



République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche
Scientifique



Université des Sciences et de la Technologie Houari Boumediene

Faculté d'Informatique
Département Intelligence Artificielle et Sciences des Données

Filière : Informatique
Spécialité : SII

Rapport

Projet de data mining : Prédiction des Incendies de
Forêt

Réalisé par :
Haddadi Mohamed Ramzi
Mami Anfal-Wafa

Table des matières

Table des matières	i
Liste des figures	ii
Liste des tableaux	iv
1 Introduction	1
1.0.1 fire dataset	2
1.0.2 Land Cover	5
1.0.3 Soil	10
1.0.4 Elevation	15
1.0.5 Climat	18
1.0.6 Fusionnement des datasets	21
1.0.7 Sampling	23
1.0.8 Definition Communes	25
1.0.9 Knn	27
1.0.10 Normalization et Encodage de données	27
1.0.11 Balancement de données	28
1.0.12 Réglage des paramètres	30
1.0.13 Réduction de dimensionnalité	30
1.0.14 Comparaison de resultats	30
1.0.15 Decision Tree	32
1.0.16 Normalization et Encodage de données	32
1.0.17 Balancement de données	32
1.0.18 Réglage des paramètres	33
1.0.19 Réduction de dimensionnalité	35
1.0.20 Comparaison de resultats	35
1.0.21 Random Forest	37
1.0.22 Normalization et Encodage de données	37
1.0.23 Balancement de données	37
1.0.24 Réglage des paramètres	40
1.0.25 Réduction de dimensionnalité	41
1.0.26 Comparaison de resultats	41
2 Conclusion	43

Table des figures

1.1	valeurs manquantes.	3
1.2	Boxplot.	3
1.3	Description de l'image.	4
1.4	boxplot	6
1.5	q-q plots	7
1.6	scatter plot	7
1.7	distribution de log_area	8
1.8	distribution de gridcode	8
1.9	distribution de lccode	8
1.10	matrice de corrélation	9
1.11	Nombres de valeurs manquante avant et après le nettoyage.	11
1.12	Histogramme des valeurs numériques	12
1.13	Boite à moustache des valeurs numériques	13
1.14	Matrice de corrélation	14
1.15	Carte de l'élévation de l'Algérie et la Tunisie	15
1.16	Histogramme des valeurs d'élévation	16
1.17	Boite à moustache des valeurs d'élévation	17
1.18	boxplot.	19
1.19	matrice de corrélation.	20
1.20	dataset fire apres sampling.	24
1.21	KNN avec données originales	28
1.22	KNN avec données random under sampling	28
1.23	KNN avec données random over sampling	29
1.24	KNN avec données smote over sampling	29
1.25	KNN avec données tomek under sampling	30
1.26	KNN - Final Results with no PCA	31
1.27	KNN - Final Results with PCA	31
1.28	Decision Tree avec données originales	33
1.29	Decision Tree avec données random under sampling	33
1.30	Decision Tree avec données random over sampling	34
1.31	Decision Tree avec données smote over sampling	34
1.32	Decision Tree avec données tomek under sampling	34
1.33	Decision Tree - Final Results with no PCA	36
1.34	Decision Tree - Final Results with PCA	36
1.35	Random Forest avec données originales	38
1.36	Random Forest avec données random under sampling	38
1.37	Random Forest avec données random over sampling	39
1.38	Random Forest avec données smote over sampling	39

1.39	Random Forest avec données tomek under sampling	40
1.40	Random Forest - Feature Importance	41
1.41	Random Forest - Final Results with no PCA	42
1.42	Random Forest - Final Results with PCA	42

Liste des tableaux

1.1	Statistiques descriptives des variables.	6
1.2	Statistiques descriptives des attributs numériques (arrondies à 2 décimales) .	12
1.3	Statistiques descriptives des valeurs d'élévation.	16
1.4	Statistiques descriptives finales des variables climatiques	19
1.5	Comparaison des performances avec et sans réduction de dimensionnalité . .	31
1.6	Comparaison des performances avec et sans réduction de dimensionnalité . .	31
1.7	Comparaison des performances avec et sans réduction de dimensionnalité . .	35
1.8	Comparaison des performances avec et sans réduction de dimensionnalité . .	35
1.9	Comparaison des performances avec et sans réduction de dimensionnalité . .	41
1.10	Comparaison des performances avec et sans réduction de dimensionnalité . .	42

Chapitre 1

Introduction

Les incendies de forêt constituent un défi environnemental et socio-économique majeur, entraînant la perte de végétation, la dégradation des sols et de graves dommages écologiques. Dans le cadre de ce projet de Data Mining, l'objectif est de développer un modèle prédictif capable d'anticiper l'occurrence des feux en s'appuyant sur des données environnementales riches et variées. La zone d'étude se concentre sur l'Algérie et la Tunisie pour l'année 2024, en fusionnant des données provenant de sources spatiales et terrestres de référence. Le projet adopte une approche structurée en trois phases : l'analyse et le prétraitement des données, l'application d'algorithmes d'apprentissage supervisé pour la prédiction, et l'utilisation de l'apprentissage non-supervisé (clustering) pour identifier des zones à haut risque. Cette démarche vise non seulement à atteindre une précision technique, mais aussi à fournir des outils d'aide à la décision pour les stratégies de prévention et de gestion des incendies.

problématique

Face à l'intensification des risques climatiques, la question centrale de cette étude est la suivante : Comment l'intégration et l'analyse de données hétérogènes (climatiques, pédologiques et géospatiales) peuvent-elles permettre de modéliser et de prédire avec précision l'occurrence des incendies de forêt en Algérie et en Tunisie ? Cette problématique se décline en plusieurs défis techniques et scientifiques :

- L'influence des variables : Dans quelle mesure les caractéristiques du sol (humidité, carbone organique, texture USDA) et les facteurs climatiques (température, précipitations) interagissent-ils pour favoriser un départ de feu ?
- Le défi des données : Comment traiter et fusionner efficacement des jeux de données aux résolutions et formats disparates (rasters d'élévation, shapefiles d'occupation du sol de la FAO, points de feu de la NASA) ?

1.0.1 fire dataset

Defintiton

Le dataset utilisé regroupe les événements d'incendies détectés en 2024 pour l'Algérie et la Tunisie, issus du service FIRMS de la NASA. Il est basé sur les observations du capteur VIIRS embarqué sur le satellite NOAA-20 et comprend l'ensemble des points de feu enregistrés au cours de l'année 2024 dans ces deux pays. Chaque enregistrement comporte des coordonnées géographiques (longitude/latitude) ainsi que des informations associées à l'occurrence d'un feu détecté par satellite.

chargement des données

Nous avons d'abord chargé le CSV brut contenant toutes les colonnes (longitude, latitude, type de feu, `bright_ti4`, `satellite`, etc.). Ensuite, nous avons importé le shapefile des frontières et filtré les polygones afin de ne conserver que ceux correspondant à l'Algérie et à la Tunisie.

- **Pandas** : lecture des fichiers CSV bet rechargement du dataset nettoyé au format Parquet
- **GeoPandas** : lecture du shapefile mondial , filtrage pour l'Algérie et la Tunisie
- **SciPy** : recherche spatiale optimisée pour associer rapidement chaque point de feu à la cellule de grille la plus proche.
- **NumPy** : création d'une grille régulière de coordonnées lon/lat (résolution 0.01°) et manipulation des matrices 2D/1D.
- **Shapely** : gestion des géométries vectorielles via GeoPandas pour créer automatiquement les points.

Nettoyage des données

- **Suppression des colonnes inutiles** : `satellite`, `instrument`, `version`, `scan`, `track` (métadonnées non pertinentes pour la modélisation).
- **Renommage des colonnes** : longitude → `lon`, latitude → `lat`, type → `fire`.
- **Sélection des colonnes essentielles** : conservation uniquement de `lon`, `lat` et `fire`; suppression des autres variables (`bright_ti4`, `confidence`, etc.).
- **Filtrage des événements de feu** : conservation des feux réels (`fire == 0`), suppression des autres classes (1, 2, 3); conversion de la classe 0 en 1 pour indiquer la présence de feu.
- **Suppression des valeurs manquantes** : élimination des lignes contenant des NaN.

