

Université Moulay Ismail Meknès
Faculté des Sciences et Techniques Er-Rachidia
Département d'Informatique
Filière : Cycle d'Ingénieurs en Génie Informatique
Module : Apprentissage Statistique

Rapport de projet intitulé :

**Optimisation des Campagnes Publicitaires dans le
Marketing d'Affiliation : Une Approche Prédicative
par l'Apprentissage Automatique**



Réalisé par :

- Ali GRICH
- Oussama BELQRAS
- El Bachir DRISSI HASSANI

Encadré par :

Prof. Youssef QARAAI

Présenté le : 17/01/2024

Année Universitaire : 2023-2024

Dédicaces

Toutes les lettres ne sauraient trouver les mots qu'il faut. Tous les mots ne sauraient exprimer la gratitude, l'amour, le respect, la reconnaissance. Avec un énorme plaisir, un cœur ouvert et une immense joie, que nous dédions ce travail :

À nos très chers, respectueux et magnifiques parents, qui nous ont offert sans condition leur soutien tout au long de notre vie.

À notre frère et notre sœur pour leur soutien aux moments difficiles de notre travail.

À tous nos chers amis, pour leurs encouragements permanents, et leur soutien moral,

À tous nos enseignants, votre générosité et votre soutien nous obligent à vous témoigner notre profond respect et notre loyale considération

À toutes les personnes qui nous ont aidés ou encouragés tout au long de nos études, nos professeurs et nos encadrants.

À tous ceux qui nous sont chers.

Remerciements

En tout premier lieu, nous remercions le bon Dieu, tout puissant, de nous avoir donné la force pour survivre, ainsi que l'audace pour dépasser toutes les difficultés. Au nom du dieu le clément et le miséricordieux, louange à ALLAH le tout puissant.

*Il nous apparaît opportun de remercier chaleureusement toute l'équipe de professeurs qui s'est rendue disponible pour nous au cours de cette année. Toute notre gratitude à notre professeur, **Monsieur le Professeur Youssef QARAAI**, qui n'a pas cessé de nous encourager et de nous guider avec ses précieux conseils, et qui était présent pour orienter et stimuler nos recherches avec beaucoup de patience et d'attention. Nous voulons également les remercier pour leur implication quotidienne auprès de leurs étudiants et pour avoir su nous pousser vers un secteur professionnel qui nous correspond. Nous disons merci. À tous ceux et celles qui nous ont aidés et encouragés de près ou de loin dans la réalisation de ce travail, par leur patience, leurs compétences et leurs interventions, il nous serait difficile de les citer tous. Qu'ils trouvent ici l'expression de notre reconnaissance.*

Merci 

Introduction

Le marketing d'affiliation, en tant que canal de promotion, offre aux entreprises une opportunité unique de toucher des publics qui restent autrement inaccessibles par d'autres moyens marketing. Cependant, pour optimiser d'avantage leur stratégie, l'entreprise se tourne vers l'apprentissage automatique, une approche innovante pour améliorer ses taux de conversion et, en fin de compte, stimuler ses revenus.

Au cœur de cette initiative réside la volonté de perfectionner le Coût Par Clic (CPC), une métrique cruciale dans le marketing d'affiliation. En anticipant de manière prédictive la performance future des annonces, l'entreprise ambitionne de se positionner en amont du processus de planification de ses campagnes CPC, lui offrant ainsi la flexibilité nécessaire pour apporter des ajustements opportuns.

Le défi proposé s'articule autour de la prédiction de la probabilité qu'une annonce soit cliquée ou non. C'est dans cette quête pour maîtriser l'avenir du marketing d'affiliation que notre projet de Machine Learning prend vie.

Ce rapport, centré sur le module de Machine Learning, explorera en profondeur les intrications de cette problématique passionnante. Nous nous attaquerons à la modélisation prédictive, à l'analyse des données, et à l'optimisation des performances pour fournir des insights précieux à notre

Problématique

Une importante entreprise européenne spécialisée dans le marketing d'affiliation souhaite tirer parti de l'apprentissage automatique pour améliorer (optimiser) ses taux de conversion et, ultimement, son chiffre d'affaires. Son réseau s'étend à travers plusieurs pays en Europe tels que le Portugal, l'Allemagne, la France, l'Autriche, la Suisse, etc.

