Présentation du Cadre :

Les URL, ou Uniform Resource Locators, sont les éléments fondamentaux qui permettent de naviguer sur l'immensité du web. Elles agissent comme des adresses numériques, dirigeant les utilisateurs vers les ressources qu'ils recherchent en un simple clic. Cependant, cette accessibilité et cette omniprésence des URL ont également ouvert la voie à une prolifération des cybermenaces, notamment sous la forme d'attaques de « phishing ».

En 2024, les attaques de phishing ont atteint des niveaux sans précédent, avec près de 1,9 million d'incidents rapportés dans le monde. Ce chiffre représente une augmentation significative par rapport aux années précédentes et reflète l'évolution des stratégies des cybercriminels​. L'une des tendances alarmantes est l'utilisation croissante de sous-domaines pour héberger des sites de phishing, avec une hausse de 51 % en 2024, atteignant ainsi plus de 450 000 sous-domaines malveillants. Ces sous-domaines sont souvent utilisés pour duper les utilisateurs en se faisant passer pour des sites légitimes.

Une autre tactique courante est l'exploitation de certificats SSL pour créer un faux sentiment de sécurité. Historiquement, les utilisateurs étaient encouragés à vérifier la présence du "https" dans l'URL pour s'assurer de la sécurité d'un site. Cependant, en 2024, 84 % des sites de phishing détectés utilisaient des certificats SSL, ce qui rend cette méthode de vérification obsolète pour évaluer la sécurité d'un site. Cela démontre que les cybercriminels sont devenus de plus en plus sophistiqués dans leurs approches, rendant difficile la distinction entre un site sûr et un site malveillant.

En plus de ces tactiques, les cybercriminels exploitent de plus en plus les nouvelles technologies pour lancer leurs attaques. Par exemple, l'utilisation du système décentralisé InterPlanetary File System (IPFS) pour héberger des sites de phishing a augmenté de 1 300 % cette année, illustrant une nouvelle vague de techniques avancées pour contourner les mesures de sécurité traditionnelles.

Face à cette menace croissante, il est essentiel que les utilisateurs d'Internet adoptent une posture de vigilance renforcée. Cela inclut la vérification minutieuse des URL avant de cliquer, la méfiance envers les emails et les messages instantanés non sollicités, et l'utilisation de solutions de sécurité robustes. Par ailleurs, les entreprises doivent continuer à sensibiliser leurs employés aux dangers du phishing et investir dans des technologies capables de détecter et de bloquer ces menaces en temps réel.

Le phishing est devenu un problème mondial qui nécessite une réponse coordonnée à l'échelle internationale. Les politiques publiques, la collaboration entre les registraires de noms de domaine, les fournisseurs de services Internet, et les agences gouvernementales jouent tous un rôle crucial dans la lutte contre cette menace. Des initiatives telles que la vérification d'identité numérique pour l'enregistrement de domaines et l'adoption de systèmes automatisés pour détecter les enregistrements suspects sont des étapes importantes pour réduire l'impact des attaques de phishing.

Mandat du Projet.

Objectif Global

Le mandat de ce projet est de développer un modèle d’apprentissage supervisé capable de détecter automatiquement les URL de phishing en se basant sur des caractéristiques précises de l’URL. L'objectif est de classifier efficacement les URL en deux catégories :

1. **URL légitime** : Les URL authentiques, souvent associées à des sites de confiance.
2. **URL de phishing** : Les URL malveillantes créées pour tromper les utilisateurs et les inciter à révéler des informations sensibles, comme leurs identifiants ou informations bancaires.

**Étapes Clés du Projet**

Le projet s'articule autour de plusieurs étapes principales :

