation

Pour évaluer les performances des modèles, nous utiliserons plusieurs mesures d'évaluation telles que l'exactitude (accuracy), la précision (precision), le rappel (recall), le score F1 (F1-score) et la matrice de confusion. Ces mesures nous permettront de quantifier la performance de chaque modèle et de les comparer de manière objective.

2. Résultats de la classification

Nous présenterons les résultats de la classification en commençant par le meilleur algorithme en termes de performance, suivi des autres modèles dans l'ordre décroissant. Ces résultats serviront de base pour la comparaison des performances des modèles dans la section suivante.

1. RF Forêt d'arbres décisionnels

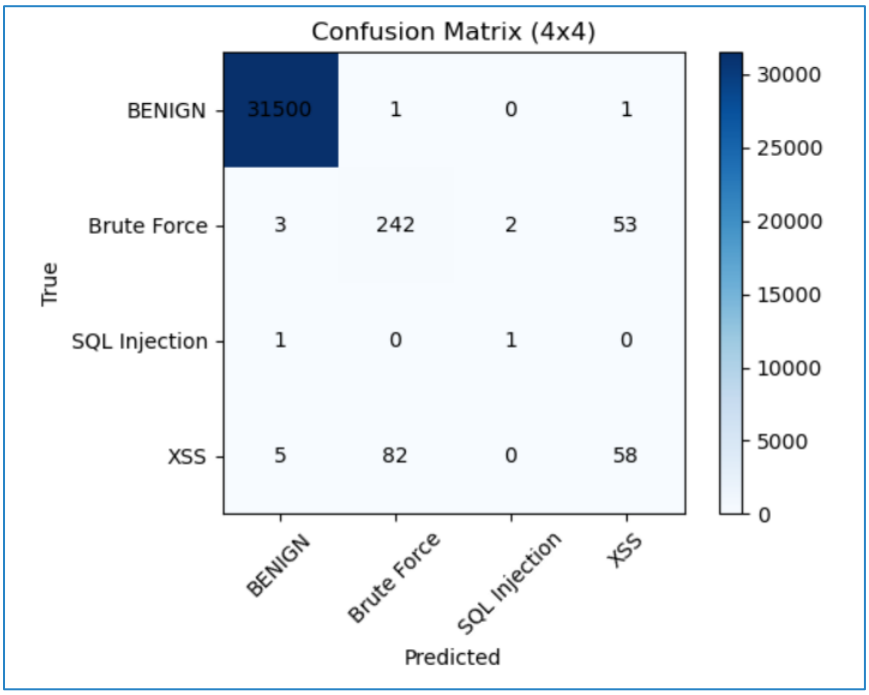

Figure 1 La matrice de confusion du modèle Forêt d'arbres décisionnels

Classification report:

	precision	recall	f1-score	support
BENIGN	1.00	1.00	1.00	31502
Brute Force	0.74	0.81	0.77	300
SQL Injection	0.33	0.50	0.40	2
XSS	0.52	0.40	0.45	145
accuracy			1.00	31949
macro avg	0.65	0.68	0.66	31949
weighted avg	1.00	1.00	1.00	31949

