1. Collecte de données

La base de données utilisée dans cette étude a été collectée à partir de la plateforme Kaggle[1]. Elle contient 345738 URLs bénignes (sites web légitimes) et 104438 URLs malveillantes (sites web de phishing), qui fait un total des 450176 observations. Cette base est également composée de quatre colonnes (features) à savoir :

1. **Unnamed: 0** : Cette variable représente les valeurs d'index pour les données ;
2. **Url** : Cette variable représente les différentes URLs ;
3. **Label** : Cette variable indique si l'URL est bénigne (URL légitime) ou malveillante (URL de phishing ou malicious)
4. **Result** : Cette variable est la version binaire de la variable label, 0 correspond à une URL bénigne et 1 correspond à une URL de phishing. Il s'agit de la variable cible (target).

Nous mettons un petit aperçu de cette base de données ci-dessous :

Tableau 1 : base de données

	Unnamed: 0	url	label	result
0	0	https://www.google.com	benign	0
1	1	https://www.youtube.com	benign	0
2	2	https://www.facebook.com	benign	0
3	3	https://www.baidu.com	benign	0
4	4	https://www.wikipedia.org	benign	0
...	...	...	...	...
450171	450171	http://ecct-it.com/docmmmnn/aptgd/index.php	malicious	1
450172	450172	http://faboleena.com/js/infortis/jquery/plugin...	malicious	1
450173	450173	http://faboleena.com/js/infortis/jquery/plugin...	malicious	1
450174	450174	http://atualizapj.com/	malicious	1
450175	450175	http://writeassociate.com/test/Portal/inicio/I...	malicious	1

450176 rows × 4 columns

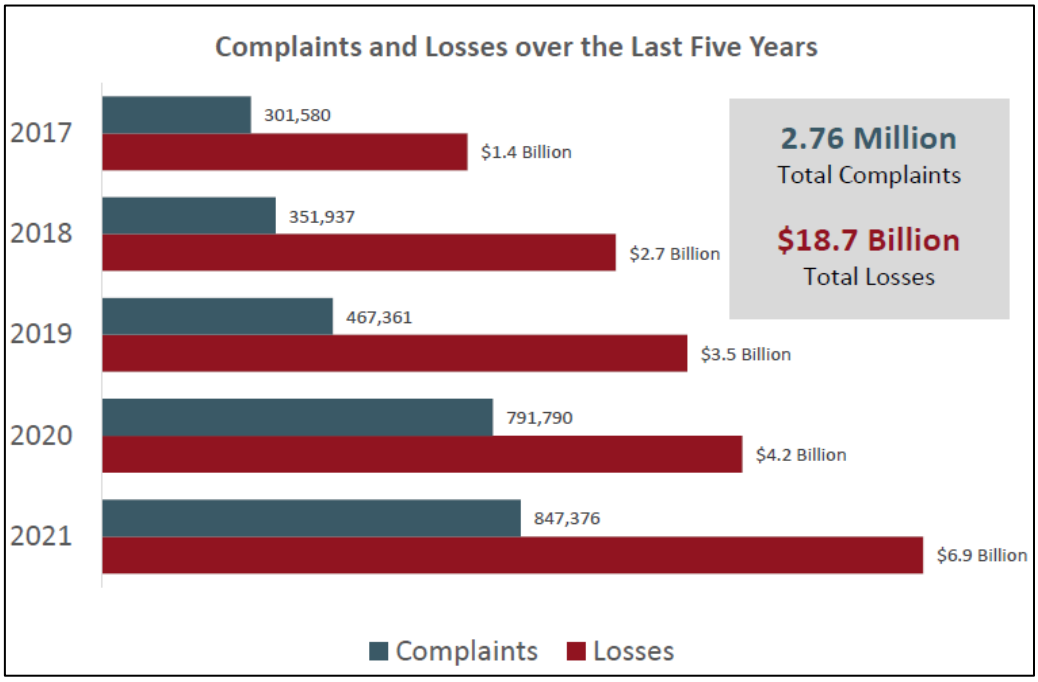
A partir de cette base de données, nous cherchons à implémenter plusieurs modèles d'apprentissage automatique afin de trouver le modèle le plus performant et le plus adapté pour classifier les URLs bénignes et malveillantes avec une grande précision. *La **finalité** de l'étude c'est de pouvoir détecter automatiquement les sites Web de phishing et éviter l'extorsion voire le vol des informations personnelles ou bancaires des utilisateurs*. Le vol de ces informations confidentielles prend le nom de **phishing** (ou **hameçonnage** en français) et se fait

[1] Cliquez ici pour visualiser cette base de données

principalement par **e-mails**. En effet, le homeçonneur utilise des astuces d'ingénierie sociale pour obtenir ces informations confidentielles de la part de leurs victimes. Il conçoit des sites de phishing qui ressemblent à des sites légitimes tels que : Amazon, BNP Paribas, la Poste, la CAF, Site d'un hôtel (Ibis par exemple), etc.

Selon le rapport sur la criminalité sur internet publié par *Internet Crime Complaint Center (IC3)* du *Federal Bureau of Investigation* des États-Unis, en 2019, l'IC3 a reçu un total de 467361 des plaintes avec des pertes dépassants 3,5 milliards de dollars. Parmi les types de crimes les plus couramment signalés on trouve entre autres l'hameçonnage, le vishing, le smishing, le pharming, etc.[2] Ces plaintes ont passé de 301580 en 2017 à 847376 en 2021 avec un coût de 1,4 milliards de dollars à 6,9 milliards de dollars[3].

Graphique 1 : d'évolution de coûts engendrés par les arnaques en lignes



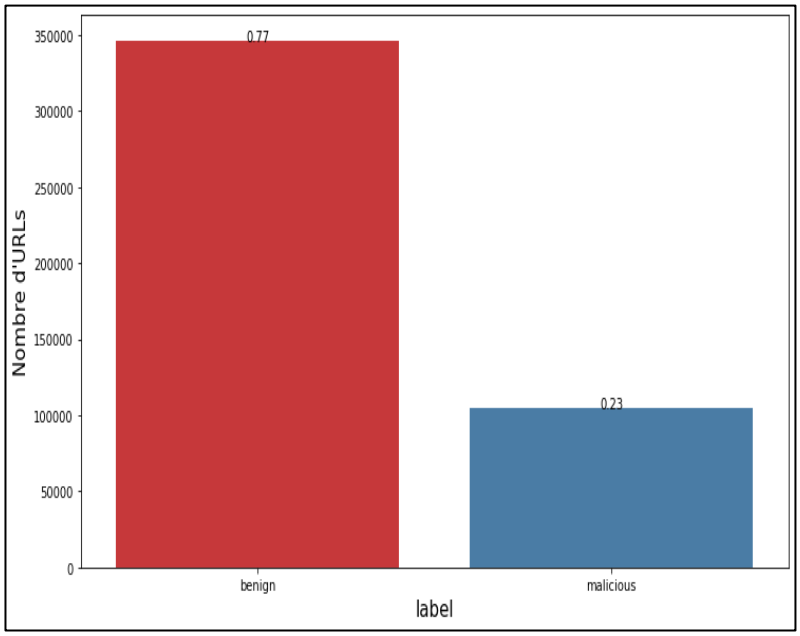
Source : FEDERAL BUREAU of INVESTIGATION, INTERNET CRIME Report 2021

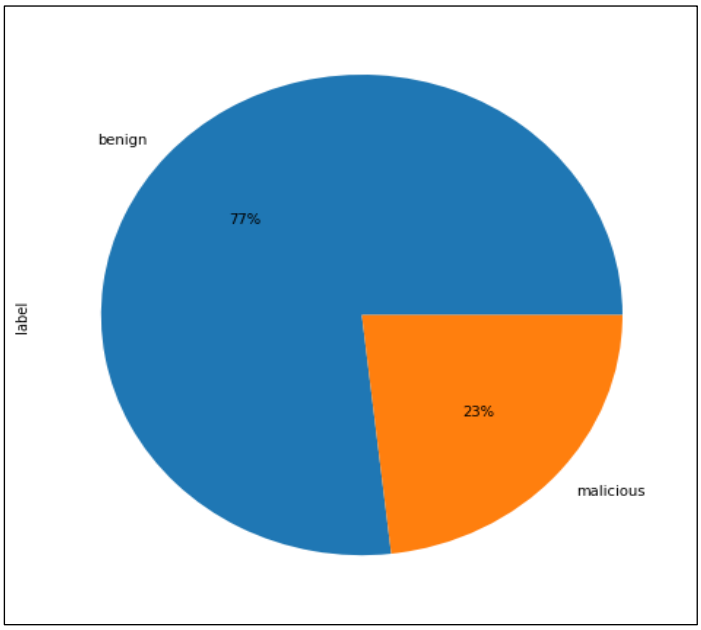
Compte tenu de ces augmentations de ces arnaques et l'évolution rapide d'internet et des commerces en ligne, nous imaginons l'importance et la vigilance que ce domaine requiert de la part de professionnels et des particuliers. Dans le cadre de cette étude, nous utilisons les algorithmes d'apprentissage automatique, pour lutter contre ces attaques en lignes.