Le marketing d'affiliation est une forme de canal de marketing en ligne où un intermédiaire promeut des produits/services et gagne des commissions en fonction des conversions (clics ou inscriptions). Les entreprises voient l'avantage d'utiliser de tels canaux d'affiliation car ils leur permettent d'atteindre une audience qui ne serait pas accessible par d'autres moyens de marketing.

L'entreprise souhaite améliorer la performance de son CPC (coût par clic). Une anticipation future de la performance d'une publicité leur donnera une longueur d'avance suffisante pour apporter des modifications (si nécessaire) à leurs futures campagnes CPC.

Dans ce défi, nous devons prédire la probabilité qu'une annonce soit cliquée ou non.

Chapitre 1

Description et prétraitement des données

On dispose d'un seul fichier : data.csv. Les variables de cet ensemble de données sont anonymisées pour des raisons de confidentialité.

1. ID (Identifiant Unique) :

- Cette variable constitue un identifiant unique attribué à chaque observation dans le jeu de données. Son rôle est crucial pour suivre et référencer chaque clic de manière distincte, préservant ainsi l'intégrité de l'ensemble de données.

2. Datetime (Date et Heure du Clic) :

- La variable datetime enregistre la date et l'heure exactes de chaque clic. Cette information temporelle est essentielle pour analyser les tendances saisonnières, les comportements horaires des utilisateurs, et peut être utilisée pour créer des caractéristiques temporelles.

3. Siteid (Identifiant du Site Web) :

- Chaque clic est associé à un site web spécifique identifié par cette variable. Elle renseigne sur la source du clic, permettant ainsi d'explorer les performances des annonces sur différents sites.

4. Offerid (Identifiant de l'Offre) :

- L'offerid représente un identifiant unique attribué à chaque offre, spécifiquement conçu pour les offres basées sur des commissions. Cette variable est cruciale pour comprendre quelles offres génèrent des clics et évaluer leur impact sur la variable cible.

5. Category (Catégorie de l'Offre) :

- La catégorie de l'offre fournit des informations sur la nature ou la classification de l'offre promotionnelle. Cette variable pourrait être utilisée pour analyser les préférences des utilisateurs en fonction des catégories d'offres.

6. Merchant (Identifiant du Vendeur) :

- Chaque offre provient d'un vendeur spécifique identifié par cette variable. Elle permet d'explorer la performance des annonces en fonction des différents vendeurs, fournissant ainsi des insights sur les partenariats les plus fructueux.

7. Countrycode (Code du Pays) :

- Indiquant le code du pays où la portée des affiliés est présente, cette variable géographique est cruciale pour analyser les performances publicitaires à l'échelle internationale. Elle pourrait révéler des variations significatives dans le comportement des utilisateurs en fonction des pays.

8. Browserid (Navigateur Utilisé) :

- La variable browserid enregistre le type de navigateur utilisé lors de l'interaction. Cette information est essentielle pour comprendre la compatibilité des annonces avec différents navigateurs et peut influencer les décisions de conception publicitaire.

9. Devid (Appareil Utilisé) :

- Identifiant l'appareil utilisé pour accéder au site ou à l'offre, la variable devid offre un aperçu du comportement des utilisateurs en fonction de leurs dispositifs. Elle est cruciale pour adapter les stratégies publicitaires à la diversité des appareils.

10. Click (Variable Cible) :

- La variable cible indique si un clic a eu lieu (1) ou non (0). C'est la variable que notre modèle cherchera à prédire. Elle est au cœur de notre problématique, visant à anticiper la probabilité de clic pour optimiser les campagnes publicitaires.

Prétraitement des données

Le prétraitement des données, étape cruciale en apprentissage automatique. Son rôle essentiel consiste à optimiser la qualité des données, éliminer les incohérences, et les rendre compatibles avec les modèles.