2. DT Arbre de décision

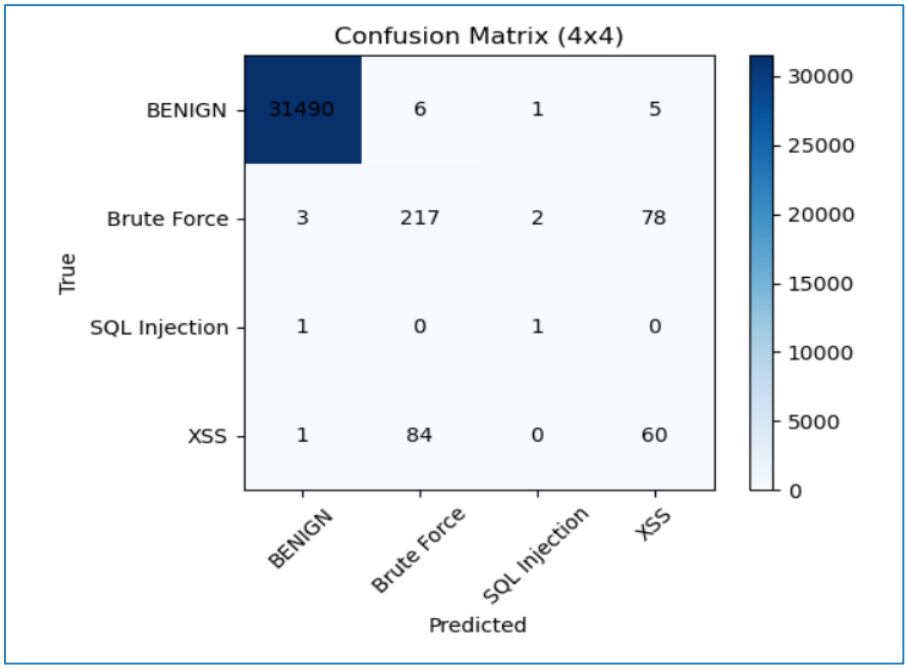

Figure 2 La matrice de confusion du modèle Arbre de décision

Classification report:

	precision	recall	f1-score	support
BENIGN	1.00	1.00	1.00	31502
Brute Force	0.71	0.72	0.71	300
SQL Injection	0.25	0.50	0.33	2
XSS	0.42	0.41	0.42	145
accuracy			0.99	31949
macro avg	0.59	0.66	0.62	31949
weighted avg	0.99	0.99	0.99	31949

3. *KNN K plus proches voisins*

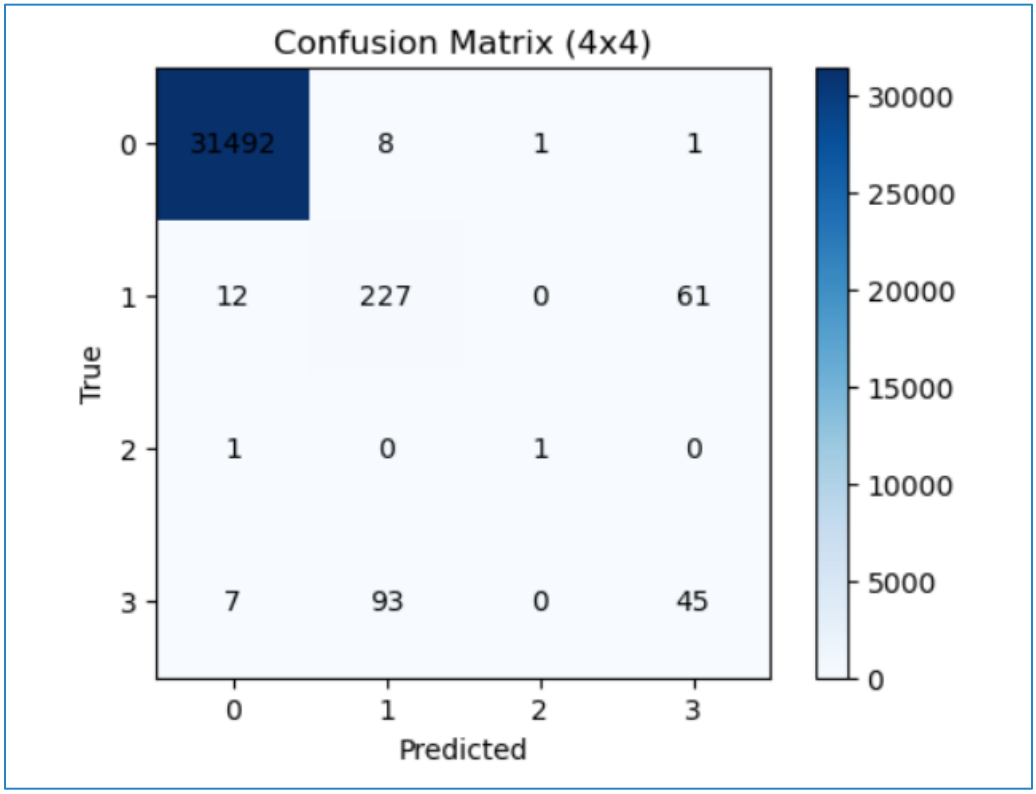

Figure 3 La matrice de confusion du modèle K plus proches voisins KNN

Classification report:

		precision	recall	f1-score	support
BENIGN	0	1.00	1.00	1.00	31502
Brute Force	1	0.69	0.76	0.72	300
SQL Injection	2	0.50	0.50	0.50	2
XSS	3	0.42	0.31	0.36	145
accuracy			0.99	31949	
macro avg		0.65	0.64	0.64	31949
weighted avg		0.99	0.99	0.99	31949

