INF8215 – Intelligence artif. : méthodes et algorithmes

TP3 – Détection de Phishing

Présenté à

Pierre-Emmanuel Savoie

Par

Julien Savoldelli 2229682
Maximiliano Falicoff 2013658

Groupe Name (Kaggle): "Savoldelli_Falicoff"

SOMMAIRE

LIST OF FIGURES

1 INTRODUCTION

L’objectif de ce travail est d’appliquer une méthode de Machine Learning à un contexte de détection de phishing. Pour ce faire nous disposons de deux bases de données différentes. L’une possède des étiquetages indiquant si le site concerné est saint ou de phishing, l’autre est une base de données servant à la vérification d’apprentissage, nous ne possédons pas l’étiquetage et nous pouvons simplement connaitre la proportion de bonne prédiction grâce a une soumission sur le site « Kaggle ».

Nous sommes dans un cas d’application d’apprentissage supervisé, et plus particulièrement d’une application a un contexte de classification. Une grande variété de méthode sont utilisable, on peut citer : « Neural Network », « Nearest Neighbors », « Gaussian process » et bien d’autre.

2 PRETRAITEMENT DES ATTRIBUTS

Quel que soit la méthode utilisée pour réaliser cette prédiction, nous savons que les algorithmes sont influencés et biaisés par le « forme » des données d’entrée, or l’ensemble des attributs de notre base de données sont variées avec notamment des moyennes et des écarts types variés. Une opération de prétraitement consistant à les mettre sont forme d’une loi normale centré réduite permet de supprimer ce biais.

Pour rendre interprétable les étiquettes de « phishing » et « legitimate », nous avons choisi de remplacé ces caractéristiques par 1 et 0.

3 METHODOLOGIE

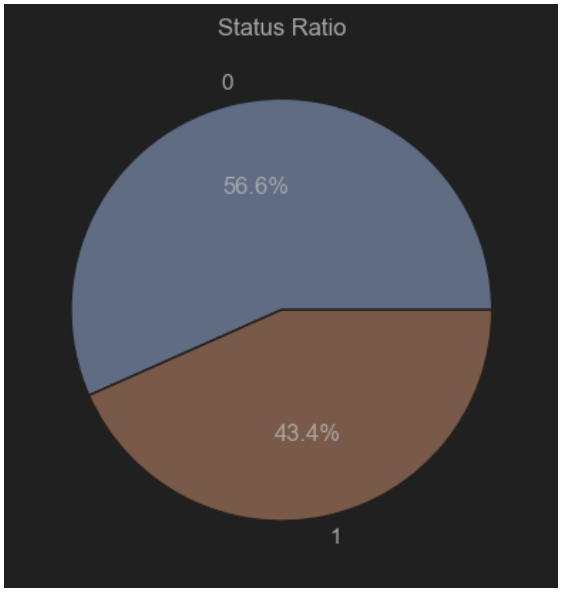
Nous avons utilisé la librairie scikit-learn dans l’ensemble du projet et des méthodes utilisées.

3.1 DESCRIPTION DES DONNEES

Procédons premièrement à analyser les données afin de voir si nous pouvons extraire certaines informations pour nous guider lors du développement de notre modèle.

Le fichier CSV fourni contient au-dessus de 8000 inputs, donc nous avons assez de points de inputs pour réaliser notre training.

Si nous considérons le ratio de legitimate par rapport au phishing, on observe le suivant :



On observe que le ratio est relativement équilibré, et donc nous n'allons pas procéder à faire un re balancement du data.

Figure 1 : Ratio legitimate et phishing

Analysons maintenant certaines des colonnes afin de voir si nous pouvons “dropper” certaines colonnes qui ne seraient pas intéressantes ou qui n’apportent peu d’information. Pour les graphes nous avons choisi les paramètres qui nous semble les plus pertinents à analyser et ce qui nous semble être le plus commun dans des url de type spam. A noter que nous séparons en fonction du status de l'url.