Après ce bref aperçu sur les arnaques en lignes, nous allons visualiser notre variable cible (result), ci-dessous, pour vérifier si les proportions de deux catégories de classes qui la composent sont plus ou moins équilibrées.

[2] FEDERAL BUREAU of INVESTIGATION, INTERNET CRIME COMPLAINT CENTER (IC3), Report 2019. P.3 : (cliquer ici pour télécharger le PDF associé)
[3] FEDERAL BUREAU of INVESTIGATION, INTERNET CRIME COMPLAINT CENTER (IC3) Report 2021. P.7. (cliquer ici pour télécharger le PDF associé)

Graphique 2 : Les proportions de catégorie de classes de la variable target (label)





En observant le barplot et le graphique en secteurs de notre jeu des données, nous constatons que les proportions des classes du jeu de données sont déséquilibrées[4] (77% pour la classe **benign** et 23% pour la classe **malicious**), ce qui pourrait poser un problème, car nous risquons de se retrouver avec un algorithme de classification peu intelligent et qui va toujours prédire la classe dominante. Or, dans notre cas, c'est la classe minoritaire qui nous intéresse vraiment.

Nous pouvons traiter ce problème en utilisant l'une de méthodes[5] suivantes :

- **Undersampling** : Il s'agit d'enlever des données de la classe majoritaire ;
- **Oversampling** : il s'agit de rajouter des nouvelles données dans la classe minoritaire.

Dans cette étude, nous optons pour la première méthode, car nous estimons qu'il est plus sûr de supprimer des observations de la classe majoritaire que d'utiliser de techniques de rééchantillonnage qui pourraient fausser les données de notre variable URL. A notre avis le rééchantillonnage de données dépend généralement de types de données dont on dispose. Par ailleurs, dans le cadre de cette étude, nous gardons les données d'origine et les données après undersampling. Cela nous permettra de faire une comparaison à la suite de performance de nos modèles d'apprentissage.

Il faut juste noter qu'en utilisant la première méthode nous risquons également de supprimer des informations utiles. Donc, la deuxième méthode est aussi importante si bien sûr on arrive à trouver des données cohérentes. Nous avons ainsi opté pour la première méthode pour ne pas nous compliquer la tâche, vue que nous avons de données textuelles brutes, mais pour un travail

[4] Pour avoir plus de détails sur ce problème de déséquilibre veuillez cliquer ici et ici
[5] Différents algorithmes de rééquilibrages de classes d'un data set sont détaillés ici

ultérieur pourquoi pas trouver d'autres données. Ainsi, nous pourrions tester en même temps les deux méthodes.

2. Nettoyage de données

Tout travail en machine Learning doit avoir une partie nettoyage et préparation de données, car des données propres et bien structurées facilitent l'implémentation de l'algorithme d'apprentissage et surtout sa performance. En effet, nous allons pour cela, vérifier et supprimer les valeurs manquantes, les colonnes inutiles, colmater les espaces dans le texte de notre variable URL, vérifier et convertir si nécessaire les types de variables. Et pour finir nous équilibrons les deux catégories de classes de notre variable cible.

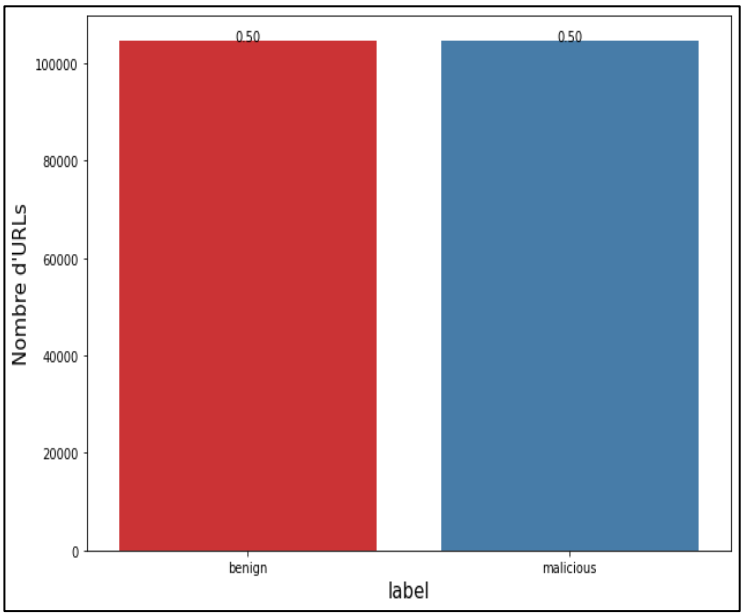
Après le nettoyage et le sous-échantillonnage de nos données, nous réaffichons ci-dessous notre base de données.

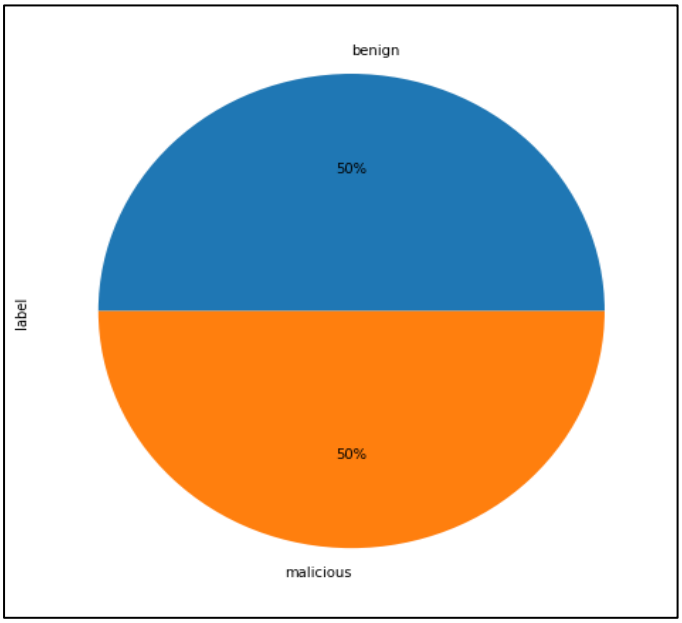
Tableau 2 : base de données après undersampling

	Unnamed: 0	url	label	result
209719	209719	https://www.heraldica.org/topics/france/apanag...	benign	0
99262	99262	https://www.softkill.tumblr.com/	benign	0
301507	301507	https://www.teslamotors.com/	benign	0
532	532	https://www.t.me	benign	0
109375	109375	https://www.tvlistings.zap2it.com/tvlistings/Z...	benign	0
...	...	...	...	...
450171	450171	http://ecct-it.com/docmmmnn/aptgd/index.php	malicious	1
450172	450172	http://faboleena.com/js/infortis/jquery/plugin...	malicious	1
450173	450173	http://faboleena.com/js/infortis/jquery/plugin...	malicious	1
450174	450174	http://atualizapj.com/	malicious	1
450175	450175	http://writeassociate.com/test/Portal/inicio/I...	malicious	1