1. **Prétraitement des Données** : Avant d'entraîner les modèles, il est crucial de nettoyer les données, de gérer les valeurs manquantes ou incohérentes, et de préparer le dataset pour l'analyse.
2. **Exploration des Données** : Cette étape consiste à explorer la distribution des données, les relations entre les variables (par exemple, la corrélation entre la longueur de l'URL et la probabilité qu’elle soit de phishing), et à identifier les caractéristiques les plus importantes.
3. **Sélection des Caractéristiques** : Toutes les variables présentes dans le dataset ne sont pas forcément pertinentes pour la classification. L'objectif ici est d'identifier et de sélectionner les caractéristiques qui apportent la plus grande valeur prédictive.
4. **Développement du Modèle** : Plusieurs algorithmes d’apprentissage supervisé sont testés (forêts aléatoires, régression logistique, réseaux de neurones, etc.) pour identifier celui qui offre les meilleures performances sur les données.
5. **Évaluation du Modèle** : Les performances des modèles sont évaluées à l’aide de métriques telles que la précision, le rappel, le F1-score, et l'AUC-ROC. Ces mesures permettent d’évaluer dans quelle mesure le modèle parvient à distinguer les URL légitimes des URL de phishing.
6. **Déploiement et Surveillance** : Une fois le modèle final sélectionné, il sera déployé dans un environnement de production et suivi en temps réel pour s’assurer que ses performances restent optimales face à de nouvelles données.

Justification de l'Utilisation de BigQuery

L’utilisation de **Google BigQuery** pour entraîner et déployer les modèles d’apprentissage automatique repose sur plusieurs avantages stratégiques :

1. **Scalabilité** : BigQuery est conçu pour traiter et analyser de grandes quantités de données rapidement et efficacement. Cela permet de manipuler un dataset volumineux comme celui que j’utilise ici (plus de 235 000 lignes et 54 colonnes) sans que la performance du modèle ne soit affectée par la taille des données.
2. **Intégration Simplifiée** : BigQuery ML permet d’entraîner des modèles directement à partir de la base de données, sans avoir à exporter les données. Cela simplifie le processus de développement et permet d'effectuer des requêtes SQL pour l'entraînement des modèles, leur évaluation et leur utilisation.
3. **Rapidité et Efficacité** : En utilisant la puissance de calcul de Google Cloud, les modèles peuvent être entraînés plus rapidement qu’avec des solutions locales. Cela est particulièrement utile lors de l’expérimentation avec plusieurs modèles ou lors de l’ajustement des hyperparamètres.
4. **Maintenance Réduite** : BigQuery étant une solution cloud, il réduit considérablement le besoin de maintenance des infrastructures matérielles, permettant ainsi de se concentrer davantage sur le développement du modèle lui-même.

Description du Dataset :

Pour faire face aux défis de détection des URL de phishing, ce **dataset spécifique**, complété le **3 mars 2024**, contient **54 colonnes** et **235 795 lignes**, dont **100 945 URL identifiées comme des URL de phishing**. Cette base de données solide est conçue pour développer des outils avancés de détection.

Principales catégories de colonnes :

1. **Identification des URL** : Informations essentielles pour identifier chaque URL unique.
   * **FILENAME** : Nom du fichier associé.
   * **URL** : Adresse web complète analysée.
   * **Domain** : Domaine principal, crucial pour identifier l'origine du site.
2. **Caractéristiques des URL** : Mesures spécifiques sur la structure de l'URL.
   * **URLLength** : Longueur totale en caractères.
   * **DomainLength** : Longueur du domaine, un indice de légitimité.
   * **IsDomainIP** : Indicateur binaire si le domaine est une adresse IP (signe potentiel de phishing).
3. **Obfuscation et sécurité** : Évaluation des aspects de sécurité.
   * **HasObfuscation** : Utilisation de techniques d'obfuscation pour masquer l’objectif.
   * **IsHTTPS** : Utilisation du protocole HTTPS, perçu comme plus sécurisé.
   * **ObfuscationRatio** : Ratio d'obfuscation dans l'URL, utile pour repérer les URL malveillantes.
4. **Contenu de la page** : Analyse du contenu réel des pages liées aux URL.
   * **HasTitle** : Présence d’un titre de page, crucial pour la confiance de l’utilisateur.
   * **DomainTitleMatchScore** : Score de cohérence entre le domaine et le titre.
   * **NoOfPopup** : Nombre de popups, souvent utilisé pour tromper les utilisateurs.
5. **Références et scripts** : Éléments techniques sur les références internes et externes.
   * **NoOfSelfRef** : Nombre de références internes.
   * **NoOfExternalRef** : Nombre de références externes, signe possible de phishing.
   * **NoOfJS** : Nombre de fichiers JavaScript, parfois utilisés pour des scripts malveillants.
6. **Marqueurs spécifiques** : Indicateurs de sites liés à des industries sensibles.
   * **Bank** : Association aux services bancaires, cible fréquente de phishing.
   * **Pay** : Lien avec les services de paiement.
   * **Crypto** : Association aux cryptomonnaies, sujet d’escroqueries en ligne.
7. **Label de classification** : Colonne cible de l'analyse indiquant si l'URL est un **phishing (1)** ou **légitime (0)**. C’est la variable que notre modèle supervisé cherche à prédire.