Pretraitement et analyse des donnees

- **Création d'une grille régulière (échantillonnage spatial)** :
 - Résolution : 0.01° (1,1 km à l'équateur)
 - Génération des coordonnées longitude/latitude avec `np.arange()` et `np.meshgrid()`
 - Application d'un buffer spatial pour couvrir entièrement la zone d'étude
- **Découpage spatial (clipping)** :
 - Conversion de la grille en GeoDataFrame (points)
 - Clipping avec `gpd.clip()` pour ne conserver que les cellules à l'intérieur des frontières d'Algérie et de Tunisie
 - Réduction significative du nombre de cellules (grille complète → grille clippée)

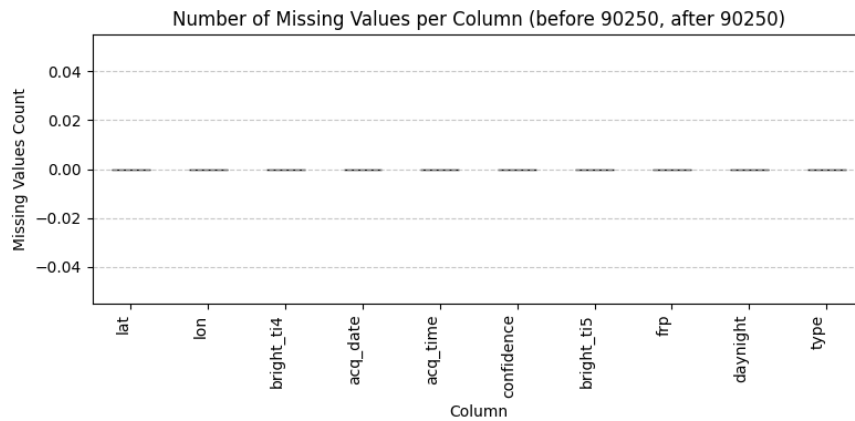


FIGURE 1.1 – valeurs manquantes.

- **Labélisation des cellules (Fire vs No-Fire)**
 - Construction d'un index spatial (`cKDTree`) sur les points de feu.
 - Requête spatiale avec une tolérance égale à la moitié de la résolution de la grille (0.005°).
 - Attribution des labels :
 - Cellules situées à moins de 0.005° d'un feu \rightarrow `fire` = 1
 - Cellules sans feu dans le rayon \rightarrow `fire` = 0
- Analyse univariée : calcul de la moyenne et autres statistiques descriptives pour chaque variable.
- Analyse bivariée : étude des relations entre variables et calcul de la corrélation.

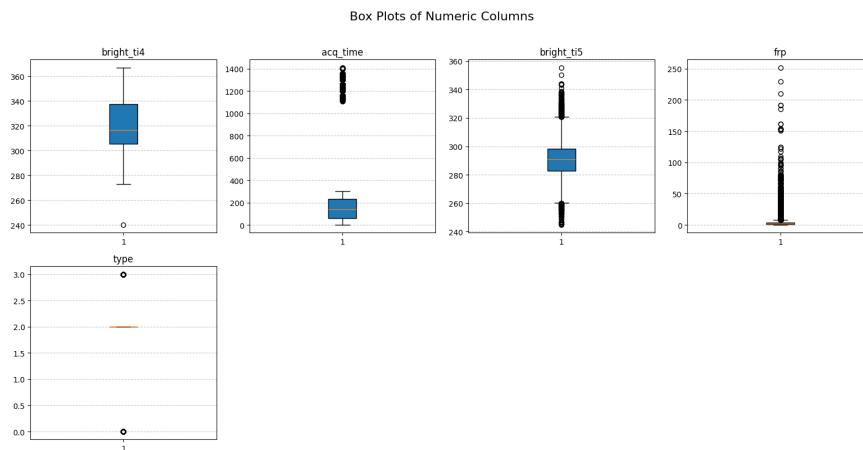


FIGURE 1.2 – Boxplot.

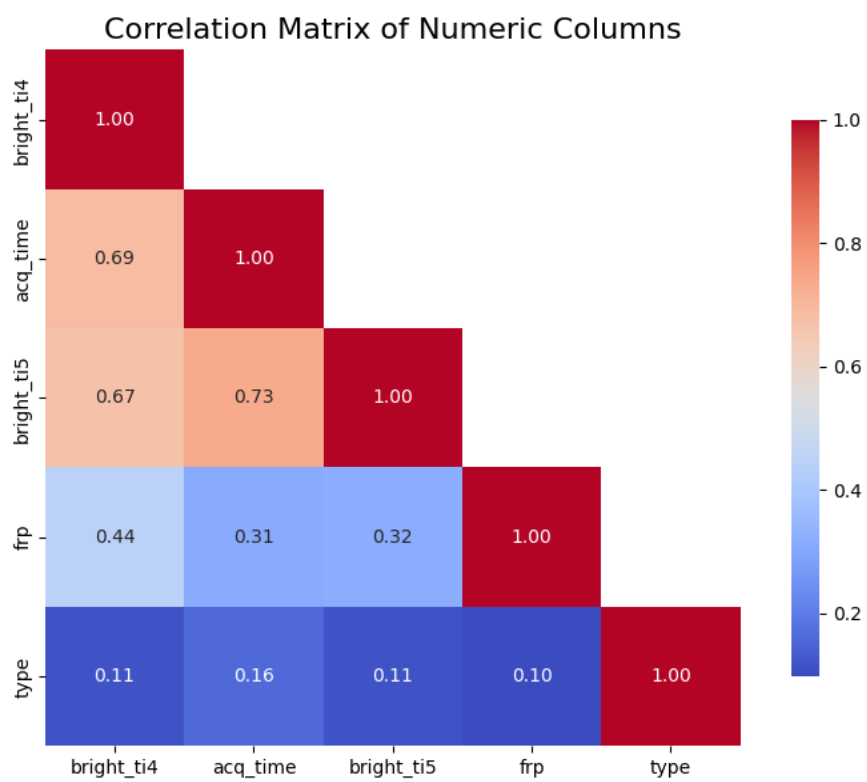


FIGURE 1.3 – Description de l'image.

1.0.2 Land Cover

Définition

Le dataset utilisé se compose de deux jeux de données distincts correspondant aux versions vectorielles nationales du produit GlobCover (2005) pour l'Algérie et la Tunisie, développé initialement par l'Agence Spatiale Européenne (ESA).

Cette étude porte sur l'analyse des données d'occupation des sols (Land Cover) pour ces deux régions géographiques. L'objectif principal est de manipuler des données géospatiales au format Shapefile afin d'extraire des informations statistiques pertinentes et de préparer un jeu de données structuré et propre, destiné à des analyses ultérieures ou à des applications de visualisation.

Lecture des données

La première étape a consisté à importer les bibliothèques nécessaires au traitement de données

- **GeoPandas** : manipulation et analyse de données géospatiales vectorielles.
- **Pandas** : gestion, fusion et analyse des données tabulaires.
- **NumPy** : calculs numériques et transformations mathématiques.
- **Matplotlib** : visualisation graphique et cartographique.
- **Seaborn** : visualisations statistiques avancées.
- **SciPy** : analyses statistiques et tests de normalité.
- **scikit-learn** : encodage des variables catégorielles.

Le système de coordonnées de référence (CRS) a été vérifié pour les deux jeux de données et est défini sur EPSG :4326 (WGS 84).

Nettoyage des données

Une analyse initiale a permis d'identifier des incohérences et des valeurs manquantes dans plusieurs colonnes clés :

- Algérie : valeurs nulles dans les champs **AREA** et **LCCCODE**.
- Tunisie : valeurs nulles dans les champs **AREA_M2** et **LCCCcode**.

Pour assurer l'homogénéité des données :

- Les colonnes de superficie ont été unifiées sous le nom **area_sqm**.
- Les codes de classification de l'occupation du sol ont été harmonisés sous **lcc_code**.

Les shapefiles ont ensuite été fusionnés en un seul jeu de données couvrant l'ensemble de la zone d'étude, suivi d'une suppression des doublons.

Prétraitement et analyse des données

Pour l'attribut **area_sqm**, un traitement des valeurs aberrantes a été appliqué :

- Transformation logarithmique pour réduire l'asymétrie des distributions.
- Détection et traitement des valeurs extrêmes via la méthode de l'écart interquartile (IQR).

de plus, une correction géométrique a été réalisée pour garantir la validité topologique des entités spatiales.

- **Analyse univariée** : calcul des indicateurs principaux (moyenne, médiane, minimum, maximum) pour chaque variable numérique.

- **Repérage des anomalies** : identification des valeurs inhabituelles et visualisation à l'aide d'histogrammes et de boîtes à moustaches.

	log_area_sqm	GRIDCODE	pays
count	438513.0	438513.0	438513
unique	-	-	2
top	-	-	Algérie
freq	-	-	386454
mean	12.9033	156.2099	-
std	0.9660	65.6615	-
min	11.5152	14.0	-
25%	12.0597	150.0	-
50%	12.6450	200.0	-
75%	13.4245	200.0	-
max	15.4718	210.0	-

TABLE 1.1 – Statistiques descriptives des variables.