208876 rows × 4 columns

Graphique 3 : visualisation de catégorie de classes équilibrées





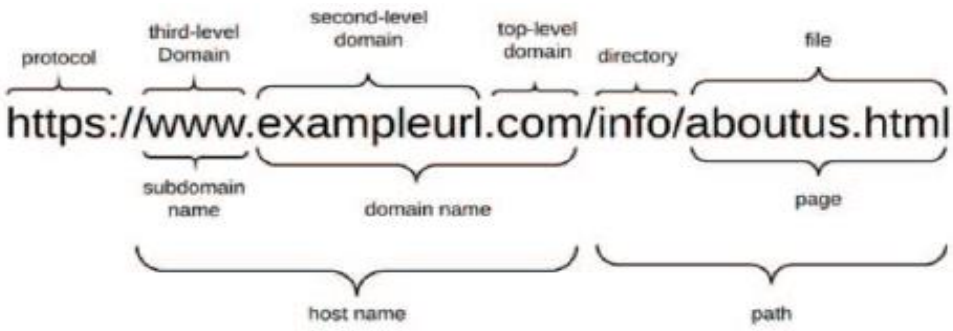
Comme nous le constatons sur les graphiques ci-dessus, nous avons opté pour un sous-échantillonnage qui nous a donnée 50% de catégorie 'benign' et 50% de catégorie 'malicious'. A ce niveau, nous pouvons également jouer sur les proportions de catégories de classes pour constater des améliorations des performances de nos modèles, car la méthode Undersampling que nous avons utilisé à pour limite la suppression des informations de la classe majoritaire qui pourraient être utiles.

3. Features Engineering / Data Preproccessing (prétraitement des données)

La technique de features Engineering fait partie du processus de machine learning permettant de prétraiter les données afin de faciliter leur utilisation par les algorithmes de machine learning. Il y a plusieurs méthodes utilisées en features engineering parmi lesquelles l'**extraction de caractéristiques**[6] (features extraction) et la **sélection de caractéristiques**[7] (features selection). Nous utilisons la première technique pour extraire de nouvelles caractéristiques à partir de notre variable URL et la deuxième technique pour sélectionner les caractéristiques les plus pertinentes afin de faire de comparaisons de performance de modèles avec toutes les caractéristiques d'une part et les caractéristiques sélectionnées en fonction de leur performance d'autre part.

Les caractéristiques extraites de l'URL servent à décortiquer la structure de l'URL afin de permettre à l'algorithme d'apprentissage de mieux classifier les URLs légitimes et malveillantes. C'est grâce à ces caractéristiques qui seront extraites que nous pourrions identifier les URLS de phishing. Donc, sans plus tarder nous allons présenter une figure squelettique de la structure d'une URL bénigne et par la suite les différentes caractéristiques qui seront extraites de l'URL[8].

La structure d'une URL bénigne ou normale



La figure ci-dessous décompose la structure d'une URL normale. En effet, dans une URL normale ou légitime, nous avons le **protocole** (protocol), qui permet d'accéder à la page Web. Et le **nom du domaine** (domain name)**,** qui permet d'identifier le serveur qui héberge la page Web. Ce nom du domaine doit être enregistrée auprès d'un registraire de nom de domaine[9]. Ce

[6] https://towardsdatascience.com/extracting-feature-vectors-from-url-strings-for-malicious-url-detection-cbafc24737a

[7] https://scikit-learn.org/stable/modules/feature_selection.html

[8] Pour avoir plus des détails sur l'ingénierie des caractéristiques veuillez cliquer ici

[9] Pour plus de détails sur le registraire de nom du domaine veuillez cliquer ici

nom du domaine comprend le **domaine du second niveau** (second-level domain) et le **domaine du premier niveau** (top-level domain (TLD)). Quant au **nom de l'hôte** (host name), il est composé du **nom de sous-domaine** (subdomain name) et du **nom de domaine** (domain name). La dernière partie de l'URL concerne le chemin (**path**). Vue la forme structurale de cette URL, un attaquant (homeçonneur) peut modifier de différentes manières la structure de l'URL pour tromper ses victimes. C'est pourquoi nous allons analyser la structure des URLs et soustraire des caractéristiques permettant de distinguer ces deux types d'URLs.

Nous analyserons les points ci-dessous afin d'extraire les différentes caractéristiques des URLs. Cette partie d'extraction des caractéristiques est inspirée des articles de recherches dans le domaine de phishing et plus particulièrement du notebook de Siddharth Kumar sur Kaggle[10] et de Katie Sylvia sur GitHub[11]. Il faut savoir que la plupart de travaux de recherches que nous avons consultés dans ce domaine utilisent les mêmes méthodes d'extraction des caractéristiques. Cette partie d'extraction des caractéristiques va se baser sur la bibliothèque **urllib.**

Nous commençons d'abord par extraire les caractéristiques liées aux signes de ponctuation :

1) Les caractéristiques de ponctuation

- ✓ **Compter le nombre de '@' (count@)**

Cette caractéristique compte le nombre du symbole @ dans l'URL. L'existence du symbole @ dans l'URL signifie que la partie gauche de l'URL n'est pas prise en compte, alors que la partie droite du symbole est l'URL réelle. Les homeçonneurs utilisent ce symbole pour masquer la partie suspecte de l'URL[12], par exemple, une URL comme 'http://www.google.com@phishing.com' dirige en fait la victime vers 'phishing.com' plutôt que vers la vraie page Google[13].

- ✓ **Compter le nombre de '?' dans l'URL (count?)**

Cette caractéristique compte le nombre du symbole '?' dans une URL. Il faut savoir qu'en général dans une URL malveillante on trouve beaucoup de symboles d'interrogation[14].

- ✓ **Compter le nombre de '% dans l'URL' (count%)**

Cette caractéristique compte le nombre de symbole '%' dans une URL. Il faut savoir que souvent dans une URL illégitime on trouve beaucoup de symboles pourcentages[15].

---

[10] Pour consulter ce notebook, veillez cliquer ici.
[11] Pour consulter ce GitHub veuillez cliquer ici
[12] Xiang, G., al. 2011. CANTINA+ : A feature-rich machine learning framework for detecting phishing Web sites. ACM Trans. Inf. Syst. Secur. 14, 2, Article 21 (September 2011). P.8
[13] Sushil Jajodia & al. Adaptive Autonomous Secure Cyber Systems. Springer Cham, ISBN 978-3-030-33431-4. DOI : https://doi.org/10.1007/978-3-030-33432-1. P.260
[14] https://github.com/ksylvia16/Phishing-URL-Detection#:~:text=Quantity%20of%20(%3F)%20in%20URL
[15] https://github.com/ksylvia16/Phishing-URL-Detection#:~:text=Quantity%20of%20(%25)%20in%20URL

- ✓ **Compter le nombre de dièses # dans l'URL (count#)**

En général, dans une URL légitime on trouve qu'un seul dièse, plusieurs dièses dans une URL indique une URL suspecte. Cette caractéristique compte le nombre de dièses dans une URL[16].

- ✓ **Compter le nombre de '.' (count.)**

Cette caractéristique compte le nombre de points dans les URLs. En effet, les URLs de phishings ont souvent plusieurs points dans leur structure que les URLs légitimes. Autrement dit, plus le nombre de points augmentent plus l'URL tendent à être illégitime[17]. Par exemple http://www.facebook.com peut être représenté par http://www.face.book.com[18].

- ✓ **Nombre de sous-domaine (count-www)**

Cette caractéristique compte le nombre de sous-domaine (www) dans l'URL. D'après Sushil Jajodia et al, si l'URL a plus d'un sous-domaine, alors c'est une URL de phishing[19].

- ✓ **Compter le nombre de '-' (count-)**

Cette caractéristique compte le nombre de tirets d'union dans l'URL. Le tiret est souvent utilisé par les homeçonneurs pour ajouter un préfixe ou suffixe au nom du domaine afin que les utilisateurs aient l'impression d'être sur une page légitime[20].