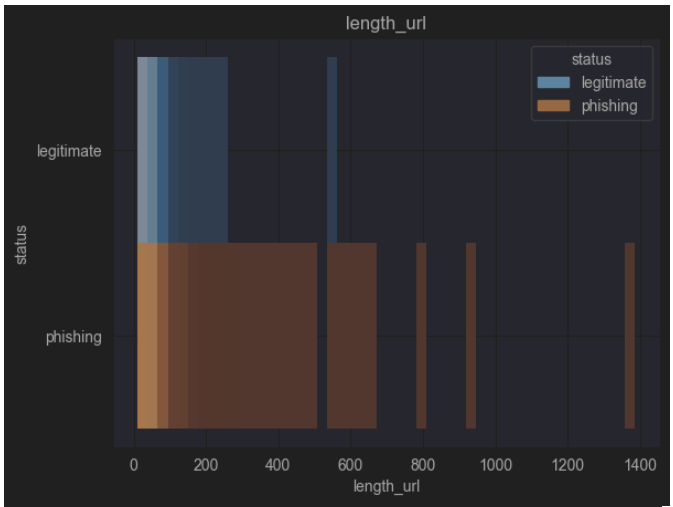


Figure 2 : Longueur url

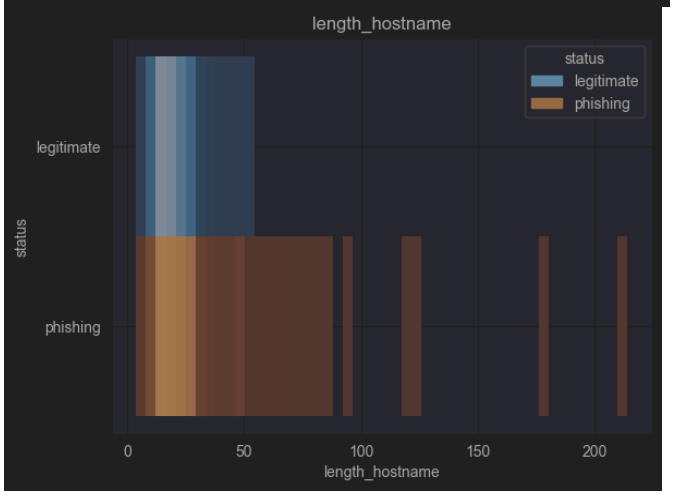


Figure 3 : Longueur du hostname

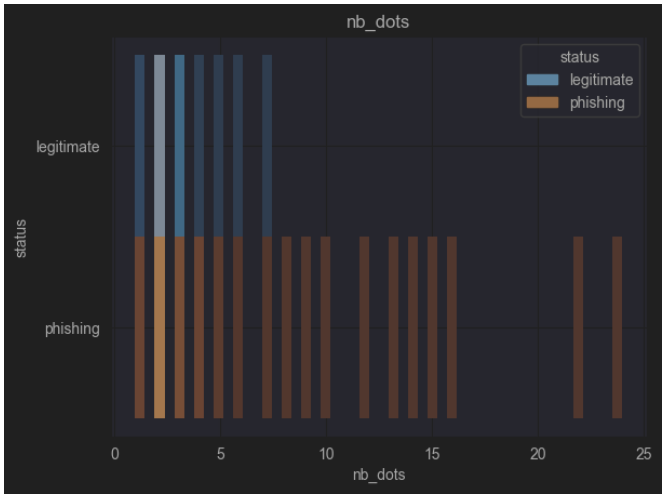


Figure 4 : Nombre de points dans url

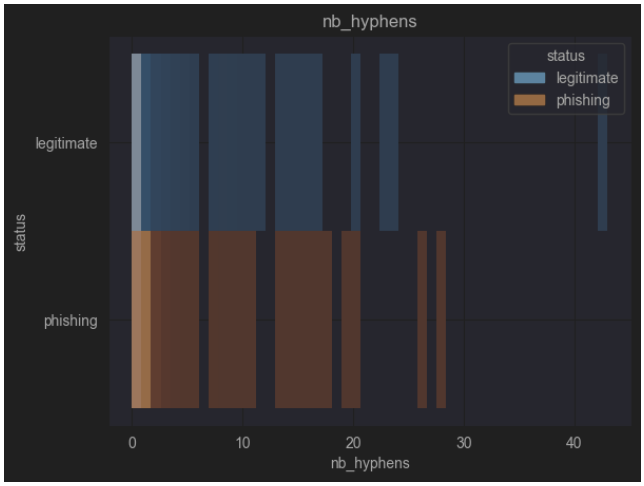


Figure 5 : Nombre de tirets dans url

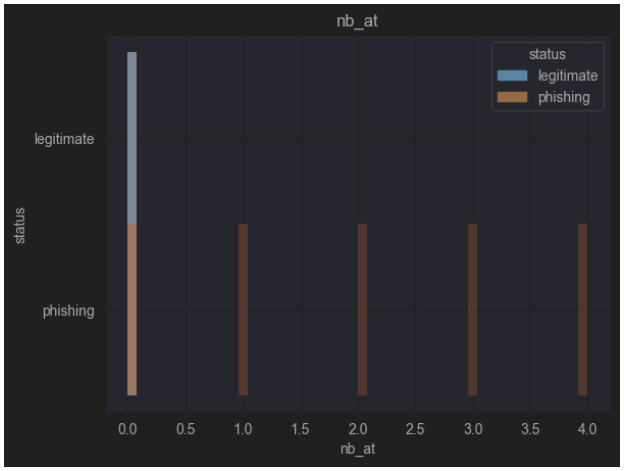


Figure 6 : Nombre de @ dans url

En soit on peut voir que en general

Regardons un peu comment les différentes colonnes sont relie entre-elles en créant une matrice de corrélation.