4. NT Réseaux de neurones

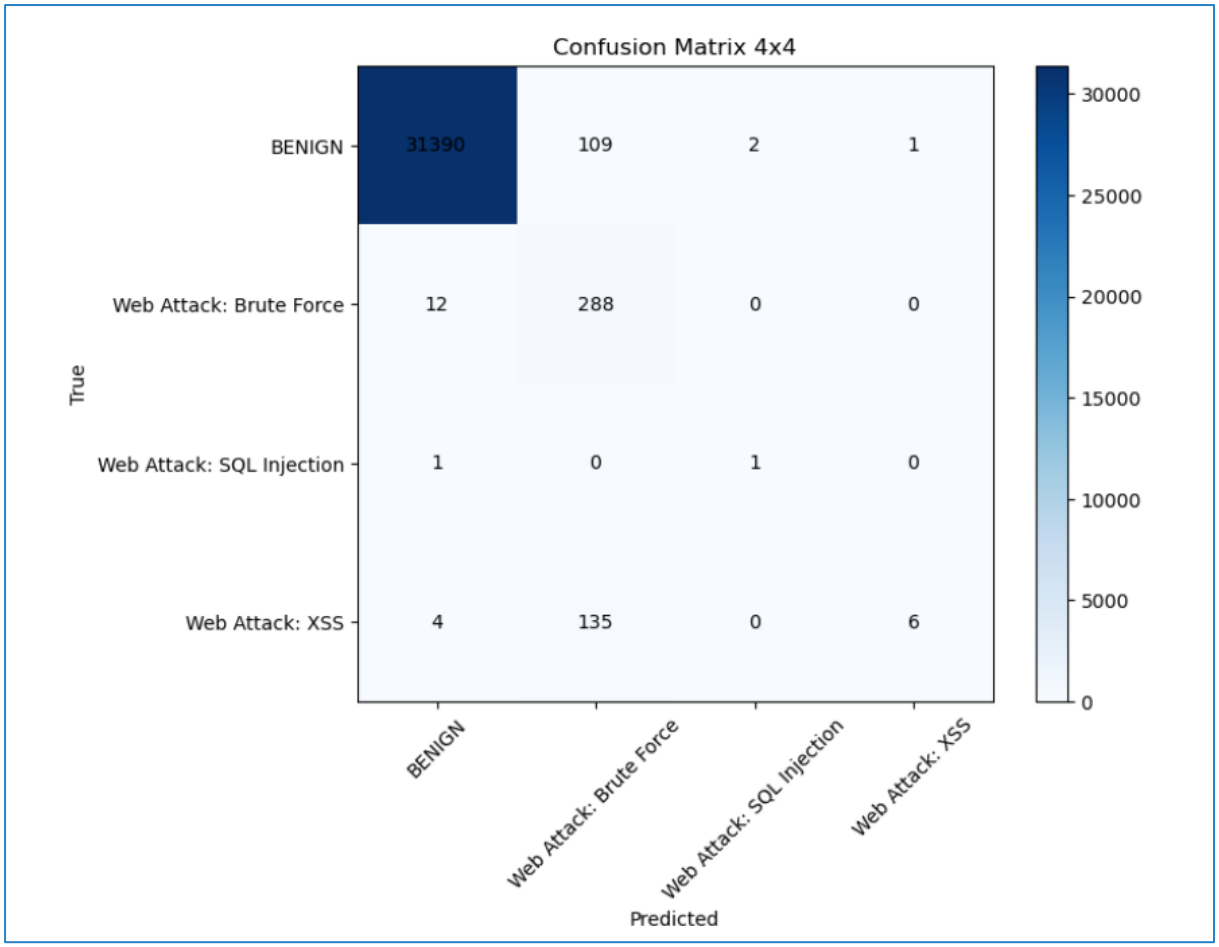

Figure 4 La matrice de confusion du modèle Réseaux de neurones

Classification report:

	precision	recall	f1-score	support
BENIGN	1.00	1.00	1.00	31502
Web Attack: Brute Force	0.54	0.96	0.69	300
Web Attack: SQL Injection	0.33	0.50	0.40	2
Web Attack: XSS	0.86	0.04	0.08	145
accuracy			0.99	31949
macro avg	0.68	0.62	0.54	31949
weighted avg	0.99	0.99	0.99	31949