- ✓ **Profondeur de l'URL (count slash single)**

Cette caractéristique basée sur la profondeur de l'URL, ne fait que calculer en fait le nombre de slash ('/') dans l'URL. En effet, cette feature détermine le nombre de sous-pages dans une URL donnée.

2) Le nombre de 'http ou 'https' dans l'URL

- ✓ **Compter le nombre de 'http'**

Cette caractéristique compte le nombre des 'http' dans de l'URL. Les homeçonneurs, pour tromper leur victime, utilisent par fois un "http" dans la partie host name de l'URL.

- ✓ **Compter le nombre de 'https'**

Cette caractéristique compte le nombre des 'http' dans de l'URL. Les homeçonneurs, pour tromper leur victime, utilisent par fois un "https"[21] dans la partie host name de l'URL.

---

[16] https://github.com/ksylvia16/Phishing-URL-Detection#:~:text=Quantity%20of%20(%23)%20in%20URL

[17] Xiang, G., al. 2011. CANTINA+ : A feature-rich machine learning framework for detecting phishing Web sites. ACM Trans. Inf. Syst. Secur. 14, 2, Article 21 (September 2011). P.8

[18] Brij B. Gupta & al. A novel approach for phishing URLs detection using lexical based machine learning in a real-time environment, Computer Communications, Volume 175, 2021, ISSN 0140-3664. DOI : https://doi.org/10.1016/j.comcom.2021.04.023. P.51

[19] Sushil Jajodia & al. Adaptive Autonomous Secure Cyber Systems. Springer Cham, ISBN 978-3-030-33431-4. DOI : https://doi.org/10.1007/978-3-030-33432-1. P.260

[20] Atharva Deshpande, al. (2021). Detection of Phishing Websites using Machine Learning. International Journal of Engineering Research & Technology. Vol. 10 Issue 05, May-2021. P.431

[21] Dans cet exemple, HTTPS est utilisé, alors qu'il existe également d'autres protocoles disponibles tels que HTTP et FTP.

3) Caractéristiques basant sur la longueur

- ✓ **La caractéristique basant sur la longueur de l'URL (url_length)**

La longueur de l'URL permet à l'attaquant de masquer la partie douteuse de l'URL. Donc, via cette caractéristique nous allons compter le nombre de caractères composant une URL[22].

- ✓ **La caractéristique basant sur la longueur du nom d'hôte (hostname_length)**

Comme indiqué ci-haut la longueur d'une URL donnée se compose de trois parties telles que le **protocole** (http/https), le **nom d'hôte** (host name) et le **chemin** (path). Alors, nous extrayons tout d'abord le nom d'hôte de l'URL et nous calculons sa longueur. La décomposition de l'URL se fera via le package '**urlparse**' qui est intégré dans le module '**urllib.parse**'. Pour récupérer le '**host name**' on utilise la structure '**netloc'**.

- ✓ **La caractéristique basant sur la longueur du chemin (path_length)**

Cette caractéristique compte le nombre de caractères composant le path afin de distinguer les URLs de phishing et les URLs légitimes. On extrait le path de l'URL et on compte les caractères qui le composent.

4) Caractéristiques binaires

- ✓ **Https dans la partie nom du domaine**

Cette caractéristique indique la présence ou nom du protocole https dans la partie nom du domaine de l'URL. Le homeçonneur utilise ce protocole dans le nom du domaine pour tromper les victimes[23].

- ✓ **Http dans la partie nom du domaine**

Cette caractéristique indique la présence ou nom du protocole https dans la partie nom du domaine de l'URL. Le homeçonneur utilise ce protocole dans le nom du domaine pour tromper les victimes.

- ✓ **Utilisation de IP**

Généralement les homeçonneurs utilisent une adresse IP ou son code hexadécimal à la place du nom de domaine de l'URL. Tandis que les sites Web légitimes utilisent normalement des noms de domaine au lieu d'une adresse IP. Alors, si le nom de domaine dans l'URL est une adresse IP, la valeur de la caractéristique est '-1', sinon sa valeur est '1'. Bref, - 1 correspond à une

[22] Certaines études fixent un nombre total sur lequel se base pour distinguer les URLs légitimes des URLs illégitimes en créant une variable binaire. Par exemple dans l'étude [18], l'auteur explique que toutes les URLs dont la longueur dépasse 54 caractères sont classées comme des URLs de phishing. Dans notre cas, nous faisons un simple comptage de différents caractères de l'URL.

[23] Atharva Deshpande, Omkar Pedamkar, Nachiket Chaudhary, Dr. Swapna Borde, 2021, Detection of Phishing Websites using Machine Learning, INTERNATIONAL JOURNAL OF ENGINEERING RESEARCH & TECHNOLOGY (IJERT) Volume 10, Issue 05 (May 2021). P.431-432. (Here)

URL de phishing et 1 à une URL légitime[24]. Par exemple une URL avec une adresse IP ou hexadécimale peut s'écrire comme suit : http://125.98.3.123/fake.html ou http://0x58.0xCC.0xCA.0x62/2/paypal.ca/index.html[25]

- ✓ **Redirection**

L'existence de double Slash (//) dans le chemin de l'URL signifie que l'utilisateur sera redirigé vers un autre site Web[26]. Ce Slash dans le cas d'une URL légitime occupe la 6éme ou 7éme position (après http ou https). Alors, si la position du Slash dans l'URL est supérieure ou égale à 8, la valeur de cette caractéristique est -1 sinon sa valeur est 1[27].

Tableau 3 : Résumé des caractéristiques extraites des URLs

Features	Type	Dataset	Description
url_length	int	urldata.csv	Compter les caractères de l'URL
hostname_length	int	urldata.csv	Compter les caractères du hostname
path_length	int	urldata.csv	Compter les caractères du path
count@	int	urldata.csv	Compter les '@' dans l'URL
count ?	int	urldata.csv	Compter les '?' dans l'URL
count%	int	urldata.csv	Compter les '%' dans l'URL
count#	int	urldata.csv	Compter les '#' dans l'URL
count.	int	urldata.csv	Compter les '.' dans l'URL
count-www	int	urldata.csv	Compter les 'www' dans l'URL
count-	int	Urldata.csv	Compter les '-' dans l'URL
count/	int	urldata.csv	Compter les '/' dans l'URL
count_https	int	urldata.csv	Compter les 'http' dans l'URL
count_https	int	urldata.csv	Compter les 'https' dans l'URL
https_Domain	int	urldata.csv	Vérifier la présence du 'https' dans la partie 'domain name'
http_Domain	int	urldata.csv	Vérifier la présence du 'http' dans la partie 'domain name'
use_ip	int	urldata.csv	Vérifier la présence de l'adresse 'IP' dans la partie 'domain name'
redirection	int	urldata.csv	Vérifier la position du double slash (//) dans l'URL

[24] Santhana Lakshmi V, Vijaya MS, a. Efficient prediction of phishing websites using supervised learning algorithms. ELSEVIER, International Conference on Communication Technology and System Design 2011. P.801

[25] ADEDIRAN ADENIYI GOODNESS. Detection of phishing websites using machine learning. Computer Science Programme, College of Computing and Communication Studies, Bowen University, iwo, osun state, Nigeria. 2021. p.31. P.19 (cliquer ici)

[26] Atharva Deshpande, al. (2021). Detection of Phishing Websites using Machine Learning. International Journal of Engineering Research & Technology. Vol. 10 Issue 05, May-2021. P.431

[27] Ekta Gandotra & Deepak Gupta (2020) : Improving Spoofed Website Detection Using Machine Learning, Cybernetics and Systems, DOI. P.14

4. Data visualization (Visualisation des données)

Dans cette partie nous allons effectuer quelques visualisations de nos données pour avoir une idée sur leur évolution, corrélation, distribution, etc. Il faut savoir qu'ici nous visualisons que les données d'origine, les données sous-échantillonnées seront utilisées que pour comparer les performances de modèles d'apprentissage automatique**.**