Boîtes à Moustaches - Analyse ciblée

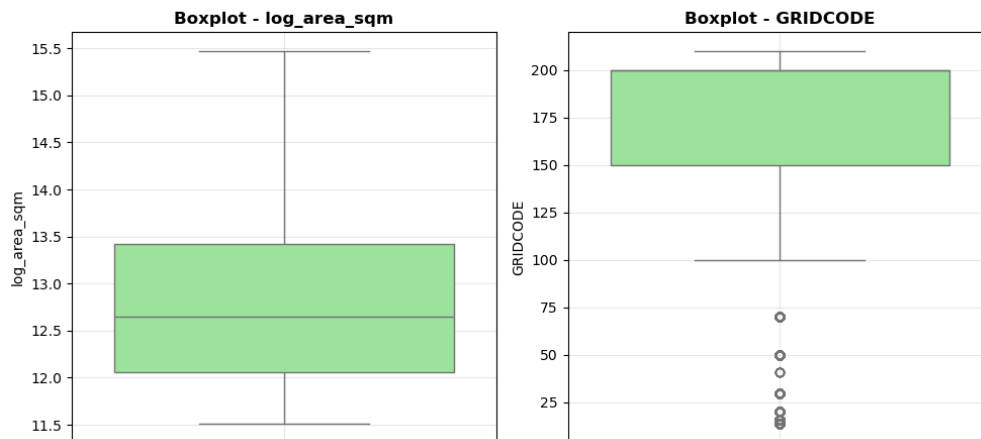


FIGURE 1.4 – boxplot

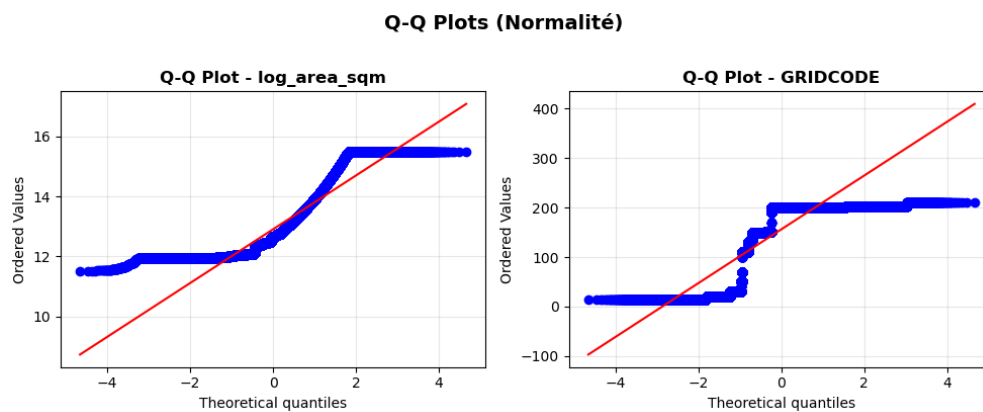


FIGURE 1.5 – q-q plots

Scatter Plots (log_area_sqm vs GRIDCODE)