**Processus de Transformation et de Consolidation des Données**

Objectif : Consolider les données provenant de plusieurs sources en un seul fichier propre afin de faciliter les analyses ultérieures et l’intégration dans BigQuery à l’aide de Python.

**1. Extraction des Données :**

* **Sources :** Trois fichiers plats distincts, nommés extract\_gcfp\_1, extract\_gcfp\_2 et extract\_gcfp\_3, ont été identifiés comme sources principales.
* **Outil d’extraction :** Chaque fichier a été importé à l’aide d’un pipeline ETL (Extract, Transform, Load), avec des configurations de gestion des erreurs pour rediriger les lignes malformées vers des sorties dédiées (trash\_error\_1, trash\_error\_2, trash\_error\_3).

**2. Transformation :**

* **Transformation des libellés :** Les données de chaque fichier ont subi une étape de transformation des libellés afin de garantir une structure et un format uniformes pour tous les ensembles de données. Ces transformations ont standardisé les noms de champs, appliqué des règles de formatage et corrigé les incohérences pour s’aligner sur le schéma cible.

**3. Gestion des Erreurs :**

* **Gestion des erreurs :** Les lignes contenant des incohérences ou des violations de format ont été redirigées vers des sorties d’erreurs (par exemple, trash\_error\_1). Ces erreurs ont été enregistrées à des fins d’audit mais exclues du traitement ultérieur pour garantir l’intégrité des données.

**4. Consolidation des Données :**

* **Opération Union :** Les trois ensembles de données transformés ont été combinés en un seul ensemble à l’aide de l’opération Union All. Cette étape a fusionné les données tout en conservant tous les enregistrements.

**5. Tri et Déduplication :**

* **Transformation de tri :** Une opération de tri a été appliquée à l’ensemble de données unifié pour organiser les données selon des clés prédéfinies.
* **Déduplication :** Les lignes en double ont été identifiées et supprimées afin de garantir que chaque enregistrement apparaisse une seule fois dans le fichier de sortie final.

**6. Exportation :**

* **Opération Multicast :** L’ensemble de données consolidé et nettoyé a été envoyé vers un fichier plat physique pour stockage.
* **Format de fichier :** Le fichier final a été enregistré dans un format structurant (par exemple, CSV ou TSV) compatible avec BigQuery pour une intégration fluide.

**Objectif :**

Le fichier consolidé sert de jeu de données principal pour ce projet, facilitant :

* L’intégration avec des scripts Python pour le téléchargement des données dans BigQuery.
* La garantie de la qualité et de la cohérence des données à toutes les étapes du projet.
* La réduction de la redondance et la minimisation des erreurs lors de la phase d’analyse des données.

Ce processus a non seulement rationalisé le flux de préparation des données mais a également permis de s’assurer que le jeu de données était optimisé pour l’entraînement des modèles de machine learning et l’analyse ultérieure dans BigQuery.

**Exploration des Données et Analyse des Corrélations**

L'exploration des données est une étape cruciale avant d'entraîner les modèles de machine learning. Elle permet de mieux comprendre la distribution des variables, d'identifier les relations entre celles-ci et la variable cible, et d’extraire des informations qui orienteront la sélection des caractéristiques et le choix des modèles. Voici un aperçu des décisions prises et des résultats obtenus lors de cette phase d'exploration.

**1. Visualisation de la Distribution des Variables**

**Décision :** Il était essentiel de comprendre comment les différentes caractéristiques du dataset sont distribuées, notamment les variables numériques et catégorielles. Pour ce faire, des visualisations sous forme d’histogrammes ont été générées pour chaque variable.

**Actions :**

* Des **histogrammes** ont été tracés pour les colonnes numériques telles que URLLength, DomainLength, NoOfSubDomain, etc. Ces visualisations ont permis d’observer les concentrations de valeurs dans certaines plages et de détecter les éventuels déséquilibres dans les distributions (comme la présence de valeurs extrêmes).
* Des **diagrammes en barres** ont été utilisés pour les variables catégorielles (comme IsHTTPS, HasPasswordField, etc.) pour comprendre la répartition des valeurs entre les différentes catégories (0 ou 1).