Nous commençons cette partie par visualiser le tableau de statistiques descriptives :

Tableau 4 : Statistiques descriptives

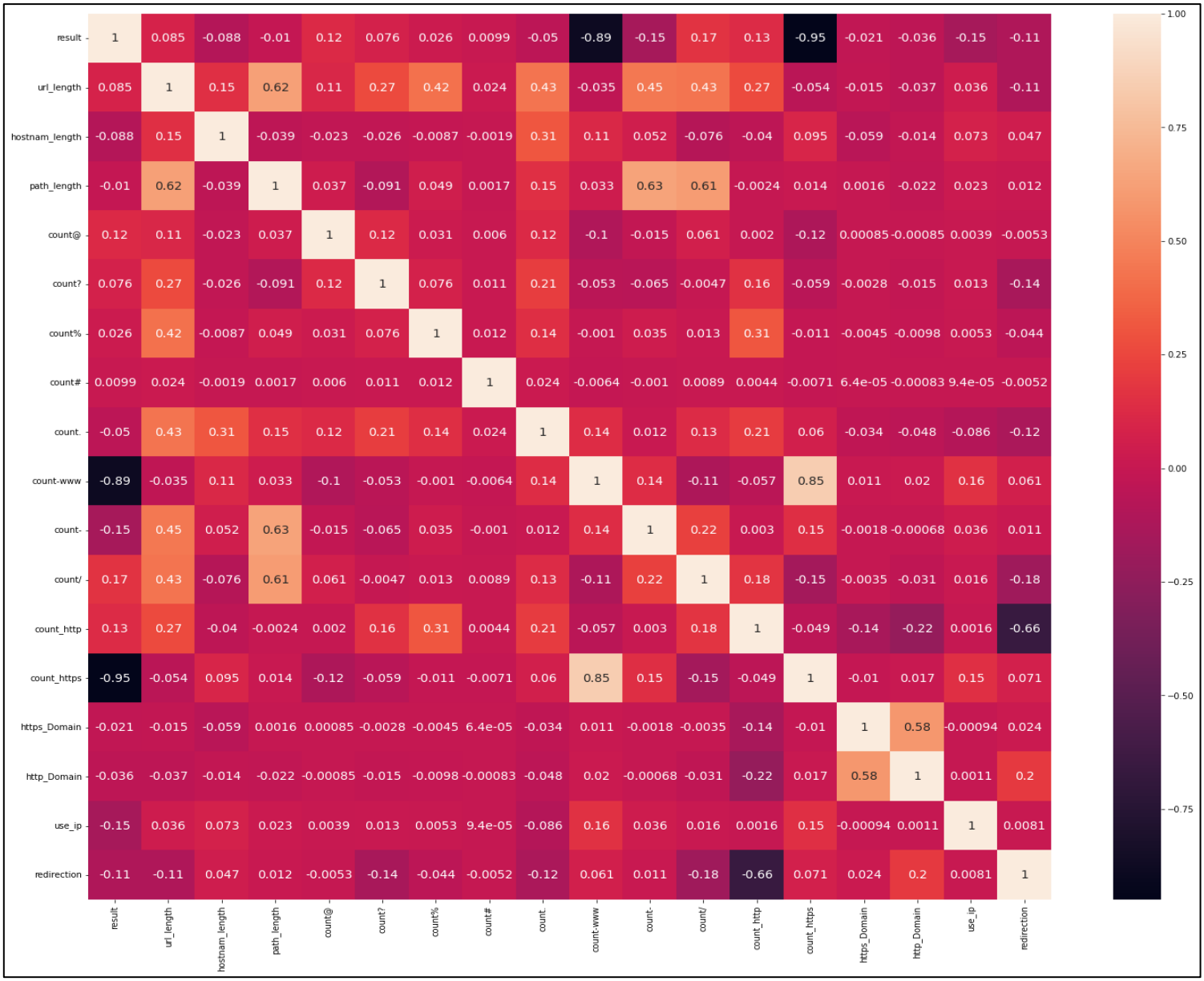
	count	mean	std	min	25%	50%	75%	max
result	450176.0	0.231994	0.422105	0.0	0.0	0.0	0.0	1.0
url_length	450176.0	60.236594	37.572056	8.0	40.0	52.0	71.0	2314.0
hostnam_length	450176.0	19.294403	6.689312	0.0	15.0	18.0	22.0	240.0
path_length	450176.0	27.435861	24.678631	0.0	11.0	22.0	38.0	1897.0
count@	450176.0	0.007017	0.096485	0.0	0.0	0.0	0.0	11.0
count?	450176.0	0.152854	0.462658	0.0	0.0	0.0	0.0	166.0
count%	450176.0	0.090007	1.166887	0.0	0.0	0.0	0.0	134.0
count#	450176.0	0.001297	0.235577	0.0	0.0	0.0	0.0	152.0
count.	450176.0	2.620553	1.144966	0.0	2.0	2.0	3.0	32.0
count-www	450176.0	0.803639	0.401434	0.0	1.0	1.0	1.0	6.0
count-	450176.0	1.251930	2.570521	0.0	0.0	0.0	1.0	42.0
count/	450176.0	4.444017	1.568950	2.0	3.0	4.0	5.0	46.0
count_http	450176.0	1.007039	0.099988	0.0	1.0	1.0	1.0	10.0
count_https	450176.0	0.784693	0.412508	0.0	1.0	1.0	1.0	8.0
https_Domain	450176.0	0.999729	0.023280	-1.0	1.0	1.0	1.0	1.0
http_Domain	450176.0	0.999200	0.039984	-1.0	1.0	1.0	1.0	1.0
use_ip	450176.0	0.986983	0.160826	-1.0	1.0	1.0	1.0	1.0
redirection	450176.0	0.992430	0.122815	-1.0	1.0	1.0	1.0	1.0

D'après ce tableau statistique, nous avons la confirmation qu'il y'ait aucune valeur manquante dans nos données. En outre, vue le minimum et maximum de variable, nous constatons qu'il y ait un large écart entre certaines variables dépendantes (par exemple *url_length* et *count-www*) ce qui pourra poser un problème d'échelle à certains algorithmes d'apprentissage automatique tels que la regression linéaire, la regression logistique, le réseau de neurones, etc. Ces algorithmes utilisent la descente de gradient comme technique d'optimisation nécessitant que les données soient mises à l'échelle[28]. A cela, il faut ajouter les algorithmes liés à la distance

[28] Pour avoir plus de détails sur le problème d'échelle et comment peut-il affecter la descente de gradient, veuillez cliquer ici.

tels que les k plus proches voisins (k-Nearest Neighbors), le k-Means et Support Vector Machine (SVM)[29]. Nous fermons cette parenthèse sur ces problèmes de mises en échelle de caractéristiques. Par ailleurs, il faut savoir que nous utilisons que quelques algorithmes d'apprentissage automatique, dans notre étude, parmi ceux qui ont été cité dans cette section. Ainsi, pour résoudre ce problème d'échelle entre les caractéristiques, nous recourons à la mise en échelle de nos variables via les méthodes de Preproccessing data[30] de la plateforme de machine learning Sklearn.