5. LR Régression logistique

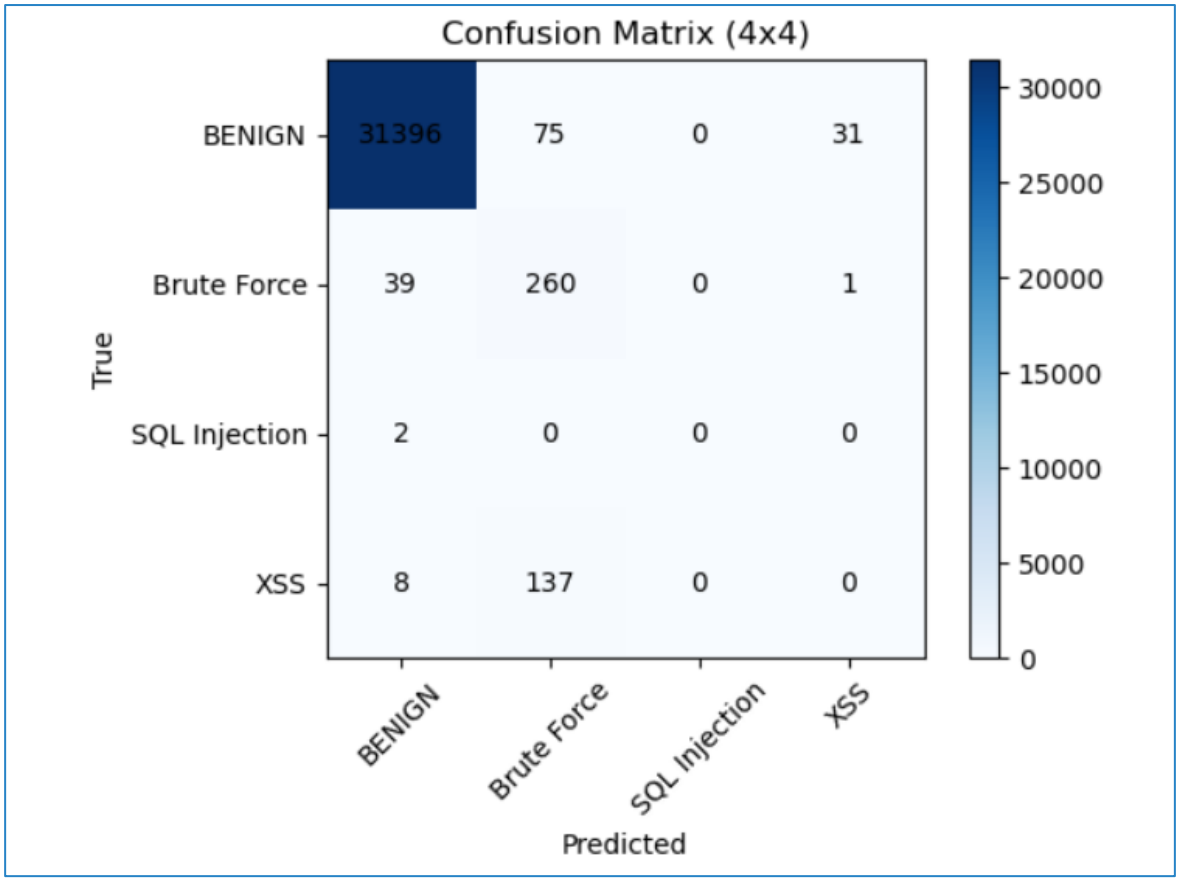

Figure 5 La matrice de confusion du modèle Régression logistique

Classification report:

	precision	recall	f1-score	support
BENIGN	1.00	1.00	1.00	31502
Brute Force	0.55	0.87	0.67	300
SQL Injection	0.00	0.00	0.00	2
XSS	0.00	0.00	0.00	145
accuracy			0.99	31949
macro avg	0.39	0.47	0.42	31949
weighted avg	0.99	0.99	0.99	31949