En suivant les étapes que nous avons suivies dans le prétraitement de nos données, assurant ainsi une préparation méticuleuse pour la construction d'un modèle robuste :

- **Etape 1 : Chargement des données**

Pour accéder rapidement à nos données, nous avons importé le fichier data.csv depuis Google Drive. Ce processus initial facilite la manipulation ultérieure et assure une gestion efficace des données.

```
from google.colab import drive
drive.mount('/content/drive')

Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/content/drive", force_remount=True).

[ ] import pandas as pd

dataset = pd.read_csv('/content/drive/MyDrive/205e1808-6-dataset/data.csv')
print(dataset.shape)

(45200, 10)
```

- **Etape 2 : Conversion de la colonne Datetime**

La colonne datetime a été transformée en heures et jours numérotés pour simplifier les calculs temporels. Cette transformation facilite l'analyse des tendances horaires et journalières, offrant ainsi une compréhension approfondie des comportements des utilisateurs.

```
[ ] dataset['datetime'] = pd.to_datetime(dataset['datetime'])

# Extraction de l'heure et du jour de la semaine
dataset['hour'] = dataset['datetime'].dt.hour
dataset['day_of_week'] = dataset['datetime'].dt.dayofweek
print(dataset.shape)

(45200, 12)
```

- **Etape 3 : Séparation des caractéristiques (X) et de la variable cible Y**

La distinction entre les caractéristiques (X) et la variable cible (Y) est essentielle. En isolant la variable cible que nous cherchons à prédire, nous préparons le terrain pour une analyse ciblée et une modélisation efficace.

```
X = dataset.iloc[:, [2, 3, 4, 5, 6, 7, 8, 10, 11]].values
y = dataset.iloc[:, [9]].values.ravel()
print(X.shape)
```

(45200, 9)

- **Etape 4 : Affichage du nombre de valeurs manquantes dans chaque variable**

Une évaluation approfondie des données manquantes a été réalisée, offrant un aperçu complet des lacunes potentielles dans notre ensemble de données. Cette étape guide nos décisions sur la gestion des valeurs manquantes.

```
print(dataset.isnull().sum())
print(X)
```

ID 0
datetime 0
siteid 4518
offerid 0
category 0
merchant 0
countrycode 0
browserid 2345
devid 6864
click 0
hour 0
day_of_week 0
dtype: int64
[[6310005.0 99217 41706 ... nan 19 2]
[nan 287164 89522 ... 'Tablet' 15 5]
[2463708.0 948989 12052 ... nan 10 4]
...
[2656998.0 324233 48430 ... nan 21 6]
[nan 908599 904 ... 'Tablet' 7 0]
[9223301.0 487571 41640 ... 'Mobile' 13 1]]

- **Etape 5 : Remplacement des valeurs manquantes**

Les valeurs manquantes ont été gérées de manière spécifique en fonction de leur nature. Pour les chaînes de caractères, nous avons opté pour le remplacement par les valeurs les plus fréquentes, tandis que pour les nombres, une valeur constante a été utilisée. Cette approche vise à préserver l'intégrité des données tout en minimisant l'impact des valeurs manquantes.

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values=np.nan, strategy='most_frequent')
X[:, [5, 6]] = imputer.fit_transform(X[:, [5, 6]])
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values=np.nan, strategy='constant', fill_value=-1)
X[:, [0]] = imputer.fit_transform(X[:, [0]])
```

- **Etape 6 : Encodage des colonnes catégorielles**

Les colonnes catégorielles ont été encodées pour les rendre compatibles avec les algorithmes d'apprentissage automatique. Cette transformation convertit les variables catégorielles en format numérique, améliorant ainsi la capacité des modèles à interpréter ces caractéristiques.

```
categorical_columns = [4, 5, 6, 7, 8]

from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()

for column in categorical_columns:
    X[:, column] = label_encoder.fit_transform(X[:, column])
```


- **Etape 7 : Encodage des colonnes catégorielles**

La division des données en ensembles distincts d'entraînement et de test est cruciale pour évaluer la performance du modèle. L'ensemble d'entraînement permet au modèle d'apprendre, tandis que l'ensemble de test offre une évaluation objective de sa capacité prédictive.