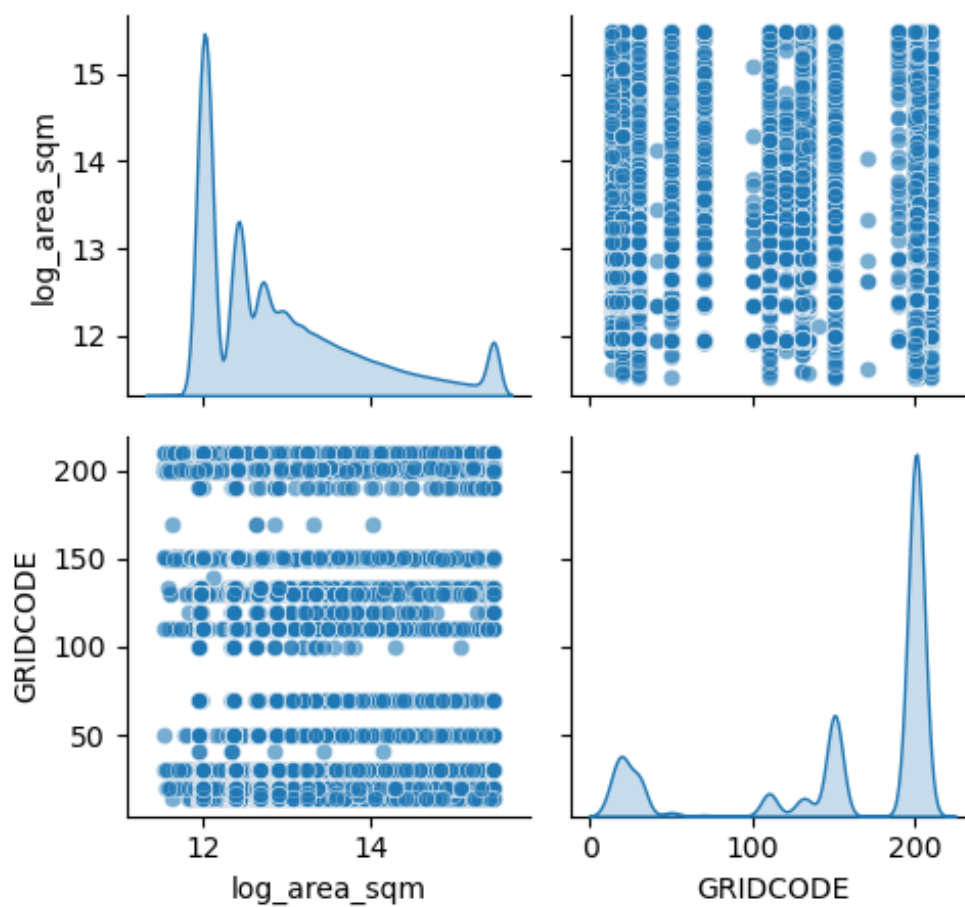


FIGURE 1.6 – scatter plot

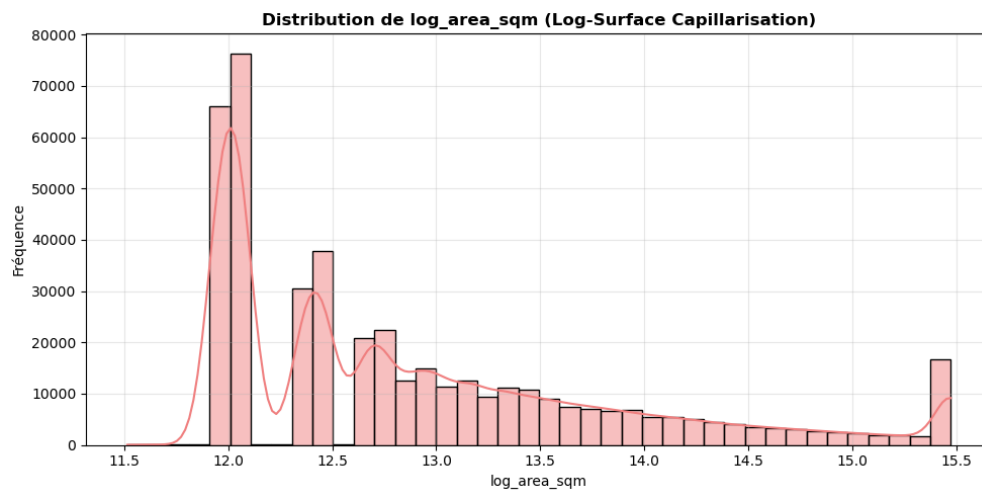


FIGURE 1.7 – distribution de log_area

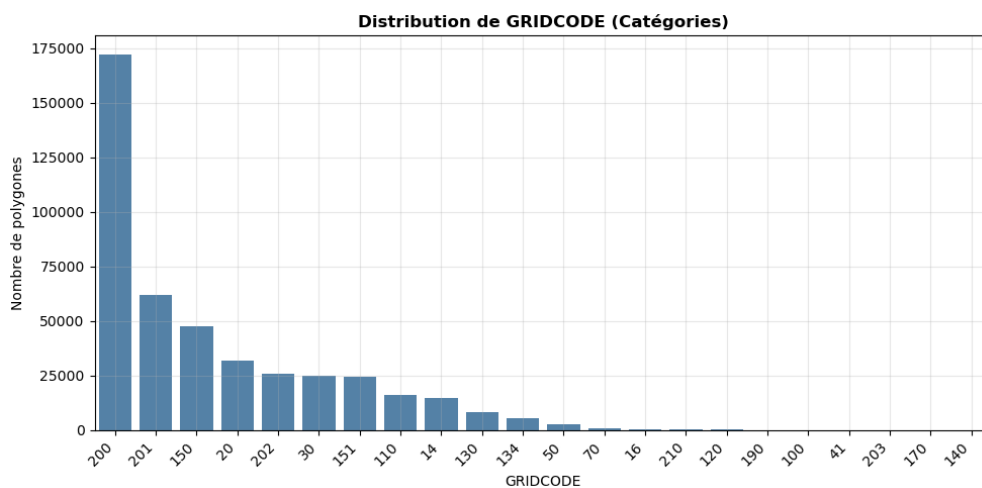


FIGURE 1.8 – distribution de gridcode

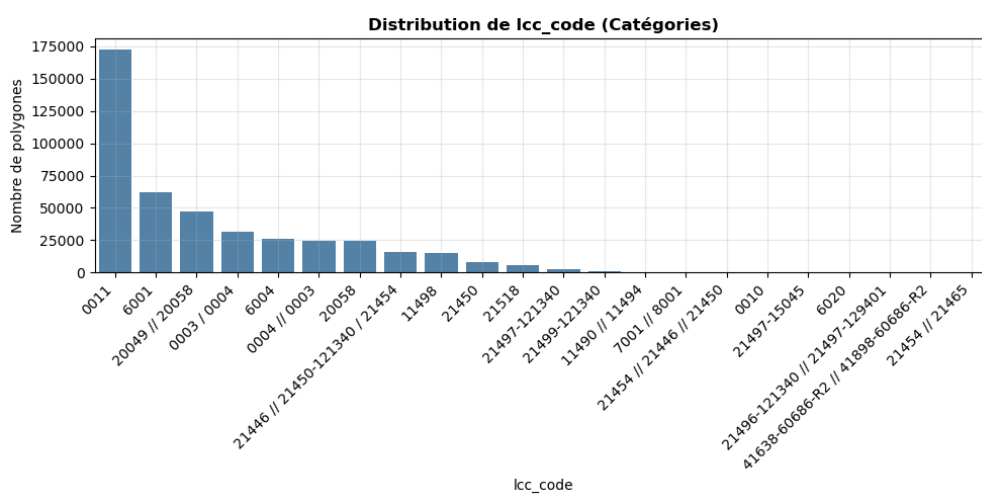


FIGURE 1.9 – distribution de lccode

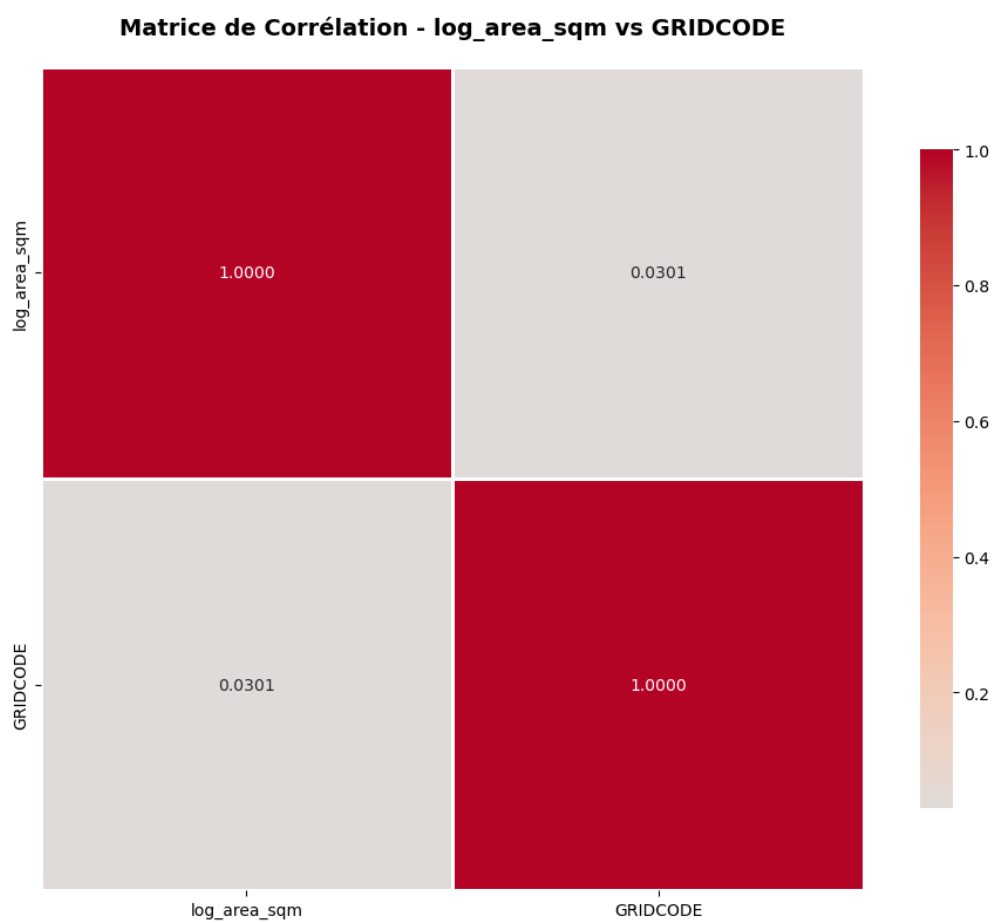


FIGURE 1.10 – matrice de corrélation

1.0.3 Soil

Definition

Le dataset du sol est constitué d'une base de données contenant plusieurs tables. La table la plus importante est la table **layers**, qui contient des enregistrements sur les propriétés des différentes couches de sol dans les zones d'Algérie et de Tunisie.

Chaque enregistrement couvre une zone et comporte plusieurs attributs, parmi lesquels :

- **HWSD2_SMU_ID** l'identifiant de l'enregistrement
- le niveau de décomposition
- le pourcentage de carbone
- le pH de l'eau
- la texture du sol (sable, limon, argile)
- la teneur en humidité
- la densité apparente
- et d'autres caractéristiques physico-chimiques du sol

Cette table permet donc d'analyser la composition et la qualité des sols selon différentes régions et couches. Dans notre étude, nous nous intéressons au niveau **D1** de la décompositions.

Il existe également un fichier HWSD2.bil, qui est un raster couvrant la carte du monde, où chaque pixel représente un point appartenant à une zone de sol spécifique.

Nous utilisons un troisième dataset contenant des fichiers de forme des pays afin de récupérer uniquement les identifiants des sols correspondant à l'Algérie et à la Tunisie. Ensuite, nous combinons tous les points du raster avec leurs enregistrements correspondants en utilisant l'identifiant SMU comme clé de jointure.

À la fin de ce processus, nous obtenons un dataset complet des propriétés du sol pour chaque point situé en Algérie et en Tunisie.

Chargement des données

Nous utilisons les bibliothèques suivantes pour charger et manipuler le dataset du sol :

- **pyodbc** : pour le chargement des tables depuis la base de données.
- **rasterio et rioxarray** : pour ouvrir et manipuler les fichiers raster.
- **geopandas** : pour ouvrir les fichiers de forme des pays et extraire les contours, ainsi que pour sélectionner les zones correspondant à l'Algérie et à la Tunisie.
- **pandas et numpy** : pour la manipulation des données, le prétraitement et l'analyse.
- **matplotlib** : pour la visualisation des graphes d'analyse.

À l'issue de ce processus, nous obtenons un dataset complet des propriétés du sol pour chaque point situé en Algérie et en Tunisie.

Nettoyage des données

Le nettoyage des données a été réalisé afin de préparer le dataset pour l'analyse. Les étapes principales incluent :

- Pour les attributs numériques, les valeurs manquantes sont codées par des valeurs négatives et sont remplacées par la moyenne des points adjacents selon les coordonnées longitude et latitude.
- Pour les attributs catégoriques, les valeurs manquantes sont représentées par le caractère '-' et sont remplacées par le mode des points voisins.

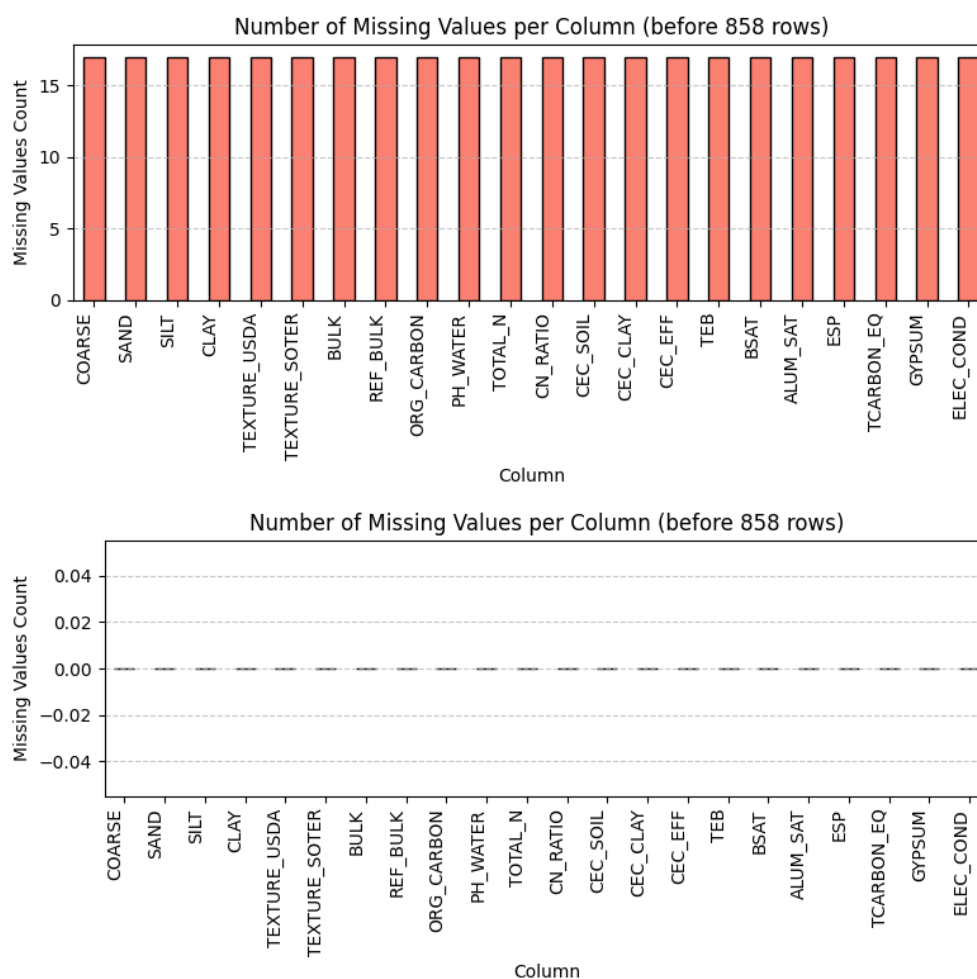


FIGURE 1.11 – Nombres de valeurs manquante avant et après le nettoyage.