**Résultats :**

* La plupart des variables numériques avaient une distribution très asymétrique. Par exemple, URLLength et DomainLength ont montré des distributions très concentrées autour de petites valeurs, avec une longue traîne de valeurs plus grandes. Cela pourrait indiquer que les URL de phishing tendent à être soit très courtes, soit très longues, ce qui nécessite une analyse plus approfondie.
* Les variables binaires (comme IsHTTPS et HasPasswordField) ont montré une répartition plutôt équilibrée, mais certaines comme Bank et Crypto étaient relativement rares dans le dataset, ce qui pourrait poser des défis pour la modélisation.

**2. Matrice de Corrélation**

**Décision :** Comprendre la relation entre les différentes variables du dataset et la variable cible (label, qui indique si l'URL est de phishing ou non) est essentiel pour la sélection des caractéristiques. Pour ce faire, une **matrice de corrélation** a été calculée pour identifier les relations entre les variables.

**Actions :**

* Une **carte thermique des corrélations** a été générée pour visualiser les relations entre les caractéristiques numériques du dataset. La corrélation entre chaque paire de variables a été calculée à l'aide du coefficient de corrélation de Pearson. La carte thermique a permis d’identifier les variables qui sont fortement corrélées les unes avec les autres ainsi que celles qui ont une forte corrélation avec la variable cible.

**Action supplémentaire :** Pour les variables ayant une forte corrélation (supérieure à 0,65) entre elles, comme URLLength et DomainLength, une des deux variables a été écartée lors de la sélection finale des caractéristiques afin d’éviter le phénomène de multicolinéarité, où des variables très corrélées entre elles peuvent introduire des redondances et nuire à la performance du modèle.

**3. Analyse des Corrélations avec la Variable Cible**

**Décision :** Il est essentiel de comprendre quelles variables sont les plus informatives pour la classification des URL de phishing. Pour ce faire, nous avons analysé les corrélations entre les caractéristiques du dataset et la variable cible (label).

**Actions :**

* Les variables ayant une **corrélation forte** (positive ou négative) avec la variable cible ont été identifiées comme des candidats potentiels pour la modélisation.
* Par exemple, HasExternalFormSubmit (qui indique si une page utilise un formulaire de soumission externe) et NoOfPopup (nombre de popups sur la page) avaient des corrélations positives modérées avec la cible, ce qui montre que ces caractéristiques pourraient être des signes distinctifs des URL de phishing.

**Résultats :**

* Les URL de phishing ont tendance à avoir certaines caractéristiques spécifiques. Par exemple, elles sont plus susceptibles d’utiliser des sous-domaines, d’avoir des longueurs de domaine variables, et de contenir des formulaires externes (ce qui pourrait être utilisé pour subtiliser des informations).
* Certaines caractéristiques de sécurité, comme l’utilisation du protocole HTTPS (IsHTTPS), étaient faiblement corrélées avec la cible. Cela démontre que la simple présence du protocole HTTPS ne garantit pas que l’URL est légitime, une observation cohérente avec la réalité des attaques de phishing modernes où de nombreux sites malveillants utilisent désormais HTTPS.

**4. Identification des Caractéristiques Problématiques (Multicolinéarité)**

**Décision :** Il est important de s’assurer que les caractéristiques sélectionnées ne sont pas redondantes entre elles (multicolinéarité) afin de garantir la stabilité des modèles. Pour cette raison, un **facteur d'inflation de la variance (VIF)** a été calculé pour chaque variable.

**Actions :**

* Le VIF a été utilisé pour mesurer la multicolinéarité entre les caractéristiques. Les caractéristiques avec un VIF supérieur à 5 ont été considérées comme ayant une multicolinéarité excessive, et ont été exclues de la sélection finale des variables.

**Résultats :**

* Des variables comme NoOfAmpersandInURL et DomainLength avaient des scores de VIF élevés, indiquant qu’elles étaient fortement corrélées avec d’autres variables, et elles ont été exclues lors de la phase de modélisation pour éviter des résultats biaisés.

**5. Sélection des Caractéristiques Finales**

**Décision :** Sur la base des corrélations et des résultats du VIF, une sélection finale des caractéristiques a été effectuée pour s’assurer que seules les variables pertinentes et non redondantes sont utilisées dans l’entraînement des modèles.