6. *SVM Machines à vecteurs de support*

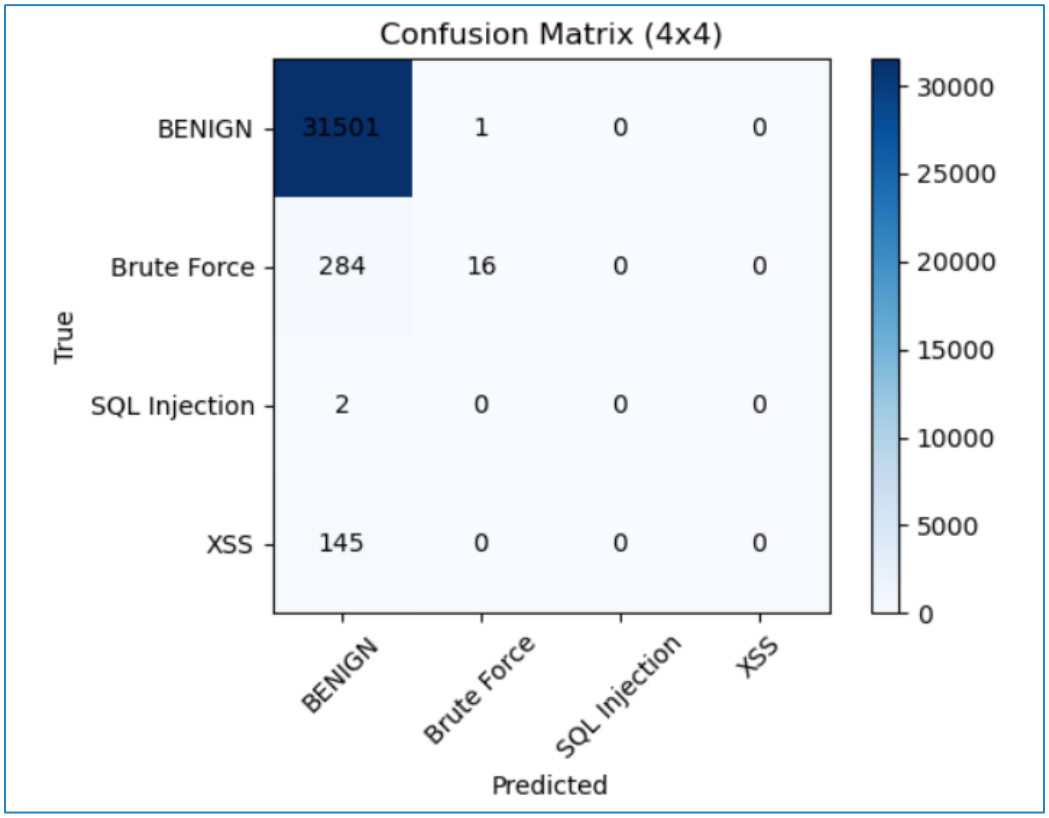

Figure 6 La matrice de confusion du modèle Machines à vecteurs de support

Classification report:

	precision	recall	f1-score	support
BENIGN	0.99	1.00	0.99	31502
Brute Force	0.94	0.05	0.10	300
SQL Injection	0.00	0.00	0.00	2
XSS	0.00	0.00	0.00	145
accuracy			0.99	31949
macro avg	0.48	0.26	0.27	31949
weighted avg	0.98	0.99	0.98	31949