Prétraitement et analyse des données

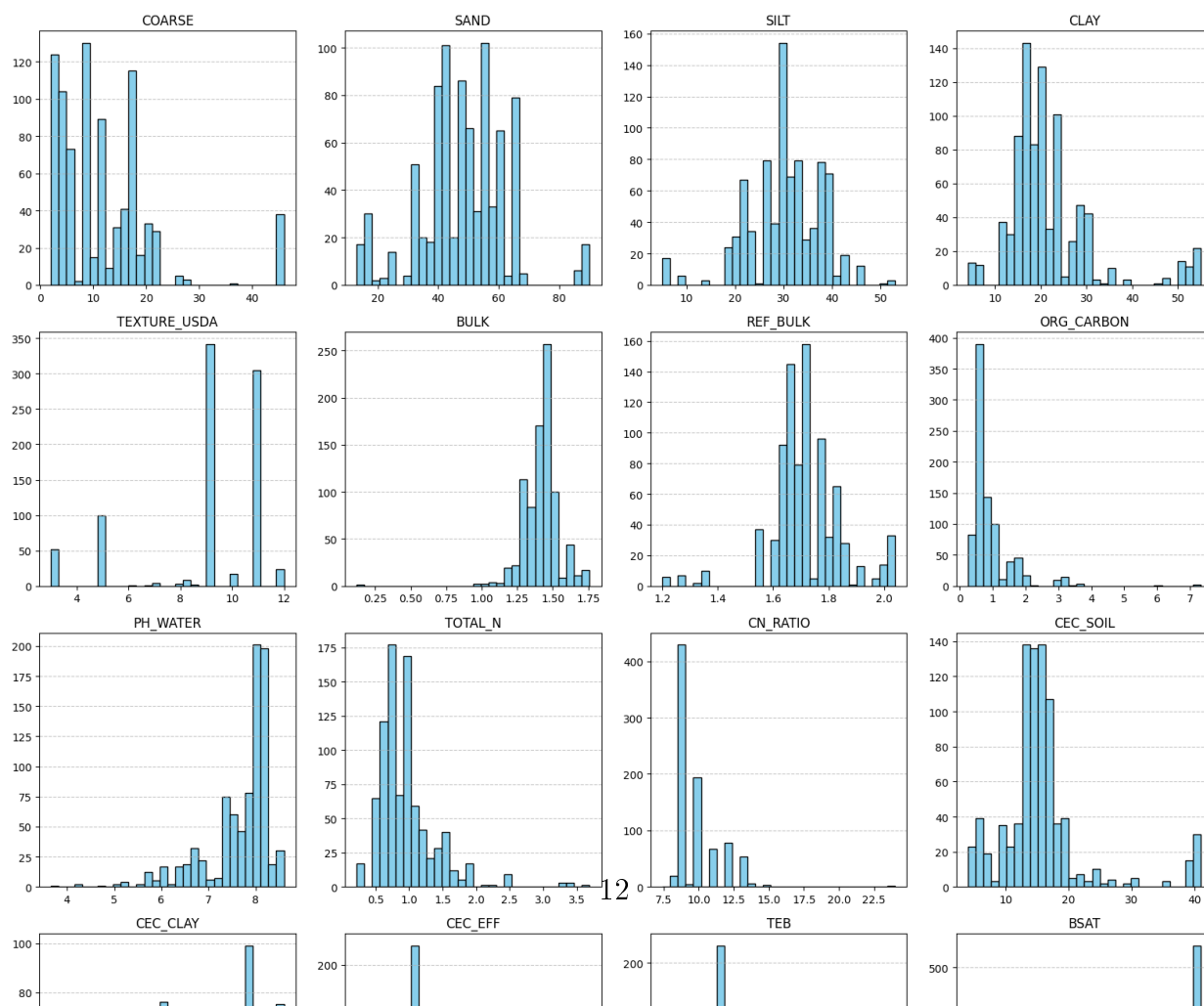
Après le nettoyage, les données ont été prétraitées et analysées selon les étapes suivantes :

- Analyse univariée : calcul des statistiques descriptives pour chaque attribut numérique, telles que la moyenne, la médiane, le minimum et le maximum...
- Détection des valeurs aberrantes : identification des valeurs extrêmes et visualisation à l'aide d'histogrammes et de boîtes à moustache.

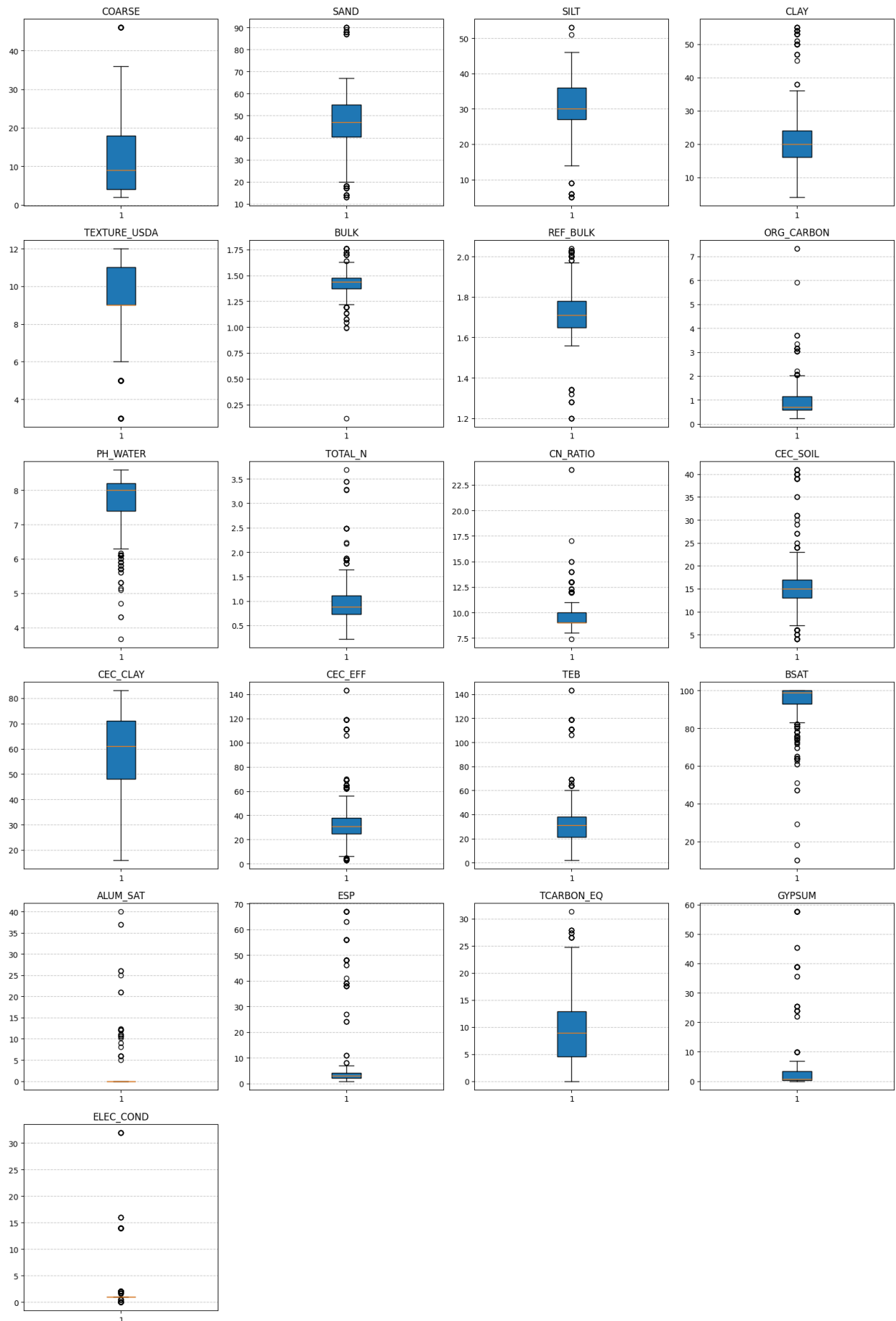
TABLE 1.2 – Statistiques descriptives des attributs numériques (arrondies à 2 décimales)