Graphique 4 : heatmap

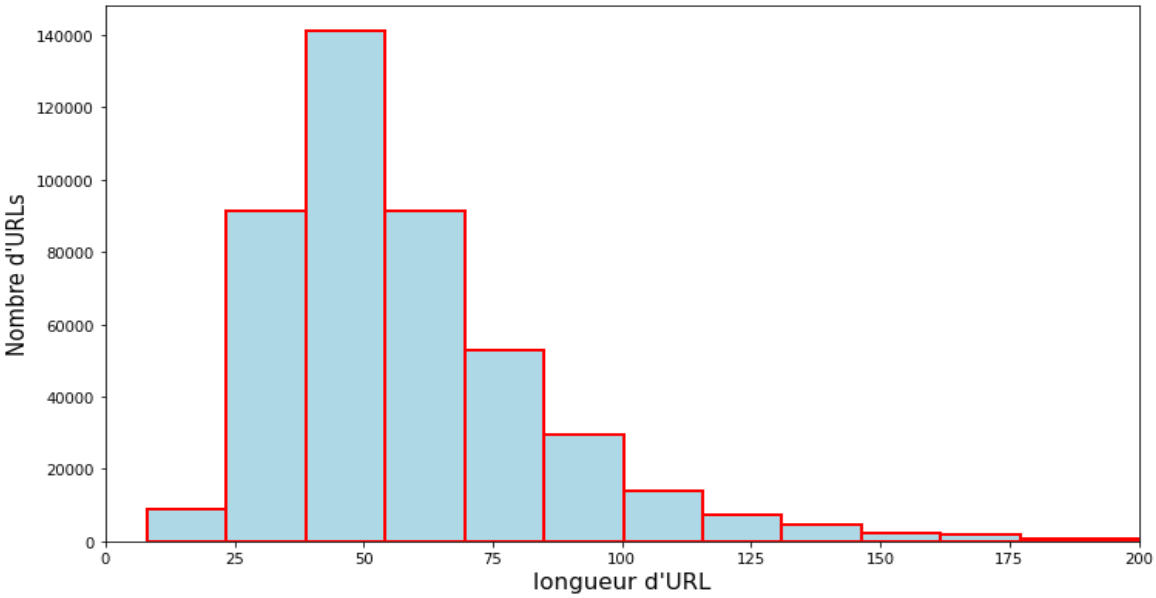


D'après ce heapmap, nous constatons qu'il y ait une forte corrélation entre certaines de nos variables explicatives (par exemple la corrélation est de 0.85 entre *count_https* et *count-www*).

[29] Pour avoir plus de détail sur ce problème d'échelle lié à la distance veuillez cliquer ici.
[30] https://scikit-learn.org/stable/modules/preprocessing.html

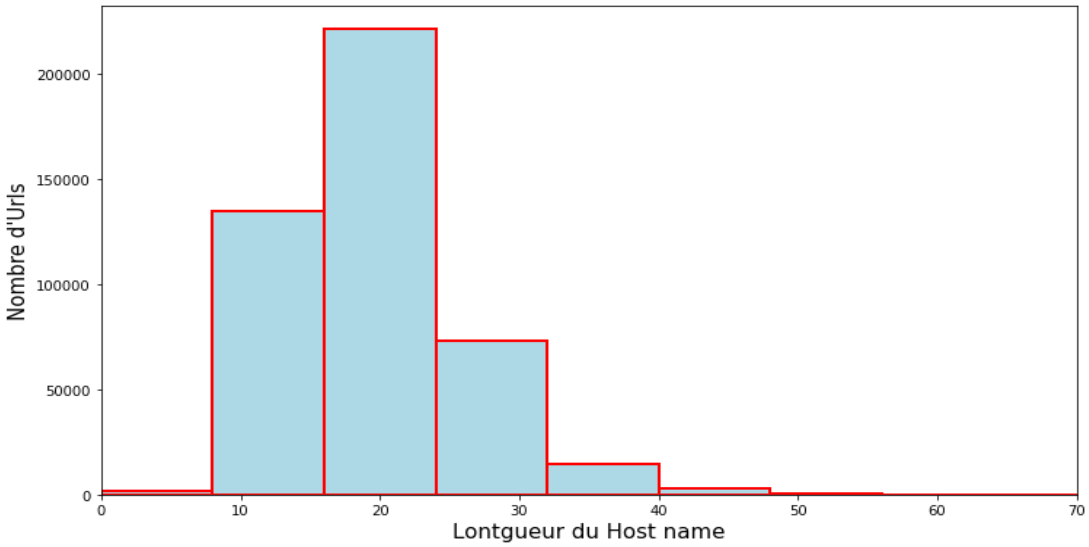
En effet, nous nous attendons à ce résultat, car toutes ces variables sont extraites d'une même caractéristique (url). Pour résoudre ce problème de corrélation, nous recourons aux méthodes de feature selction[31] de Sklearn. En outre, nous utilisons ces méthodes de sélection afin de comparer nos résultats avec et sans sélection de caractéristiques.

Graphique 5 : longueur d'URL



D'après cet histogramme, nous constatons que la plupart des URLs dont nous disposons dans notre base de données comprennent entre 25 et 100 caractères en longueur.

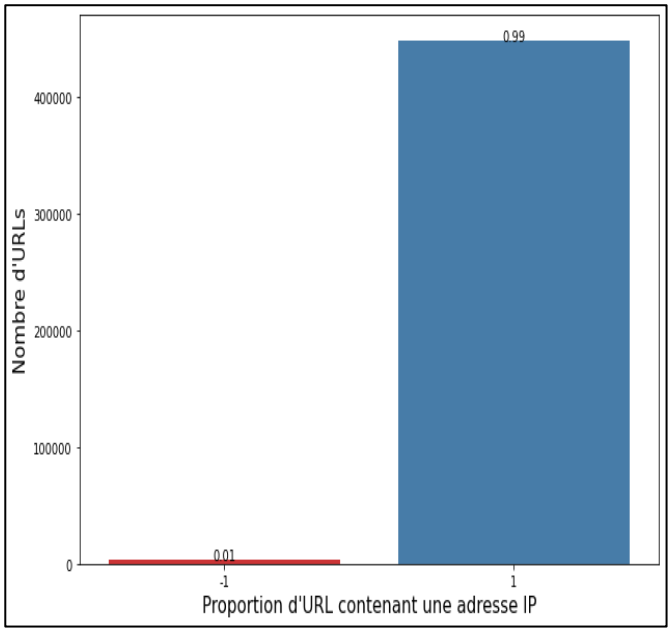
Graphique 6 : Longueur de l'host name

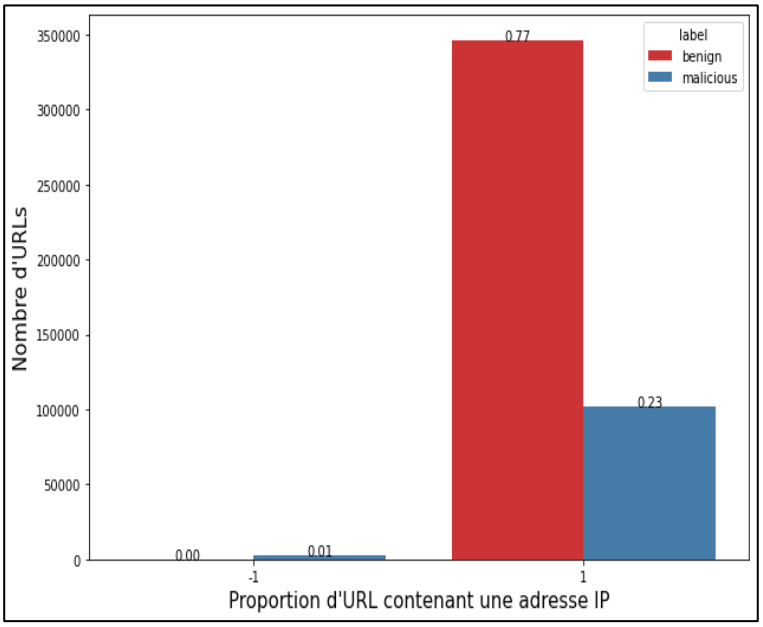


D'après cet histogramme, nous constatons que la plupart des URLs dont nous disposons dans notre base de données comprennent entre 10 et 40 caractères dans leur partie '**nom d'hôte**'.