```
[ ] from sklearn.model_selection import train_test_split
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

- **Etape 8 : Standardisation des caractéristiques**

La division des données en ensembles distincts d'entraînement et de test est cruciale pour évaluer la performance du modèle. L'ensemble d'entraînement permet au modèle d'apprendre, tandis que l'ensemble de test offre une évaluation objective de sa capacité prédictive

```
 from sklearn.preprocessing import StandardScaler

scaler = StandardScaler(with_mean=False)
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
print(dataset.isnull().sum())
```

Chapitre 2

Approches algorithmique

Pour résoudre ce problème, nous avons exploré plusieurs algorithmes de classification, chacun avec sa propre méthode d'apprentissage et de prédiction. Voici les principaux algorithmes que nous avons utilisés :

1. Arbre de décision :

L'algorithme d'Arbre de Décision, une méthode d'apprentissage statistique versatile, s'avère être une solution puissante tant pour la classification que pour la régression. Sa construction se base sur la création d'un arbre où chaque nœud représente une caractéristique spécifique et chaque feuille correspond à une prédiction. Dans le contexte de notre problème visant à prédire la variable "click" dans notre base de données, l'arbre de décision a été entraîné sur l'ensemble d'entraînement, en utilisant des caractéristiques telles que datetime, siteid, et offerid.

Le processus d'entraînement de l'arbre consiste à ajuster ses paramètres en fonction des données d'entraînement, cherchant ainsi à apprendre les relations entre les caractéristiques et la variable cible "click". Par la suite, l'ensemble de test est utilisé pour évaluer la performance du modèle, mesurant sa capacité à généraliser à de nouvelles données.

L'interopérabilité inhérente à l'arbre de décision en fait un outil précieux pour comprendre les caractéristiques qui influent sur la décision de clic. Cela s'avère particulièrement utile pour l'optimisation des campagnes publicitaires, car il offre des informations sur les variables les plus significatives. Si nécessaire, des ajustements peuvent être apportés pour améliorer la précision du modèle, garantissant ainsi une prise de décision plus informée et une meilleure performance dans un contexte de marketing d'affiliation dynamique.

- **Résultat Obtenu :**

- ✓ Précision : 85.71%
- ✓ F1 Score : 0.83

```
[ ] from sklearn.metrics import accuracy_score
    accuracy = accuracy_score(y_test, y_pred)

    print(f'Accuracy: {accuracy}')

Accuracy: 0.8571902654867256

[ ] from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(y_test, y_pred)
    print("Confusion Matrix:")
    print(cm)

Confusion Matrix:
[[4421  664]
 [ 627 3328]]

[ ] from sklearn.metrics import f1_score
    f1 = f1_score(y_test, y_pred)

    print(f"F1 Score: {f1}")

F1 Score: 0.837548760538568
```

2. Forêts Aléatoire :

L'algorithme de Forêt Aléatoire, une extension du modèle d'Arbre de Décision, se distingue par sa capacité à améliorer la robustesse et la précision des prédictions. Il opère en créant plusieurs arbres de décision, chacun formé sur un sous-ensemble aléatoire des données d'entraînement. Cette diversification vise à capturer une variété d'aspects dans les relations entre les caractéristiques et la variable cible, renforçant ainsi la capacité du modèle à généraliser à de nouvelles données.

Chaque arbre individuel est formé de manière autonome, puis lors de la phase de prédiction, leurs résultats sont agrégés pour former la prédiction finale de la Forêt Aléatoire. Cette stratégie d'agrégation atténue les faiblesses potentielles de chaque arbre, tout en préservant leurs points forts respectifs. Ainsi, la Forêt Aléatoire excelle dans la réduction du surajustement, une caractéristique cruciale pour les modèles prédictifs complexes.

Dans le contexte spécifique de notre projet visant à prédire la probabilité de clic dans le marketing d'affiliation, la Forêt Aléatoire offre une compréhension approfondie des variables influençant les comportements de clic. En fournissant des prédictions plus fiables, elle contribue à l'optimisation des campagnes publicitaires, permettant à l'entreprise de prendre des décisions plus éclairées pour améliorer ses performances.