Attribut	Count	Mean	Std	Min	25%	50%	75%	Max
COARSE	858.0	12.08	9.69	2.0	4.0	9.0	18.0	46.0
SAND	858.0	47.95	14.23	13.0	40.42	47.0	55.0	90.0
SILT	858.0	30.22	7.6	5.0	27.0	30.0	36.0	53.0
CLAY	858.0	21.74	9.67	4.0	16.0	20.0	24.0	55.0
TEXTURE_USDA	858.0	8.95	2.4	3.0	9.0	9.0	11.0	12.0
BULK	858.0	1.43	0.12	0.12	1.37	1.44	1.48	1.76
REF_BULK	858.0	1.71	0.13	1.2	1.65	1.71	1.78	2.04
ORG_CARBON	858.0	0.93	0.67	0.24	0.59	0.7	1.16	7.33
PH_WATER	858.0	7.68	0.7	3.66	7.4	8.0	8.2	8.6
TOTAL_N	858.0	0.96	0.43	0.22	0.73	0.88	1.12	3.69
CN_RATIO	858.0	9.98	1.52	7.4	9.0	9.0	10.0	24.0
CEC_SOIL	858.0	15.75	7.11	4.0	13.0	15.0	17.0	41.0
CEC_CLAY	858.0	59.29	14.67	16.0	48.0	61.0	71.0	83.0
CEC_EFF	858.0	36.81	22.98	3.0	25.0	31.0	38.0	143.0
TEB	858.0	35.82	23.65	2.0	21.0	31.0	38.0	143.0
BSAT	858.0	93.23	11.79	10.0	93.0	99.0	100.0	100.0
ALUM_SAT	858.0	0.58	3.46	0.0	0.0	0.0	0.0	40.0
ESP	858.0	6.92	13.52	0.88	2.0	3.0	4.0	67.0
TCARBON_EQ	858.0	9.31	6.69	0.0	4.5	8.9	12.9	31.3
GYPSUM	858.0	4.45	10.97	0.0	0.3	0.6	3.3	57.6
ELEC_COND	858.0	2.19	4.43	0.0	1.0	1.0	1.0	32.0

Unique Values per Column



Box Plots of Numeric Columns



- Analyse de corrélation : calcul et affichage de la matrice de corrélation avant et après suppression des valeurs fortement corrélées.

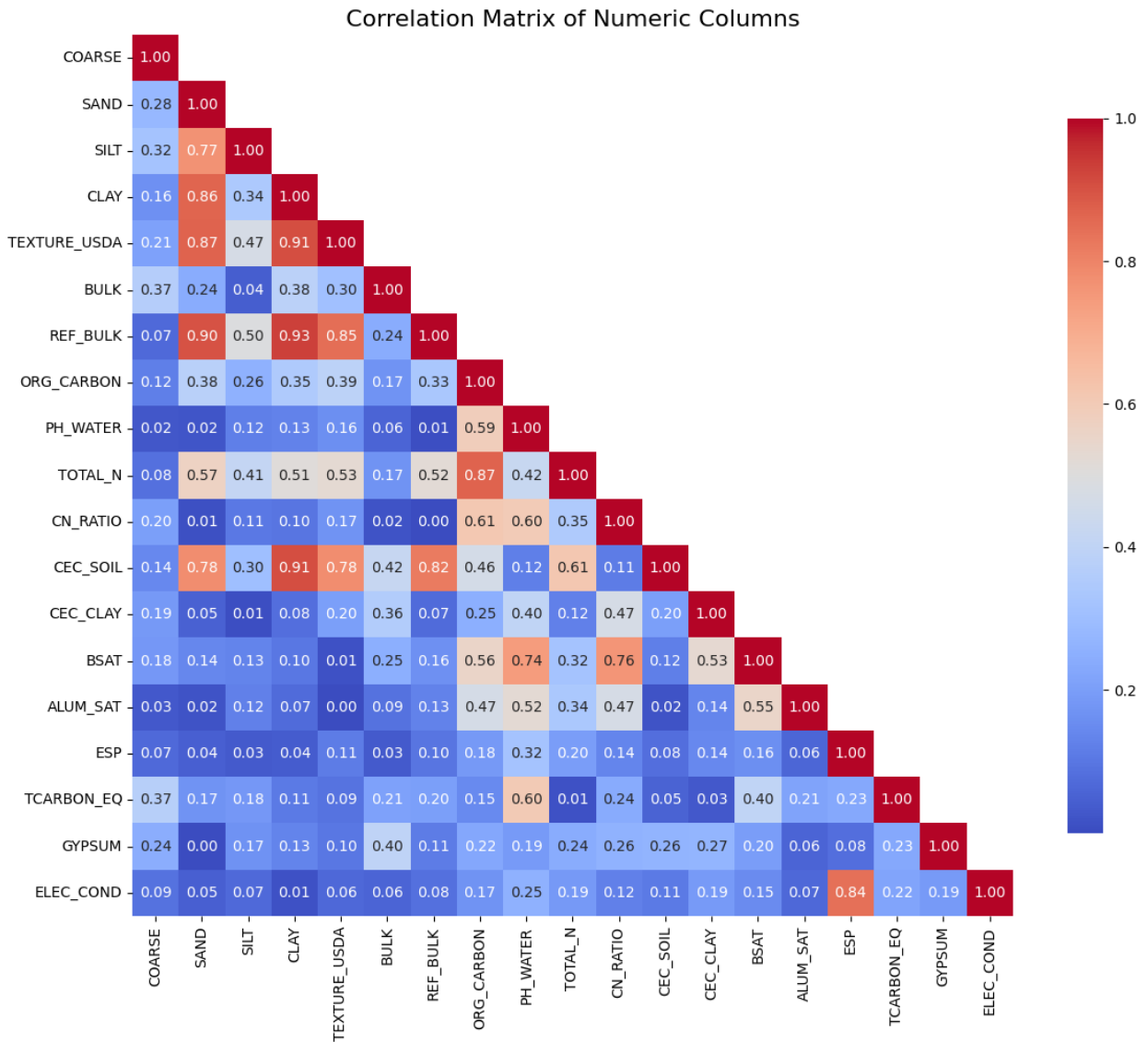


FIGURE 1.14 – Matrice de corrélation

- Exportation des données nettoyées : sauvegarde du dataset final sous forme de fichier .parquet.

1.0.4 Elevation

Defintion

Le dataset d'élévation **be15_grd** est constitué d'un seul raster, où la valeur de chaque pixel représente l'élévation à cet emplacement.

Ce raster est structuré sous la forme d'un dossier contenant plusieurs fichiers **.adf**, qui composent ensemble un raster au format ESRI Grid couvrant toute la carte du monde.

Pour convertir les coordonnées d'un pixel en longitude et latitude, nous utilisons les systèmes de référence définis par la norme EPSG, notamment l'EPSG :4326 (WGS84), qui décrit la manière standard de représenter les coordonnées géographiques sur la Terre en longitude et latitude.

Chargement des données

Nous utilisons les bibliothèques suivantes pour charger et manipuler les datasets de l'élévation :

- **rioxarray** : pour ouvrir et manipuler les fichiers raster.
- **geopandas** : pour ouvrir les fichiers de forme des pays et extraire les contours, ainsi que pour sélectionner les zones correspondant à l'Algérie et à la Tunisie.
- **matplotlib** : pour la visualisation des graphes d'analyse.

Ainsi, comme pour le dataset du sol, nous obtenons au final une liste des valeurs d'élévation pour chaque point en Algérie et en Tunisie.

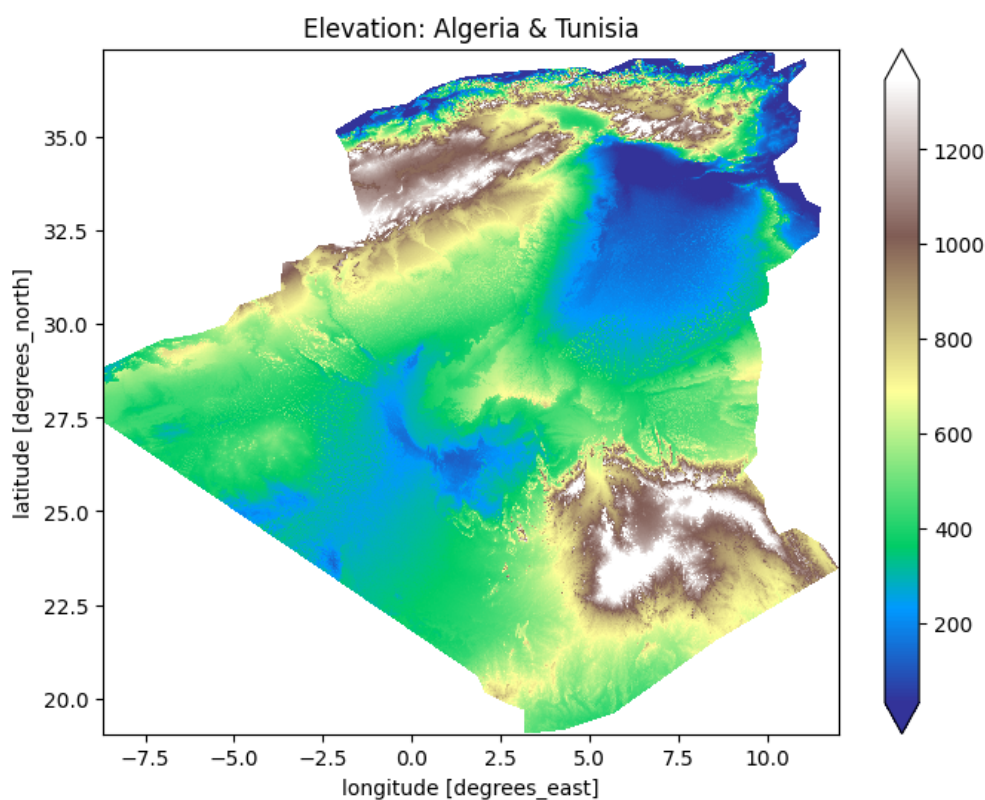


FIGURE 1.15 – Carte de l'élévation de l'Algérie et la Tunisie

Prétraitement de analyse de données

Comme il n'y a pas de données manquantes ou aberrantes, nous passons directement au prétraitement et analyse :

- **Analyse univariée** : calcul des statistiques descriptives pour l'attribut de l'élévation.