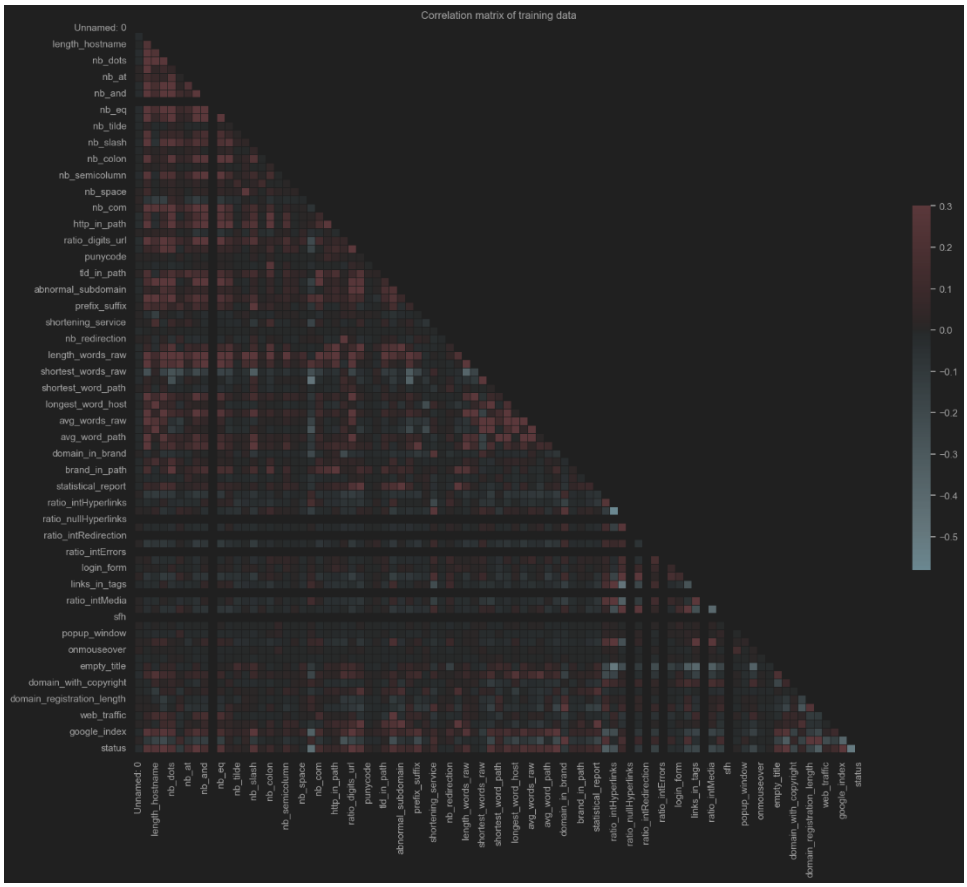


Figure 1 : Ratio legitimate et phishing

3.2 PREPROCESSING

Bullet points to convert to text:

- Description of preprocessing that we have done (columns dropped

3.3 MODEL AND PARAMETER CHOOSING

Bullet points to convert to text:

- Description of differents models that we tested
- Description of how we chose hyperparamters (gridsearchcsv)

Afin d’observer la performance de nos algorithmes, nous avons décidé de les tester sur un faible échantillon des données training, en effet la taille de la base de données training est suffisante pour réquisitionner une partie pour tester nos différentes solutions. Sur les 8172 entrées de la base de données, 7700 ont été utilisés pour l’apprentissage, le reste pour observer la précision.