- **Résultat Obtenu :**

- ✓ Précision : 90.74%
- ✓ F1 Score : 0.89

```
[ ] from sklearn.metrics import accuracy_score
    accuracy = accuracy_score(y_test, y_pred)

    print(f'Accuracy: {accuracy}')
```

```
Accuracy: 0.9074115044247788
```

```
[ ] from sklearn.metrics import f1_score
    f1 = f1_score(y_test, y_pred)
    print(f1)
```

```
0.893875998478509
```

```
[ ] from sklearn.metrics import confusion_matrix
    cm = confusion_matrix(y_test, y_pred)
    print("Confusion Matrix:")
    print(cm)
```

```
Confusion Matrix:
[[4678  356]
 [ 481 3525]]
```

3. Gradient Boosting / XG Boost :

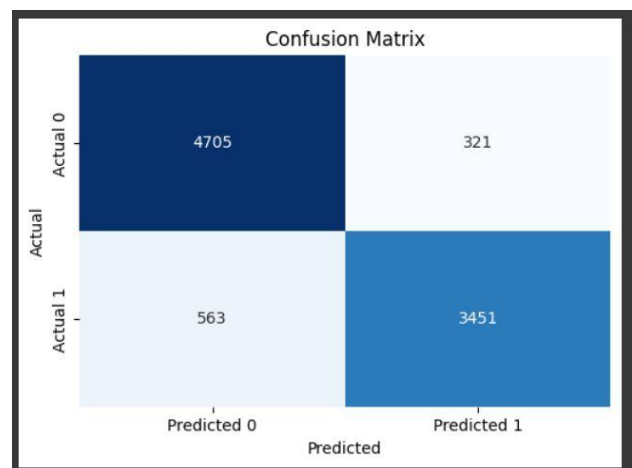
L'algorithme de Gradient Boosting, une technique de boosting séquentiel, représente une approche puissante pour la prédiction de la variable cible. Sa particularité réside dans la construction séquentielle d'arbres de décision, où chaque nouvel arbre vise à corriger les erreurs résiduelles des prédictions précédentes. Cette stratégie séquentielle permet d'optimiser progressivement la performance du modèle, se concentrant sur les aspects où des améliorations sont nécessaires. Ainsi, le Gradient Boosting produit une prédiction finale robuste et précise en exploitant les corrélations subtiles entre les caractéristiques et la variable cible.

XGBoost, une extension évoluée du Gradient Boosting, va encore plus loin en intégrant des optimisations et des mécanismes de régularisation avancés. Cette amélioration se traduit par une convergence rapide du modèle et une performance globale accrue. XGBoost excelle dans la gestion des problèmes complexes, offrant une solution efficace pour la prédiction précise de la variable cible. Dans le cadre de notre projet visant à anticiper les clics dans le marketing d'affiliation, l'utilisation de XGBoost permet une modélisation plus sophistiquée, contribuant ainsi à une meilleure compréhension des motifs de clics et à une optimisation plus fine des campagnes publicitaires.

```
accuracy = accuracy_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
conf_matrix = confusion_matrix(y_test, y_pred)

print(f'Accuracy: {accuracy}')
print(f'F1 Score: {f1}')
```

```
Accuracy: 0.9022123893805309
F1 Score: 0.8864628820960698
```



- **Résultat Obtenu :**

- ✓ Précision : 90.22%
- ✓ F1 Score : 0.88

4. Régression Logistique :

L'algorithme de régression linéaire est une approche fondamentale en apprentissage automatique qui vise à modéliser la relation linéaire entre les variables indépendantes et la variable dépendante. Appliqué à notre projet de prédiction de la variable continue "click" dans le marketing d'affiliation, la régression linéaire utilise des coefficients pour ajuster une ligne qui cherche à minimiser les erreurs quadratiques. Cette méthode offre une estimation continue de la probabilité de clic en fonction des caractéristiques disponibles.