[31] https://scikit-learn.org/stable/modules/feature_selection.html

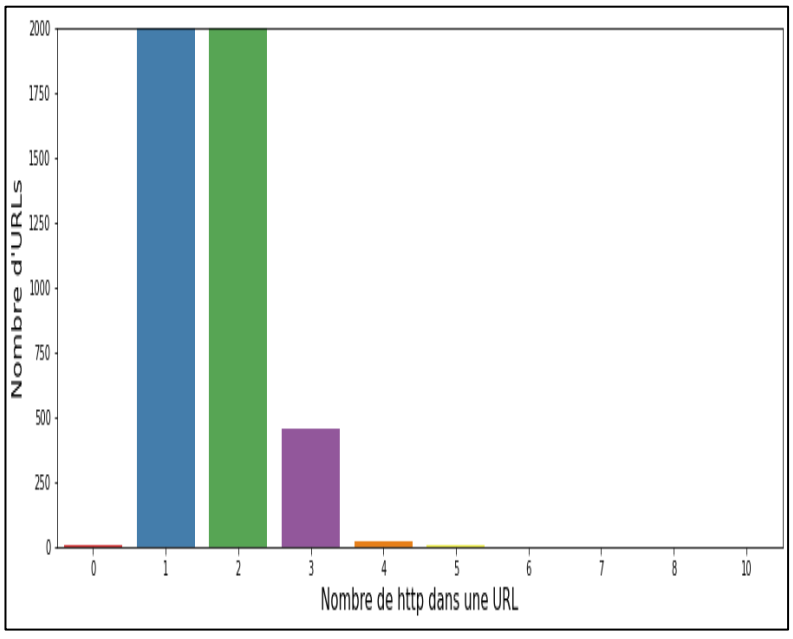
Graphique 7 : Proportion d'URLs contenant une adresse IP

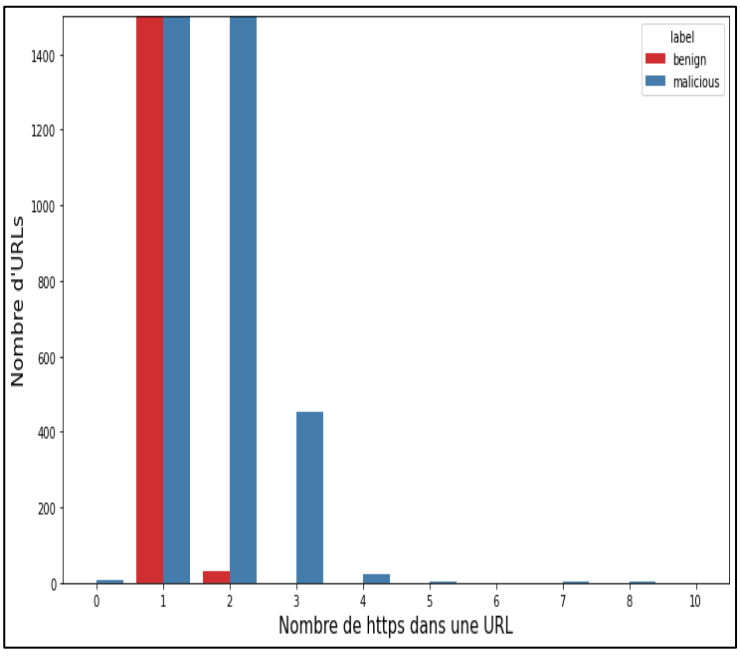




D'après ces countplots, nous constatons qu'il y ait peu d'URLs qui ont une adresse IP dans la partie '**nom du domaine**' et en plus toutes les URLs qui ont une adresse IP dans la partie '**nom du domaine**' sont des URLs de phishing ce qui confirme ce que nous avons dit ci-haut.

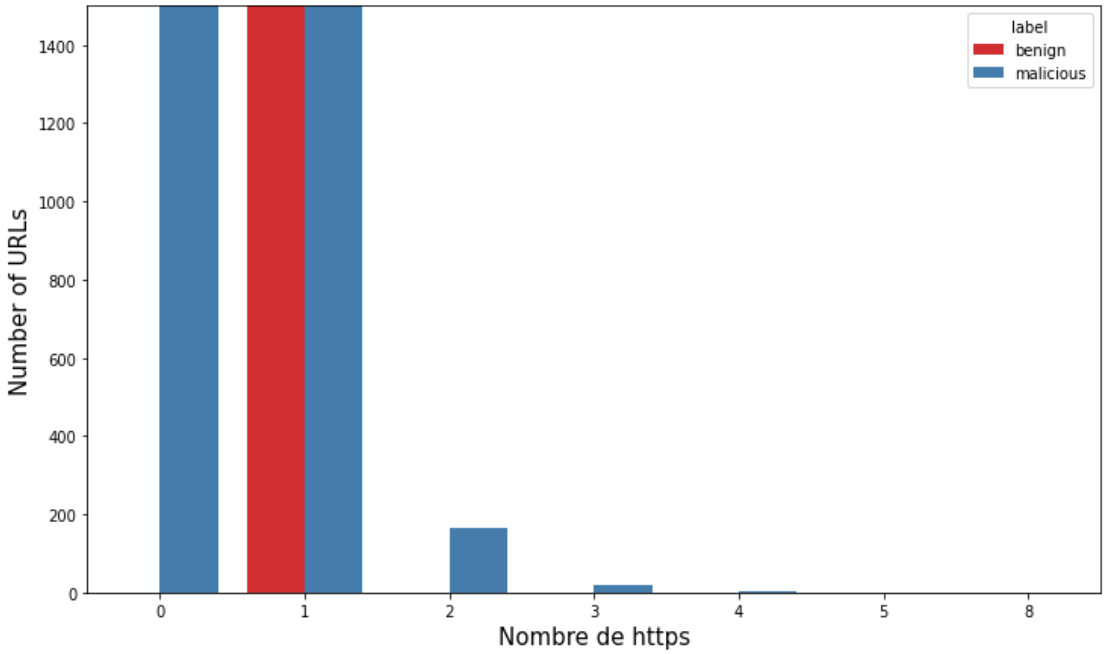
Graphique 8 : Nombre de http dans une URL





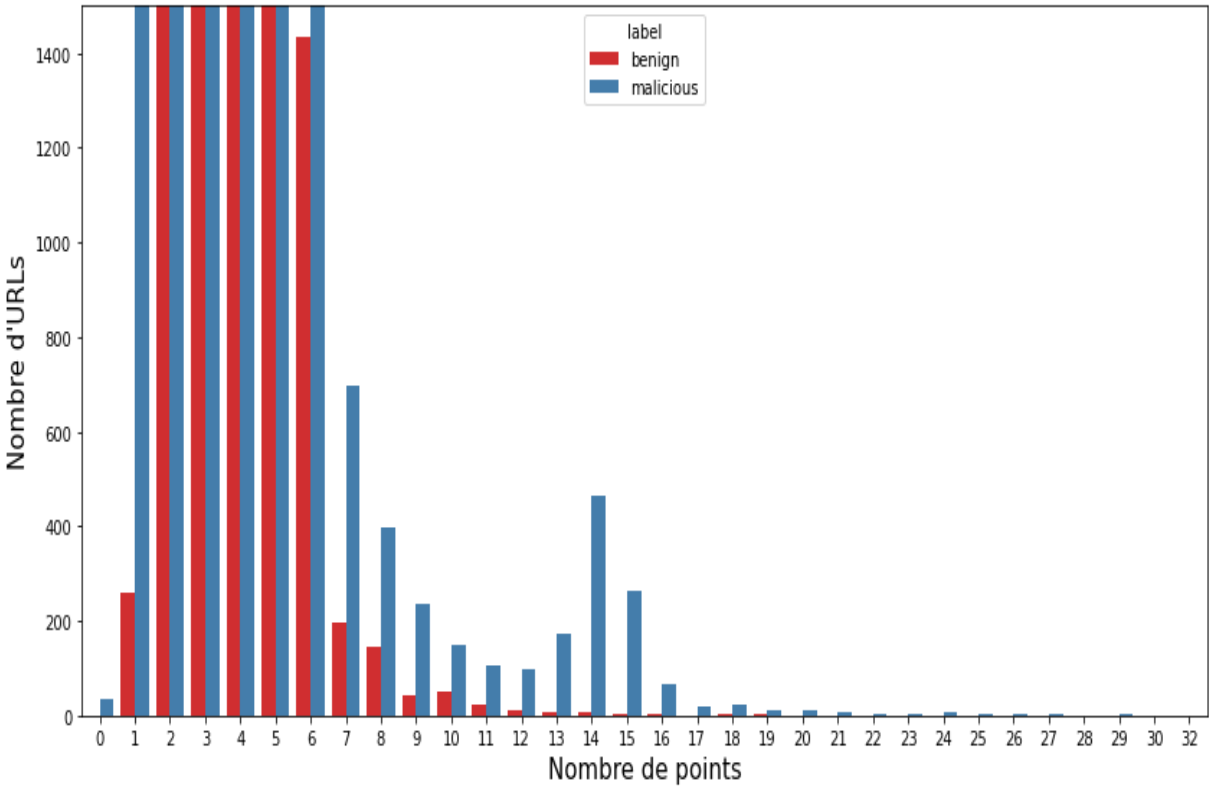
D'après ces countplots la plupart des URLs comprennent entre 0 et 4 '**http**' et en plus les URLs bénignes ne contiennent que deux '**http**', en revanche les URLs malveillantes en contiennent plus de deux.