**Résultats :**

* Un ensemble de variables optimisé a été constitué pour l’entraînement, garantissant que les caractéristiques sélectionnées sont informatives, non redondantes et pertinentes pour la classification des URL de phishing.

**Conclusion de l'Exploration des Données**

L'exploration des données a permis d’identifier des tendances claires concernant les URL de phishing. Plusieurs caractéristiques, notamment la longueur de l'URL, l'utilisation de sous-domaines, et les indicateurs de sécurité comme la soumission de formulaires externes, sont apparues comme des variables clés dans la détection des URL malveillantes. En éliminant les variables redondantes et en se concentrant sur les plus informatives, nous avons créé une base solide pour les prochaines étapes.

**Prétraitement des Données**

Le prétraitement des données est une étape fondamentale dans tout projet d'apprentissage automatique. Avant de pouvoir entraîner des modèles, il est essentiel de s'assurer que les données sont propres, cohérentes, et prêtes à être exploitées par les algorithmes de machine learning. Dans ce projet, plusieurs étapes clés de prétraitement ont été effectuées pour garantir l'intégrité et la qualité des données. Voici les détails des décisions prises lors de cette phase.

**1. Gestion des Valeurs Manquantes**

**Décision :** Les valeurs manquantes dans les données peuvent biaiser les résultats du modèle et diminuer sa capacité de généralisation. Ainsi, la première étape a consisté à identifier les colonnes présentant des valeurs manquantes dans le dataset.

**Actions :**

* J’ai utilisé la méthode .isnull().sum() pour compter le nombre de valeurs manquantes dans chaque colonne.
* Dans le cadre de mon dataset il n’y avait aucunes données manquantes

**2. Transformation des Variables Catégorielles**

**Décision :** Les colonnes catégorielles ou binaires (par exemple, IsHTTPS, Bank, Crypto, etc.) ont été transformées en variables numériques pour être traitées par les modèles.

**Actions :**

* Pour les colonnes binaires (comme IsHTTPS, HasPasswordField), les valeurs 0 et 1 ont été conservées sans transformation supplémentaire.
* Pour les colonnes catégorielles non binaires (comme le **domaine de premier niveau** TLD), une méthode de **codage one-hot** a été utilisée. Cela consiste à créer une colonne pour chaque catégorie possible et à indiquer par un 1 ou un 0 si l'instance appartient à cette catégorie.

**Raisonnement :** Le codage one-hot est particulièrement utile pour les modèles de machine learning qui n’interprètent pas correctement les relations ordinales entre les catégories (par exemple, un modèle ne doit pas penser que .com est "plus grand" que .org). Cela permet d’éviter d’introduire des biais dans les modèles.

**3. Suppression des Caractéristiques Redondantes**

**Décision :** Certaines caractéristiques peuvent être redondantes, c'est-à-dire qu'elles apportent des informations similaires ou doublons. Afin d’éviter une surabondance d’informations similaires qui pourrait dégrader les performances du modèle (multicolinéarité), il a été décidé de supprimer certaines colonnes après analyse.

**Actions :**

* Nous avons utilisé une **analyse de la corrélation** pour identifier les caractéristiques fortement corrélées entre elles. Les paires de colonnes avec des corrélations supérieures à un seuil de 0,6 ont été examinées.
* Parmi ces colonnes corrélées, celles qui apportaient le moins de valeur ajoutée (en termes de compréhension du phishing) ont été supprimées. Par exemple, si deux colonnes étaient fortement corrélées, nous avons conservé celle qui était la plus intuitive pour la détection des URL malveillantes.

**Raisonnement :** La suppression des colonnes redondantes permet d’améliorer l'efficacité du modèle tout en réduisant le risque de surapprentissage (overfitting). De plus, cela simplifie le modèle, facilitant ainsi son interprétation et son utilisation en production.

**4. Traitement des Outliers (Valeurs Anormales)**

**Décision :** Les outliers peuvent fausser les résultats d’un modèle, en particulier dans les algorithmes sensibles aux valeurs extrêmes (comme la régression logistique ou les arbres de décision). Il a donc été décidé de traiter ces valeurs extrêmes pour éviter tout impact négatif sur le modèle.