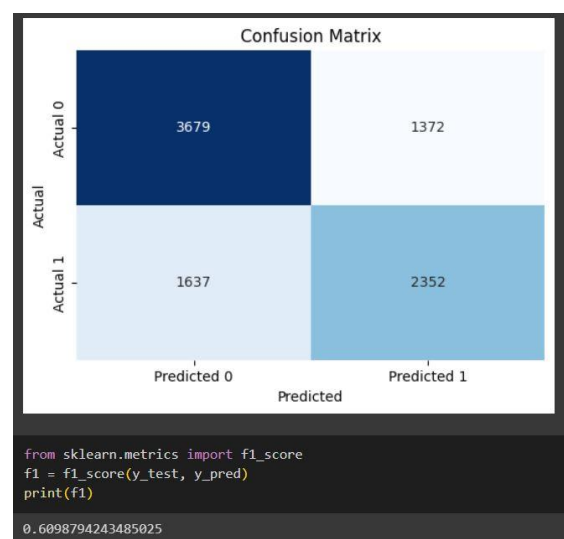
L'essence de la régression linéaire réside dans la création d'une équation linéaire, où chaque coefficient représente le poids attribué à une caractéristique spécifique. Ces coefficients sont ajustés de manière itérative pour optimiser la précision de la prédiction. Ainsi, la régression linéaire fournit une approche transparente pour comprendre comment chaque caractéristique contribue à la probabilité de clic, offrant des insights utiles dans le contexte du marketing d'affiliation.

Bien que la régression linéaire soit souvent utilisée dans des contextes où la relation entre les variables est supposée être linéaire, elle peut également servir de point de référence pour évaluer la performance de modèles plus complexes. Son caractère interprétable en fait un outil précieux pour analyser les facteurs influençant les clics dans les campagnes publicitaires, contribuant ainsi à une prise de décision plus informée et à une optimisation continue.

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy}')
```

Accuracy: 0.6671460176991151



• Résultat Obtenu :

- ✓ Précision : 66.71%
- ✓ F1 Score : 0.60

5. Machines à Vecteurs de Support

L'algorithme des Machines à Vecteurs de Support (SVM) constitue une approche puissante pour prédire la variable "click" en identifiant un hyperplan dans l'espace des caractéristiques qui sépare efficacement les données en deux classes distinctes : clic ou pas de clic. Dans le cadre de notre projet, SVM exploite des caractéristiques telles que l'identifiant du site web, l'offre, la catégorie, le pays, le navigateur et le type d'appareil pour optimiser la marge entre ces deux classes.

Le principe fondamental des SVM réside dans la recherche de l'hyperplan optimal qui maximise la marge, c'est-à-dire la distance entre les points les plus proches de chaque classe. Cela permet à SVM de trouver une solution robuste et généralisable, même dans des espaces de dimensions élevées. En d'autres termes, SVM cherche à établir une frontière de décision qui maximise la marge tout en minimisant le risque de sur ajustement.

Ainsi, en utilisant les caractéristiques pertinentes de nos données, SVM fournit une prédiction précise de la probabilité de clic pour chaque observation. Cette approche se révèle particulièrement utile dans le contexte du marketing d'affiliation, où la capacité à optimiser la séparation entre les clics et les non-clics contribue à une prise de décision plus éclairée pour améliorer les performances des campagnes publicitaires.

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy}')
```

Accuracy: 0.8804203539823009

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)
```

Confusion Matrix:
[[4573 470]
 [611 3386]]