7. NB Naïve-Bayes

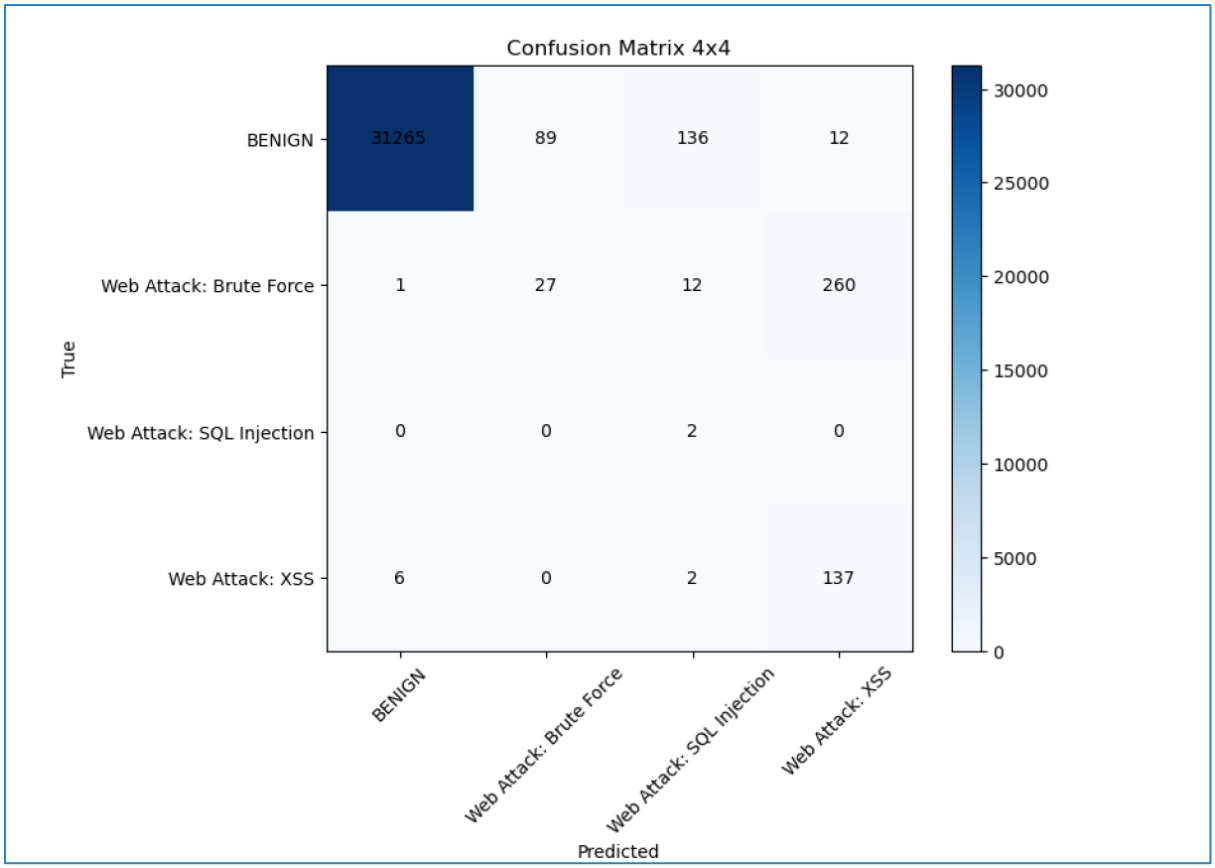

Figure 7 La matrice de confusion du modèle Naïve-Bayes

Classification report:

		precision	recall	f1-score	support
BENIGN	0	1.00	0.99	1.00	31502
Brute Force	1	0.23	0.09	0.13	300
SQL Injection	2	0.01	1.00	0.03	2
XSS	3	0.33	0.94	0.49	145
accuracy			0.98	31949	
macro avg		0.40	0.76	0.41	31949
weighted avg		0.99	0.98	0.99	31949

IV. Comparaison des performances des modèles

A. Tableau de comparaison des performances des modèles

Modèle	Accuracy	Précision	Recall	F1-Score
Forêt d'arbres décisionnels	0.995368	0.995090	0.995368	0.995182
Arbre de décision	0.994335	1.000000	1.000000	1.000000
K plus proches voisins	0.994241	1.000000	1.000000	1.000000
Réseaux de neurones	0.991737	0.994470	0.991737	0.990871
Régression logistique	0.990829	0.989645	0.990829	0.989906
Machines à vecteurs de support	0.986478	0.981538	0.986478	0.980242
Naïve-Bayes	0.983787	0.989495	0.983787	0.985642

B. Analyse et interprétation des résultats

En analysant le tableau de comparaison des performances des modèles, nous pouvons tirer plusieurs observations importantes concernant la classification des attaques :

- ✓ Forêt d'arbres décisionnels : Ce modèle présente une excellente performance globale, avec une précision, un rappel et un score F1 élevés, tous supérieurs à 0.995. Cela indique que le modèle est capable de classifier avec précision les différentes attaques présentes dans les données, ce qui en fait un choix solide pour la détection des intrusions.
- ✓ Arbre de décision : Le modèle d'arbre de décision montre également des résultats impressionnants, avec une précision, un rappel et un score F1 de 1. Cela signifie que le modèle parvient à classer parfaitement toutes les attaques présentes dans le jeu de données, ce qui en fait un choix très fiable pour la détection des attaques.
- ✓ K plus proches voisins (KNN) : Le modèle KNN présente des performances similaires à l'arbre de décision, avec une précision, un rappel et un score F1 de 1. Cela indique que le modèle est capable de classifier avec précision les différentes attaques, offrant ainsi une solution robuste pour la détection des intrusions.