**Actions :**

* Une **analyse des boîtes à moustaches (box plots)** a été effectuée pour chaque caractéristique numérique afin d’identifier les outliers (valeurs situées au-delà de 1,5 fois l’intervalle interquartile).
* Les outliers identifiés ont été directement supprimer. Car le dataset ne contenait que très peu de outliers et donc leurs suppressions n’auras pas un impact majeur sur les modèles.

**Raisonnement :** Les outliers peuvent déformer les prédictions du modèle et introduire des biais. En les supprimant on s’assure de seulement conserver les informations importantes du dataset.

**6. Équilibrage des Classes**

**Décision :** Le dataset présente un léger déséquilibre entre les URL légitimes et celles identifiées comme des URL de phishing (environ 100 945 instances de phishing contre 134 850 instances légitimes). Cet **imbalance de classe** peut nuire à la performance du modèle, notamment en réduisant sa capacité à identifier correctement les URL de phishing.

**Actions :**

* Il a été décidé d'utiliser une approche de **rééquilibrage automatique** lors de l’entraînement des modèles en activant l'option auto\_class\_weights dans BigQuery ML. Cette option ajuste les poids des classes de manière à compenser l’imbalance et à donner plus d’importance aux classes minoritaires.

**Raisonnement :** Le rééquilibrage des classes permet d'améliorer la précision globale du modèle, en particulier pour la classe minoritaire (les URL de phishing). Cela assure que le modèle ne privilégie pas la prédiction des URL légitimes simplement parce qu'elles sont plus nombreuses dans le dataset.

**Conclusion du Prétraitement**

Ces étapes de prétraitement nous ont permis de nettoyer et d'organiser le dataset de manière optimale pour l'entraînement des modèles de machine learning. Le choix des techniques de normalisation, d'équilibrage des classes, et de gestion des valeurs manquantes a été guidé par la nécessité de garantir que les modèles puissent apprendre efficacement à partir des données sans être influencés par des anomalies ou des redondances.

**Entraînement des Modèles**

Après le prétraitement et l’exploration des données, la prochaine étape consiste à entraîner plusieurs modèles d'apprentissage supervisé pour classifier les URL en tant que légitimes ou de phishing. Le but de cette phase est de tester divers algorithmes et de comparer leurs performances afin de choisir le modèle le plus performant et le mieux adapté à notre problématique.

**1. Choix des Modèles**

**Décision :** Pour maximiser les chances de trouver un modèle performant, plusieurs algorithmes d’apprentissage supervisé ont été testés. Les modèles sélectionnés couvrent une variété d’approches, allant des modèles linéaires aux modèles non linéaires et aux algorithmes d'arbres. Voici les modèles qui ont été choisis pour être testés :

* **Régression Logistique** : Ce modèle linéaire est souvent utilisé pour des tâches de classification binaire, comme ici. Il fournit une bonne première base pour comprendre si les caractéristiques sont linéairement séparables en phishing et légitimes.
* **Arbre de Décision Boosté (Boosted Decision Tree)** : Cet algorithme permet de modéliser des relations non linéaires et complexes entre les caractéristiques. Il est particulièrement efficace pour capturer les interactions entre variables.
* **Forêt Aléatoire (Random Forest)** : Ce modèle consiste en une multitude d'arbres de décision formés sur des sous-échantillons du dataset. Il est réputé pour sa robustesse et sa capacité à gérer des caractéristiques redondantes ou bruitées.
* **XGBoost (Extreme Gradient Boosting)** : Cette version améliorée des arbres de décision boostés est conçue pour offrir des performances élevées tout en gérant efficacement les grands datasets.
* **Classificateur DNN (Deep Neural Network)** : Bien qu'un modèle de réseau de neurones profonds (DNN) soit parfois plus complexe et gourmand en ressources, il peut capturer des patterns plus subtils dans les données, notamment des relations non linéaires complexes.

**2. Paramètres des Modèles**

**Décision :** Pour garantir que les modèles puissent être entraînés rapidement tout en fournissant des résultats représentatifs, des **hyperparamètres minimaux** ont été sélectionnés pour chaque modèle dans cette phase initiale. Cela permet de tester les modèles sans un coût de calcul excessif et de déterminer rapidement lesquels sont les plus prometteurs.