Statistique	Valeur
Count	13 183 993
Mean	536.37
Std	325.48
Min	-872
25%	310
50%	463
75%	697
Max	2877

TABLE 1.3 – Statistiques descriptives des valeurs d'élévation.

- **Distribution de données** : visualisation de la distribution des valeurs de l'élévation à l'aide d'histogrammes et de boîtes à moustache.

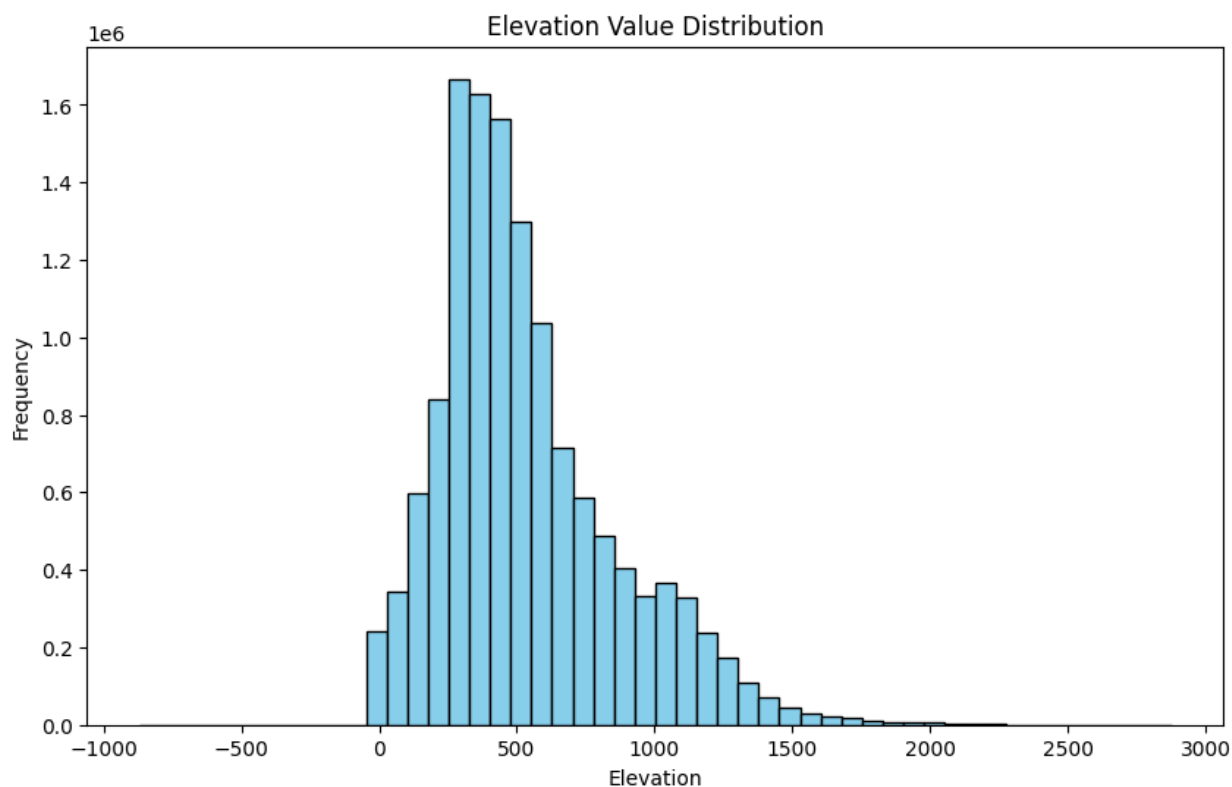


FIGURE 1.16 – Histogramme des valeurs d'élévation

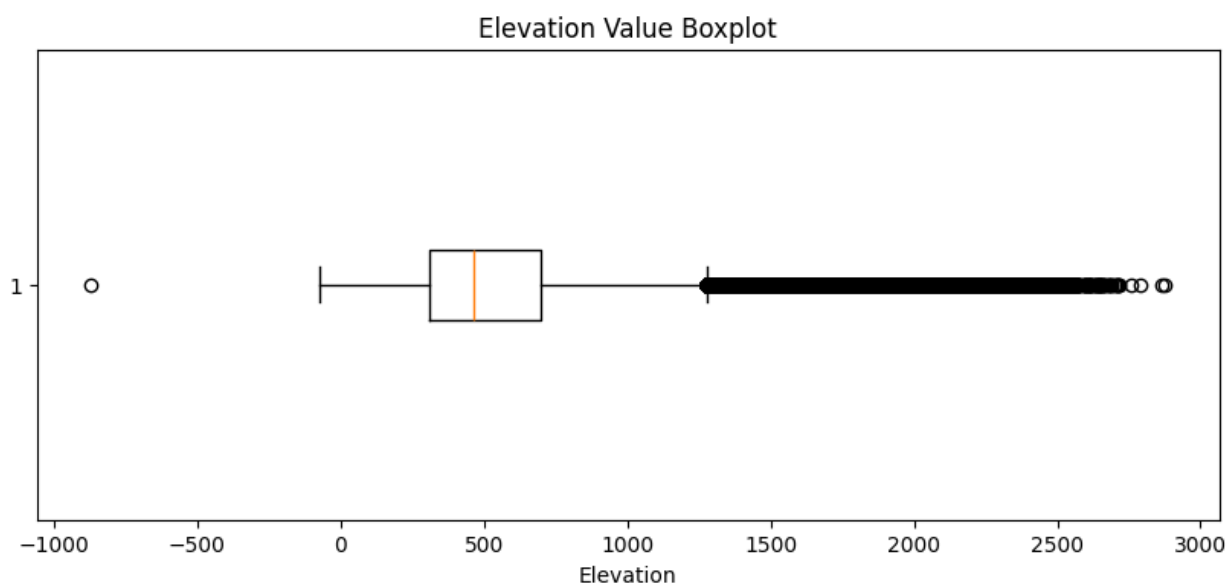


FIGURE 1.17 – Boîte à moustache des valeurs d'élévation

1.0.5 Climat

Defintion

Le jeu de données climatique utilisé provient de la base *WorldClim*, une base de données climatique mondiale largement utilisée en sciences de l'environnement. Il fournit des variables climatiques mensuelles, notamment la température et les précipitations, à une résolution spatiale de 5 minutes d'arc (environ 10 km). Ces données sont issues de l'interpolation de mesures provenant de stations météorologiques à l'échelle mondiale et sont adaptées aux analyses climatiques, écologiques et géospatiales à grande échelle.

Chargement de données

L'analyse repose sur plusieurs bibliothèques Python complémentaires.

- **GeoPandas** est utilisé pour lire le fichier Land Cover au format GeoPackage et servir de masque spatial pour le découpage des données climatiques.
- **rioxarray** constitue l'outil principal pour la lecture, le découpage spatial et la sauvegarde des rasters climatiques au format `.tif`.
- **xarray** permet la gestion de données raster multidimensionnelles (temps, espace) et l'ouverture simultanée de plusieurs fichiers climatiques.
- **Rasterio** est employé pour une lecture bas-niveau des rasters et l'accès aux méta-données lors des analyses univariées.
- **Pandas** est utilisé pour la gestion des dates et la création de DataFrames destinés aux analyses statistiques.

Les rasters climatiques (précipitations, température minimale et maximale) sont découpés à l'aide d'un masque spatial issu du fichier Land Cover, afin de restreindre l'étude à la zone couvrant l'Algérie et la Tunisie.

Netoyage de données

Le nettoyage des données vise à garantir la qualité, la cohérence et la fiabilité du jeu de données climatique. Les étapes principales sont :

- Suppression des colonnes techniques non pertinentes issues des formats raster, telles que `band` et `spatial_ref`.
- Détection, quantification et élimination des valeurs manquantes pour éviter tout biais dans les analyses.
- Identification et suppression des doublons afin d'assurer l'unicité des observations.
- Standardisation des noms de colonnes spatiales (`x` renommé en `longitude` et `y` en `latitude`).
- Réinitialisation de l'index pour obtenir un DataFrame plat et directement exploitable.