4 RESULTATS

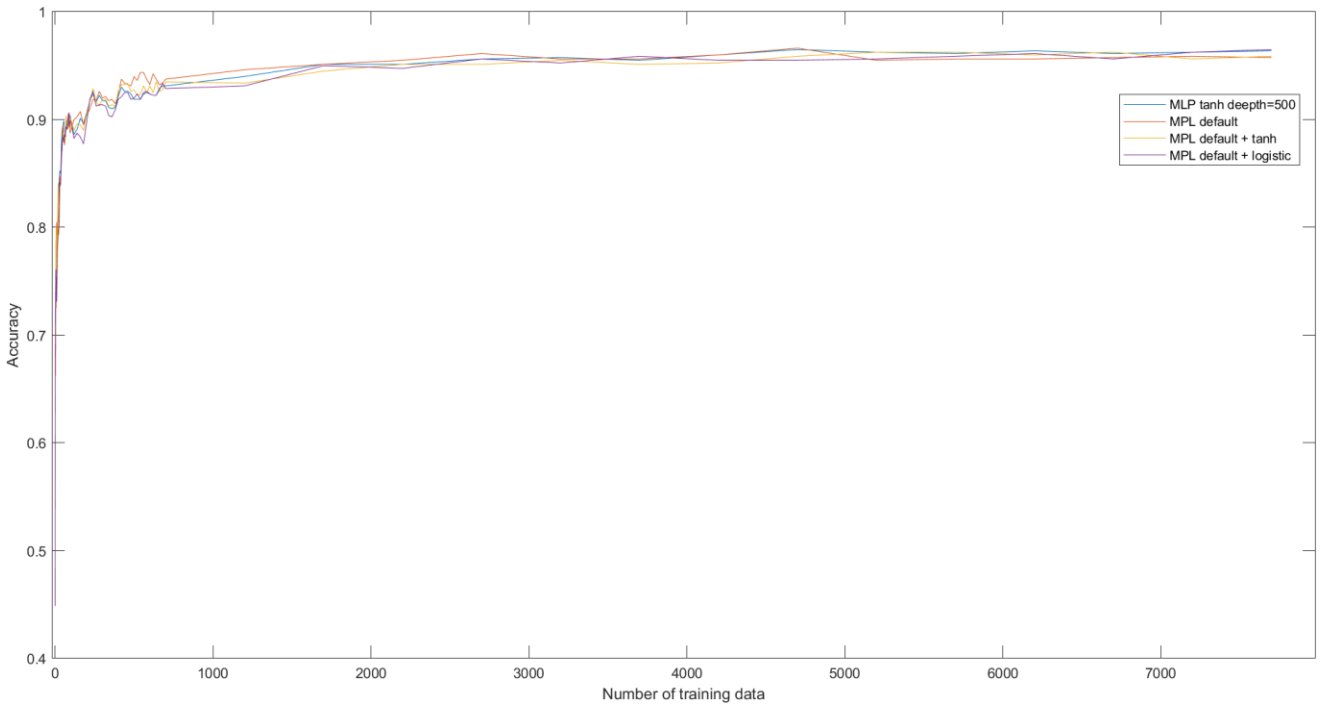


Figure 1 : Convergence - MPL – Comparaison

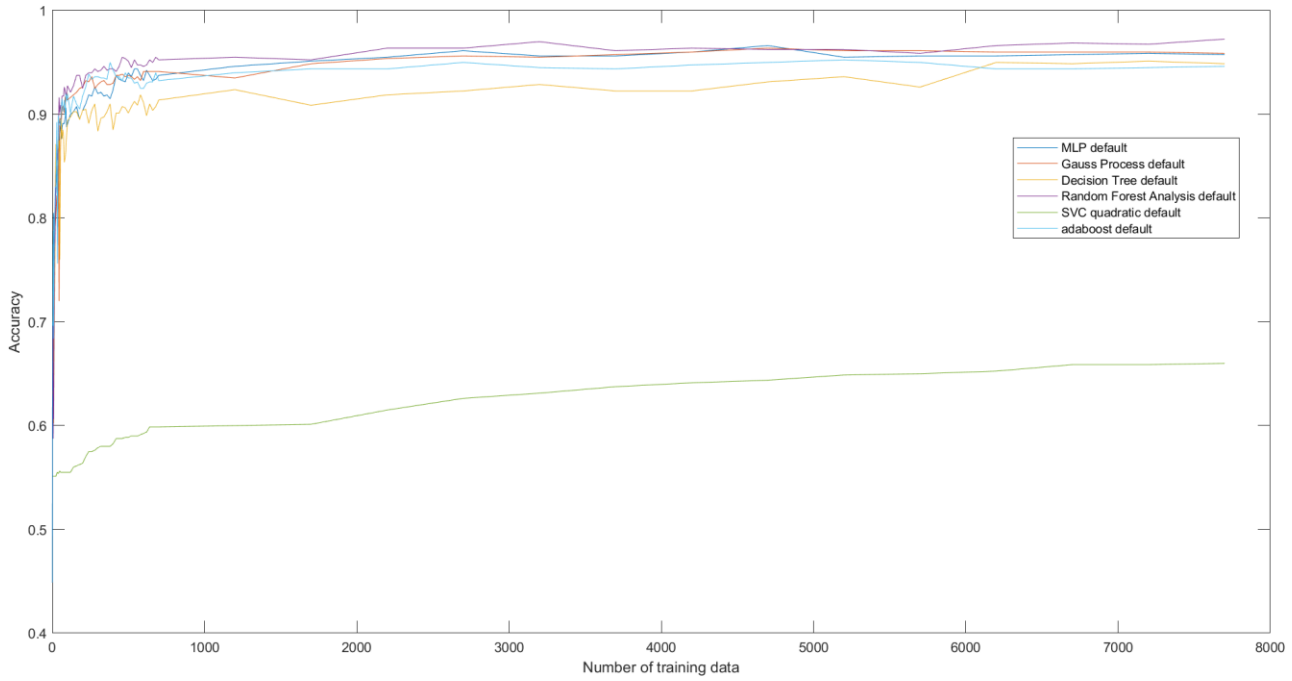


Figure 2 : Convergence – all model – comparaison

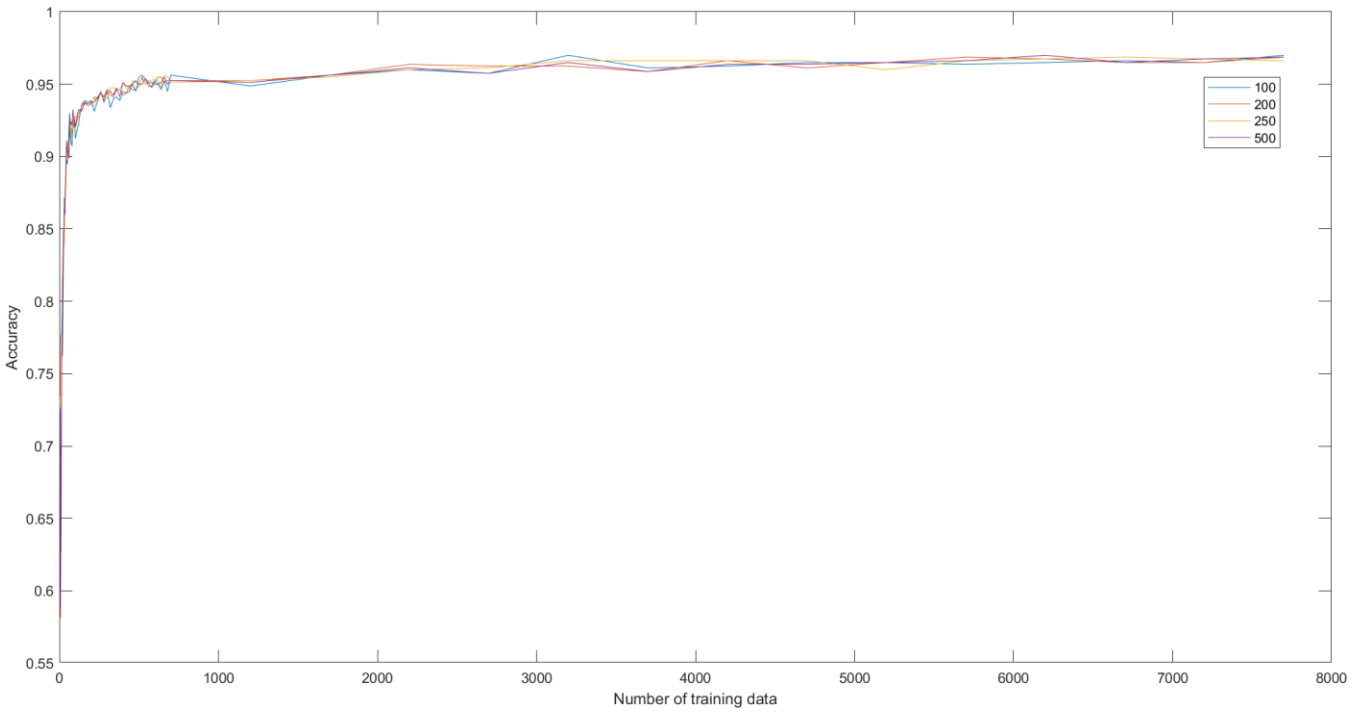


Figure 3 : Convergence - comparison - Random Forest Model - depending on the number of trees in the forest

5 DISCUSSION

On s’aperçoit que les algorithmes testés (réseau de neurones, Gauss) apprennent particulièrement rapidement à discriminer les sites de phishing, avec une précision asymptotique proche de 95%, on s’aperçoit aussi qu’une modification des paramètres comme la fonction d’activation ou la profondeur du réseau ne permette pas d’avoir une amélioration drastique de l’apprentissage.

On remarque aussi sur la Figure 2 que le modèle de gauss subit moins le phonème de désapprentissage que le modèle de MLP, en contrepartie les temps de calculs pour le modèle de gauss sont bien plus important que pour le modèle de MLP.

Pour ce qui est de la précision asymptotique qui n’est pas exactement 1, cela peut être du au type de données. On peut imaginer que certain site de phishing utilise des url particulières, qui sont rares et donc imprévisibles pour l’algorithme, pour améliorer la précision et gagner les derniers pourcentages de précisions il faudrait extraire des sites d’autres paramètres qui permettrait de discriminer plus en détaille les url.

6 BIBLIOGRAPHY

[1] Open-source, «scikit-learn (Supervised learning),» [En ligne]. Available: https://scikit-learn.org/stable/supervised_learning.html.