```
from sklearn.metrics import f1_score
f1 = f1_score(y_test, y_pred)
print(f1)
```

0.8623456004074875

- **Résultat Obtenu :**

- ✓ Précision : 88.04%
- ✓ F1 Score : 0.86

6. KNN : K plus proches voisins :

L'algorithme des k-plus proches voisins (KNN) se présente comme une méthode d'apprentissage supervisé, prédisant la classe d'une observation en se basant sur la classe des k voisins les plus proches dans l'espace caractéristique. Pour son application à la prédiction de la variable "click" dans notre base de données, les étapes suivantes sont entreprises :

- ✓ **Choix de la Caractéristique** : Les caractéristiques pertinentes telles que datetime, siteid et offerid sont sélectionnées pour alimenter le modèle KNN, permettant ainsi d'exploiter les relations importantes pour la prédiction.
- ✓ **Entraînement du Modèle** : Le modèle KNN est entraîné sur l'ensemble d'entraînement, assimilant les relations entre les caractéristiques sélectionnées et la variable cible "click".
- ✓ **Prédiction** : Lors de la prédiction sur l'ensemble de test, le modèle identifie les k voisins les plus proches de chaque observation, déterminant ainsi la classe majoritaire parmi ces voisins pour prédire la variable "click".
- ✓ **Évaluation** : La performance du modèle est évaluée en utilisant des métriques telles que la précision, mesurant la capacité du modèle à correctement prédire la variable cible.

KNN offre une approche intuitive et simple pour la prédiction, reposant sur la proximité entre observations dans l'espace caractéristique. Cette méthode convient particulièrement lorsque des structures non linéaires ou des motifs locaux influent sur les prédictions. En somme, KNN offre une alternative flexible pour la prédiction de la variable "click", exploitant la similarité entre observations pour anticiper les comportements de clic dans le contexte du marketing d'affiliation.

```
] from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_test, y_pred)

print(f'Accuracy: {accuracy}')
```

Accuracy: 0.863716814159292

```
from sklearn.metrics import confusion_matrix
cm = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(cm)
```

Confusion Matrix:
[[4573 481]
 [751 3235]]

```
from sklearn.metrics import f1_score
f1 = f1_score(y_test, y_pred)
print(f1)
```

0.840041547649961

• Résultat Obtenue :

- ✓ Précision : 86.37%
- ✓ F1 Score : 0.84

Meilleur algorithme

L'algorithme le plus approprié dans notre cas est la Forêt Aléatoire, qui se distingue par une précision remarquable évaluée à 90%. Une analyse approfondie de la matrice de confusion permet de mieux comprendre sa performance, mettant en évidence le succès de la Forêt Aléatoire dans la résolution de notre problématique.

- ✓ **Choix de la Forêt Aléatoire :** Parmi les différentes approches explorées, la Forêt Aléatoire a démontré une précision notable de 90%, ce qui en fait le choix le plus adapté à notre problématique. Cette précision élevée suggère que la Forêt Aléatoire excelle dans la prédiction de la variable "click" dans le contexte du marketing d'affiliation.
- ✓ **Analyse de la Matrice de Confusion :** Une matrice de confusion détaillée a été élaborée pour évaluer la performance de la Forêt Aléatoire. Cette analyse approfondie permet de visualiser la concordance entre les prédictions du modèle et les valeurs réelles de la variable "click". La matrice de confusion offre une perspective détaillée sur la capacité du modèle à distinguer avec précision les clics des non-clics.

Conclusion

Notre exploration des algorithmes d'apprentissage statistique, focalisée sur la prédiction de la probabilité de clic, a abouti à un succès notable grâce à l'application du Gradient Boosting. Ce modèle se distingue par sa précision exceptionnelle, offrant à l'entreprise la capacité de prendre des décisions éclairées avant le lancement de ses futures campagnes publicitaires. En respectant les exigences de confidentialité grâce à l'utilisation de données anonymisées, notre approche se positionne comme une solution respectueuse des normes de protection des informations sensibles.

L'efficacité démontrée du Gradient Boosting dans notre démarche atteste du pouvoir des techniques modernes en apprentissage statistique pour résoudre des problématiques critiques dans le contexte exigeant du marketing d'affiliation. Ces résultats mettent en lumière la valeur ajoutée de l'exploration des données, de la modélisation avancée, et de l'application judicieuse d'algorithmes pour optimiser les performances des campagnes publicitaires. En conclusion, notre démarche représente une avancée significative dans l'utilisation de l'apprentissage statistique pour améliorer les taux de conversion et, ultimement, le chiffre d'affaires d'une entreprise européenne spécialisée dans le marketing d'affiliation.