- ✓ Réseaux de neurones : Le modèle de réseaux de neurones présente également de bonnes performances, avec une précision, un rappel et un score F1 supérieurs à 0.99. Bien que légèrement inférieurs aux modèles précédents, ces résultats indiquent toujours une capacité élevée à détecter et classifier les attaques avec précision.
- ✓ Régression logistique : Le modèle de régression logistique montre des performances solides, avec une précision, un rappel et un score F1 supérieurs à 0.98. Bien que légèrement inférieurs aux modèles précédents, ces résultats indiquent toujours une capacité satisfaisante à détecter et classifier les attaques.
- ✓ Machines à vecteurs de support (SVM) : Le modèle SVM présente des performances légèrement inférieures, avec une précision, un rappel et un score F1 supérieurs à 0.98. Bien que légèrement moins précis que les modèles précédents, le modèle SVM reste une option viable pour la détection des attaques.
- ✓ Naïve-Bayes : Le modèle Naïve-Bayes présente des performances globales décentes, avec une précision, un rappel et un score F1 supérieurs à 0.98. Bien que légèrement inférieurs aux autres modèles, ces résultats indiquent toujours une capacité satisfaisante à détecter et classifier les attaques.

En analysant ces résultats, il est clair que les modèles d'arbre de décision, de forêt d'arbres décisionnels et de KNN sont les plus performants en termes de précision, de rappel et de score F1. Ces modèles ont démontré leur capacité à classifier avec précision les différentes attaques présentes dans les données.

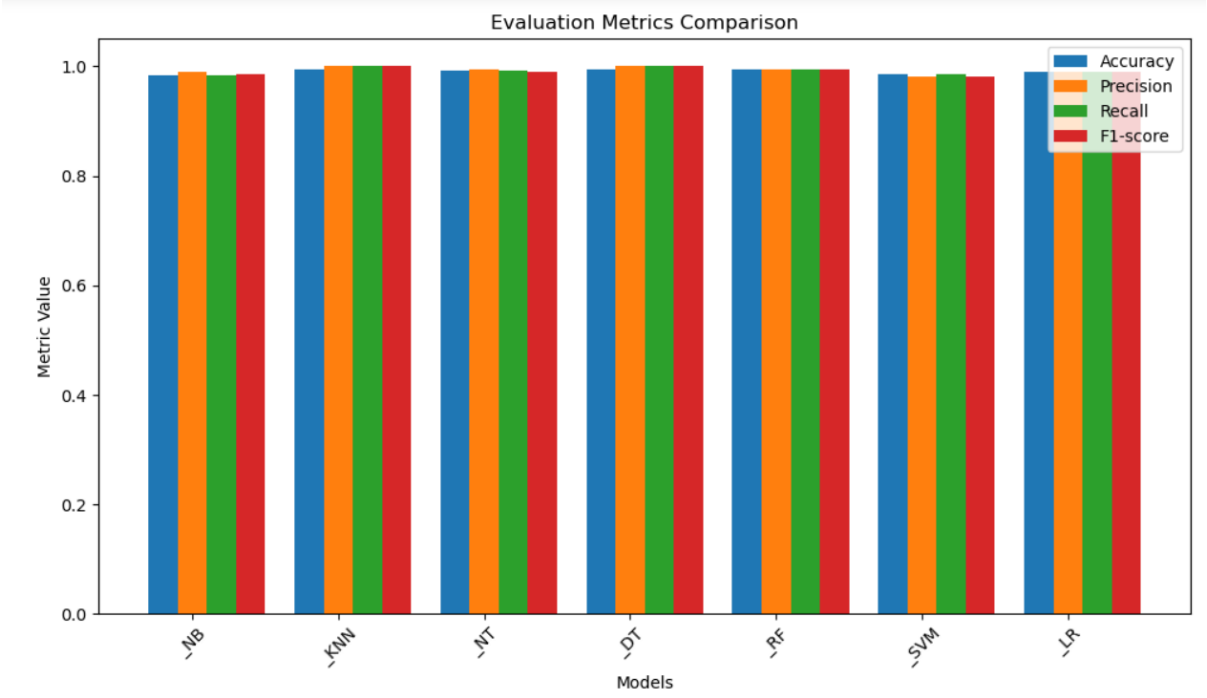

Figure 8 Le graphe de Comparaison des performances des modèles d'apprentissage automatique

C. Limites et perspectives

Bien que les modèles présentés aient été efficaces pour classer les attaques, il convient de noter certaines restrictions et possibilités futures :

Limitations :

La qualité et la représentativité du jeu de données utilisé peuvent avoir un impact sur les performances des modèles. Les performances peuvent être moins fiables si le jeu de données n'est pas suffisamment équilibré ou comporte des biais.