Graphique 9 : nombre de 'https' dans une URL



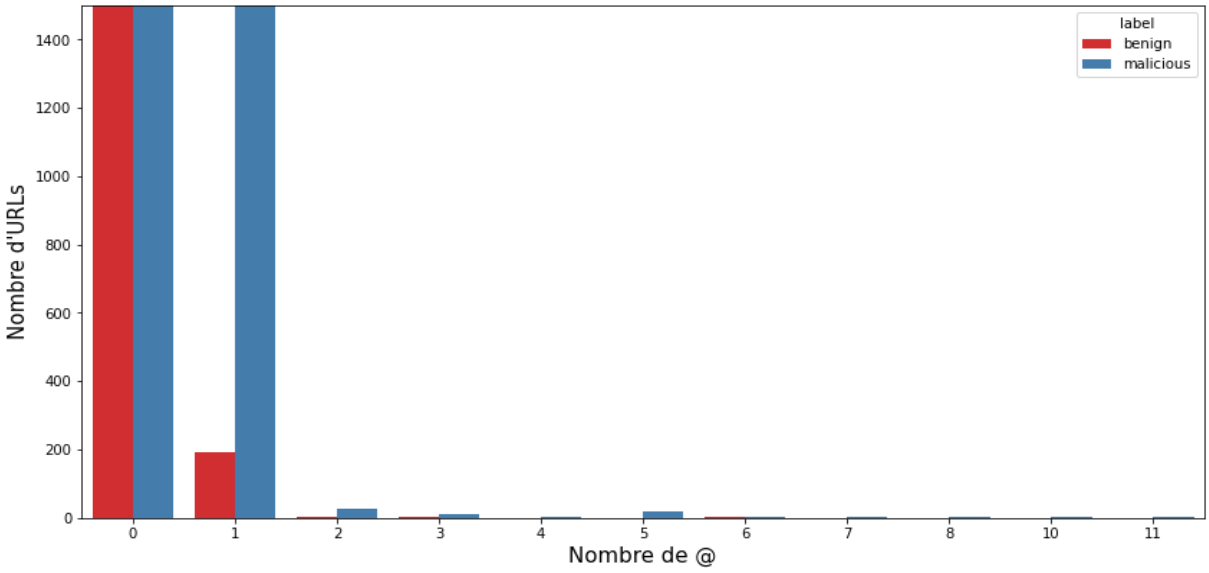
Concernant les '**https**', nous constatons que les URLs bénignes ne contiennent qu'un seul '**https**' tandis que les URLs malveillantes en contiennent soit 0 soit 1 et plus.

Graphique 10 : nombre de points dans une URL



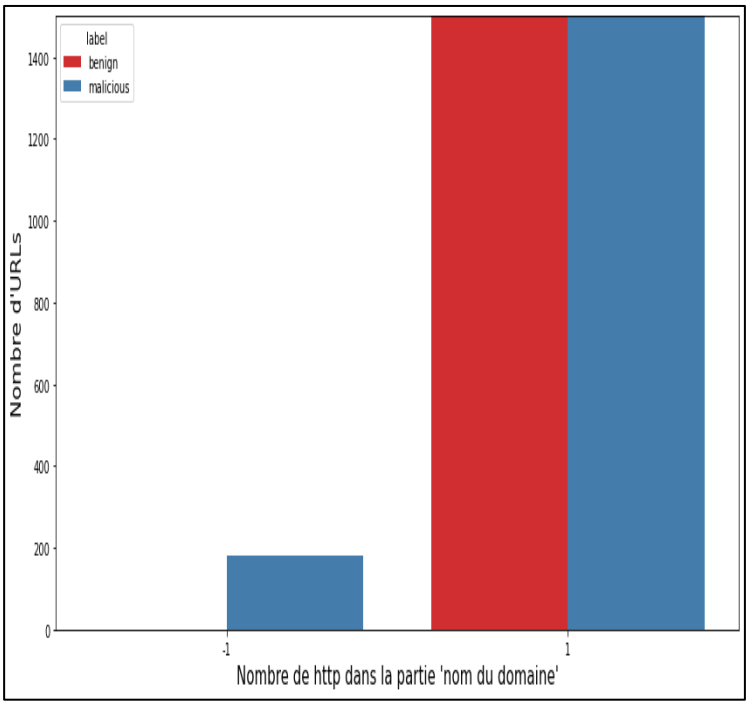
D'après ce graphique, nous constatons que la plupart d'URLs malveillantes ont plus de points par rapport aux URLs bénignes.

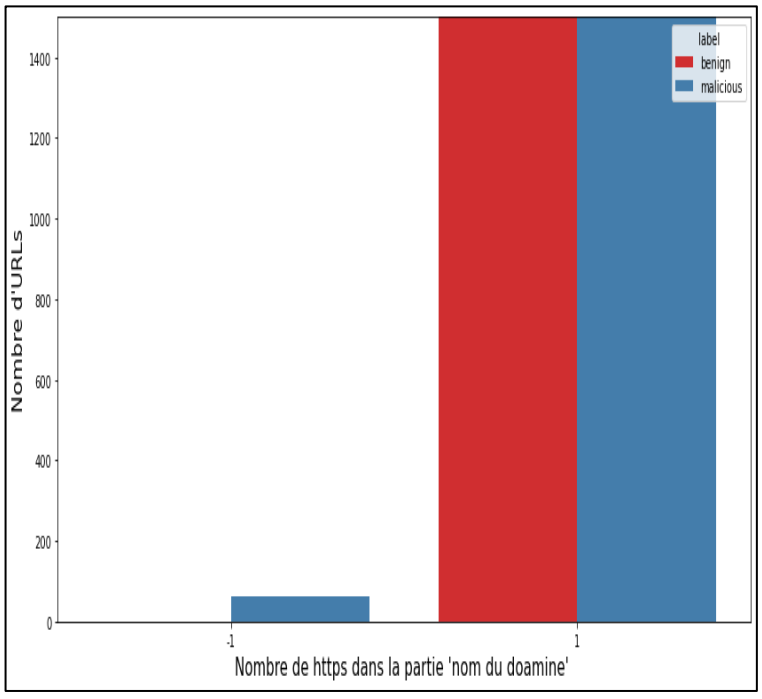
Graphique 11 : nombre de @ dans une URL



D'après ce graphique juste ci-dessus, nous constatons que la plupart d'URLs malveillantes ont plus de '@' par rapport aux URLs bénignes qui n'ont 0 ont un seul '@'.

Graphique 12 : nombre de http ou https dans la partie nom du domaine





D'après ces deux graphiques juste ci-dessus, nous constatons que ça soit dans le cas du 'http' ou 'https' dans la partie du 'nom de domaine', il y a que les URLs malveillantes qui contiennent de 'http' ou 'https' dans la partie 'nom du domaine'. Nous ne pouvons pas tout visualiser malheureusement, du coup nous nous arrêtons ici.