**Actions :**

* **Régression Logistique** : Nous avons activé auto\_class\_weights pour équilibrer les classes automatiquement et utilisé un nombre d'**itérations maximum de 10** pour accélérer l'entraînement.
* **Arbre de Décision Boosté** : Le nombre maximum d'itérations a également été fixé à **10**. Les itérations limitées permettent de tester rapidement le modèle tout en observant ses performances sur des sous-ensembles de données.
* **Forêt Aléatoire** : Le modèle a été configuré pour une **fraction de validation de 30 %** (ce qui signifie que 70 % des données sont utilisées pour l’entraînement et 30 % pour l’évaluation). Les paramètres par défaut de BigQuery ML pour la forêt aléatoire ont été conservés dans un premier temps.
* **XGBoost** : Le nombre d'itérations a été fixé à **10** et le même fractionnement des données (70 % entraînement, 30 % validation) a été appliqué. Ce modèle étant une version optimisée des arbres de décision boostés, il est particulièrement efficace avec ces hyperparamètres initiaux.
* **Classificateur DNN** : En raison de la complexité du DNN, une **seule couche cachée** avec **32 unités** a été configurée. Cela permet d’évaluer rapidement la performance du réseau sans entraîner des couches trop profondes, qui prendraient plus de temps à converger.

**3. Entraînement des Modèles**

**Décision :** L'entraînement des modèles a été réalisé directement au sein de Google BigQuery à l'aide de BigQuery ML. Cela permet de traiter le dataset volumineux sans avoir à l'exporter vers un autre environnement et de tirer parti de la puissance de calcul de Google Cloud pour un entraînement plus rapide.

**Actions :**

* Chaque modèle a été créé à l'aide de la commande SQL CREATE OR REPLACE MODEL fournie par BigQuery ML. Les modèles ont été configurés pour utiliser un **fractionnement aléatoire des données** (data\_split\_method = 'RANDOM') avec 30 % des données réservées pour l'évaluation après l'entraînement.

**Résultats :**

* Chaque modèle a été entraîné avec succès sur les données, et les temps d'entraînement ont été mesurés pour comparer leur efficacité. Par exemple, la régression logistique a été le modèle le plus rapide à s'entraîner en raison de sa simplicité, tandis que les modèles plus complexes comme XGBoost et le DNN ont pris plus de temps, comme attendu.

**4. Évaluation des Modèles**

**Décision :** L'évaluation des modèles est cruciale pour déterminer lequel offre la meilleure performance sur notre problématique de classification des URL de phishing. Plusieurs métriques ont été utilisées pour évaluer la qualité des modèles, notamment :

* **Précision (Accuracy)** : Le pourcentage d'URL correctement classées par le modèle.
* **Rappel (Recall)** : La capacité du modèle à identifier les URL de phishing (vrai positif).
* **F1-Score** : Une moyenne harmonique de la précision et du rappel, particulièrement utile lorsque les classes sont déséquilibrées.
* **AUC-ROC** : La zone sous la courbe ROC, qui mesure la capacité du modèle à séparer les URL de phishing des URL légitimes.

**Résultats :**

* Les résultats de l’évaluation ont montré des différences notables entre les modèles :
  + La **régression logistique** a donné une précision de base mais n'a pas excédé en termes de rappel, ce qui est attendu pour un modèle linéaire.
  + L'**arbre de décision boosté** et la **forêt aléatoire** ont donné des résultats meilleurs en termes de précision et de F1-score, indiquant qu’ils capturent mieux les relations complexes entre les caractéristiques.
  + Le **XGBoost** s’est avéré être le modèle le plus performant, avec une **AUC-ROC supérieure à 0,90**, indiquant une excellente capacité à différencier les URL légitimes des URL de phishing. Ce modèle a également obtenu un F1-score élevé, ce qui en fait un bon candidat pour la classification finale.

**Conclusion de l’Entraînement des Modèles**

Après avoir testé plusieurs modèles d'apprentissage supervisé, nous avons observé que le modèle **XGBoost** offre les meilleures performances globales sur notre dataset. Il présente un bon équilibre entre précision, rappel, F1-score et AUC-ROC, ce qui le rend particulièrement efficace pour la classification des URL de phishing.

L'utilisation de BigQuery ML a permis d'entraîner rapidement ces modèles sur un grand volume de données tout en bénéficiant de la puissance de Google Cloud. En fin de compte, le modèle XGBoost a été retenu comme modèle final en raison de ses excellentes performances.