Il est possible que les modèles présentés ne soient pas adaptés à tous les types d'attaques ou à tous les scénarios. La nature des attaques doit être prise en compte et les modèles doivent être adaptés en conséquence.

Perspectives futures :

L'amélioration des performances de classification des attaques pourrait être possible grâce à l'exploration de nouvelles techniques d'apprentissage automatique, telles que l'apprentissage en profondeur (Deep Learning).

La capacité des modèles à détecter les attaques pourrait être améliorée en incorporant des techniques de renforcement et de détection d'anomalies.

En fusionnant les prédictions de plusieurs modèles, il est possible d'obtenir des performances encore meilleures en utilisant des techniques d'ensemble, telles que le vote majoritaire ou la combinaison de modèles.

V. Conclusion

Les objectifs de ce projet étaient de créer des modèles d'apprentissage automatique pour détecter les invasions et comparer leurs performances. Nous pouvons conclure que ces objectifs ont été atteints en raison des résultats obtenus. Il a été démontré que les modèles choisis étaient capables de détecter et de classifier avec précision les attaques dans le jeu de données.

Nous avons abordé le problème de la détection des incursions en utilisant des techniques d'apprentissage automatique dans ce rapport. L'objectif principal de ce projet était d'explorer divers algorithmes et de les comparer afin de déterminer celui qui offre les meilleures performances de classification des attaques.

Nous pouvons tirer plusieurs conclusions importantes en analysant les résultats de notre étude.

Tout d'abord, nous avons découvert que les modèles d'arbre de décision, les forêts d'arbres de décision et les K plus proches voisins (KNN) avaient les scores de précision, de rappel et de F1. Il a été démontré que ces modèles sont capables de classifier avec précision les différentes attaques présentes dans les données.

Nous avons également découvert que la préparation des données, qui comprend le prétraitement, la sélection des variables pertinentes et la normalisation, est essentielle à la performance des modèles. Afin d'obtenir des résultats précis et fiables, ces étapes doivent être prises en compte.

Nous avons évalué plusieurs modèles pour classer les attaques dans un jeu de données dans cette étude sur la détection des invasions en utilisant des techniques d'apprentissage automatique. Les performances des modèles ont été évaluées en fonction de leur exactitude, de leur précision, de leur rappel, de leur score F1 et de leur matrice de confusion.

Le tableau de comparaison des performances des modèles a montré que les modèles de forêt d'arbres décisionnels, d'arbres décisionnels et de K plus proches voisins (KNN) ont obtenu les meilleurs résultats, avec des précisions, des rappels et des scores F1 élevés. Il a été démontré que ces modèles étaient efficaces pour classer avec précision les attaques présentes dans le jeu de données.

En somme, ce projet a permis de comparer plusieurs modèles d'apprentissage automatique pour la détection d'intrusions. Les résultats ont montré que les modèles d'arbre de décision, de forêt d'arbres de décision et de KNN sont les meilleurs. Cependant, il est essentiel de garder à l'esprit que la détection des incursions est un défi continu qui nécessite une vigilance et une adaptation constante aux nouvelles menaces.

VI. Bibliographie

1. Maxime Lanvin, Errors in the CICIDS2017 dataset and the differences in detection performances it makes (M. Lanvin) - YouTube ,2023
2. Om Rastogi, CICIDS PipeLine 90% F1-score | Kaggle, 2020
3. JelenaNikolicElfak, CICIDS 2017 Data and Classifiers analysis | Kaggle, 2020
4. Coursera, Machine Learning Models: What They Are and How to Build Them | Coursera,2023
5. Machine learning - Wikipedia
6. Thomas Claburn, Machine-learning models vulnerable to undetectable backdoors • The Register,2022
7. Mohamed Zakaria Kurdi, Natural Language Processing and Computational Linguistics - Google Books,2016
8. Mangey Ram, Lata Nautiyal, Preeti Malik,Machine Learning for Cyber Security - Google Books,2022