Pretraitement et analyse des donnees

Le prétraitement des données a pour objectif d'améliorer la distribution des variables et d'optimiser le jeu de données . Une transformation logarithmique est appliquée aux précipitations afin de réduire l'asymétrie de leur distribution. Une variable dérivée, appelée amplitude thermique, est créée à partir de la différence entre la température maximale et minimale pour mieux représenter la variabilité thermique. Une forte corrélation entre les variables de température est observée, conduisant à la suppression de la température minimale afin de réduire

la multicolinéarité. Le jeu de données final est ainsi limité à trois variables pertinentes : les précipitations transformées, la température maximale et l'amplitude thermique. Les valeurs aberrantes sont détectées à l'aide de la méthode de l'IQR sans être supprimées, car elles peuvent correspondre à des événements climatiques extrêmes réels.

Les données sont ensuite explorées à travers des analyses univariées et bivariées.

TABLE 1.4 – Statistiques descriptives finales des variables climatiques

Statistique	log_precip	tmax	amplitude_thermique
Count	148,292	148,292	148,292
Mean	1.937	27.828	13.428
Std	1.035	8.223	2.213
Min	0.000	5.250	5.500
25%	1.082	21.000	12.000
50%	1.825	28.000	13.750
75%	2.733	34.750	15.000
Max	5.588	48.000	20.250

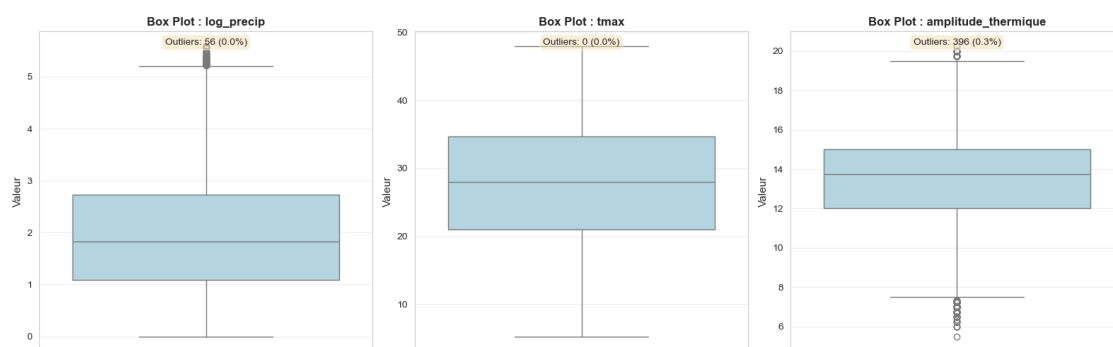


FIGURE 1.18 – boxplot.

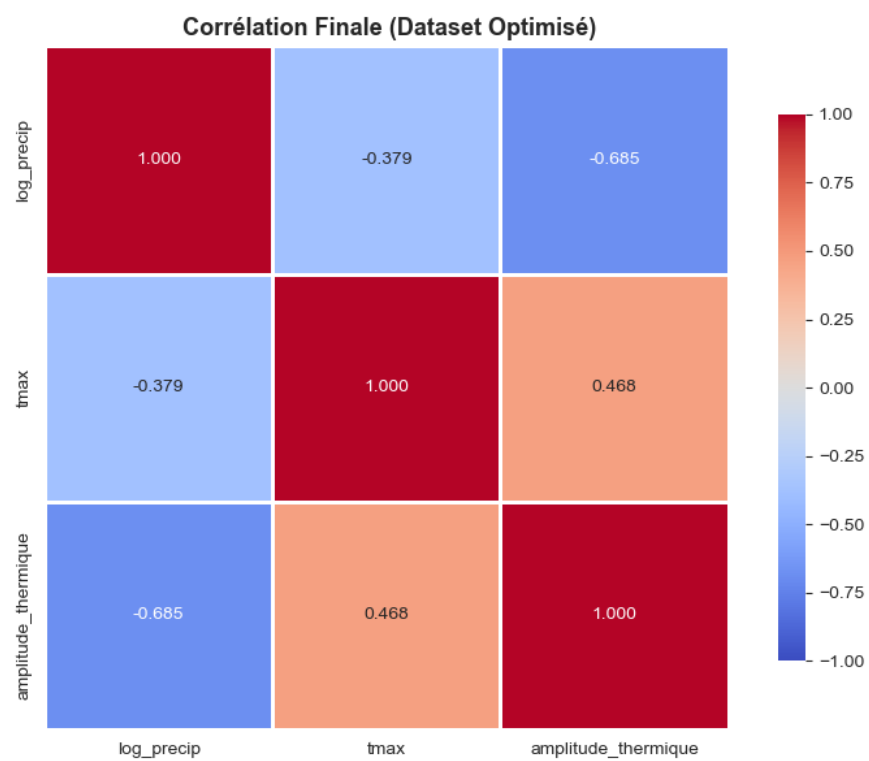


FIGURE 1.19 – matrice de corrélation.

1.0.6 Fusionnement des datasets

À la fin du traitement et du nettoyage des différents datasets, il est nécessaire de les combiner en un seul dataset contenant l'ensemble des attributs de chacun, afin de l'utiliser dans la phase d'entraînement. Le fusionnement se fait sur les attributs longitude et latitude, car ce sont les informations communes qui permettent d'identifier et de relier les mêmes points géographiques entre les différents jeux de données.

Pour réaliser cette fusion, nous appliquons une jointure à gauche (`left join`) entre le dataset des feux et les autres datasets. Ce choix garantit que tous les points liés aux feux sont conservés, et qu'aucune observation contenant la valeur à prédire — l'existence ou non d'un feu — n'est perdue lors de l'opération de fusion.

Fusionnement des données Fire Land Cover

Les points Fire ont été associés aux données climatiques via un **matching spatial par plus proche voisin** utilisant `cKDTree` de SciPy. Les données climatiques, organisées en grille régulière, ont été agrégées par saison : somme pour les précipitations (`log_precip`) et moyenne pour les températures (`tmax`, `amplitude_thermique`). Chaque point de feu a ainsi reçu les valeurs climatiques de la cellule la plus proche, assurant une correspondance spatiale précise et une couverture complète.

Fusionnement des données avec Climat

L'intégration des données Land Cover a combiné **buffer spatial** et **KDTree**. Les points Fire+Climat ont été convertis en `GeoDataFrame` et projetés en EPSG :4326. Un buffer de 0.005° (500 m) autour de chaque point a permis d'assigner les attributs des polygones Land Cover via un `spatial join`. Pour les points hors polygones, le land cover du polygone le plus proche a été attribué via `KDTree`. Les duplications ont été résolues en conservant uniquement la première correspondance, garantissant précision et couverture complète.

Fusionnement du dataset des feux avec le dataset du sol

Pour cette opération, nous constatons que les valeurs de longitude et de latitude dans les deux datasets ne sont pas alignées sur la même grille spatiale. Par conséquent, une jointure classique reposant sur une comparaison directe de ces coordonnées ne peut pas fonctionner.

Afin d'associer chaque point de feu avec son point du sol correspondant, nous devons effectuer une jointure spatiale reposant sur la recherche du point le plus proche (`nearest neighbor`). Une solution efficace consiste à utiliser la structure de données `KDTree`, un arbre multidimensionnel qui permet d'effectuer des recherches de proximité très rapides.

L'arbre est construit à partir des positions géographiques des points du dataset du sol. Ensuite, pour chaque point du dataset du feu, nous interrogeons le `KDTree` pour identifier le point du sol le plus proche.

À la fin du processus, nous obtenons un dataset fusionné dans lequel chaque observation de feu est enrichie par les attributs du point du sol spatialement le plus proche.

Fusionnement du dataset des feux avec le dataset d'élévation

Cette opération est beaucoup plus simple, car le dataset d'élévation est enregistré sous forme de raster. Ce format offre directement une méthode d'échantillonnage : en fournissant

une position géographique (longitude et latitude), il est possible de récupérer la valeur du pixel du raster auquel ce point appartient.

Ainsi, pour chaque point du dataset des feux, il suffit d'utiliser cette fonction d'échantillonnage pour extraire la valeur d'élévation correspondante. À la fin du processus, chaque point de feu est enrichi par la valeur d'altitude issue du raster.

1.0.7 Sampling

Pour gérer le déséquilibre entre les classes **fire** et **no-fire** et améliorer la représentativité spatiale des données, deux méthodes ont été appliquées.

1. Sous-échantillonnage de la classe majoritaire

- La classe **no-fire** étant largement majoritaire, un sous-échantillonnage aléatoire a été effectué pour obtenir un ratio contrôlé de 1 :3 (**fire:no-fire**).
- Cette approche permet de réduire le risque de biais des modèles tout en conservant une dominance réaliste de la classe majoritaire.

2. Binning spatial

- Les points proches dans la grille ont été regroupés pour éviter la redondance spatiale et uniformiser l'échantillonnage.
- Chaque cellule de 0.01° (1,1 km) a conservé un seul point représentatif, garantissant une couverture spatiale homogène.

Impact sur le dataset final

- Déséquilibre initial de 1 :99 (fire :no-fire) réduit à 1 :3.
- Réduction de la sur-représentation de la classe **no-fire**.
- Optimisation du nombre de points et du temps de calcul.
- Validation visuelle confirmant une distribution spatiale équilibrée des points **fire** et **no-fire**.

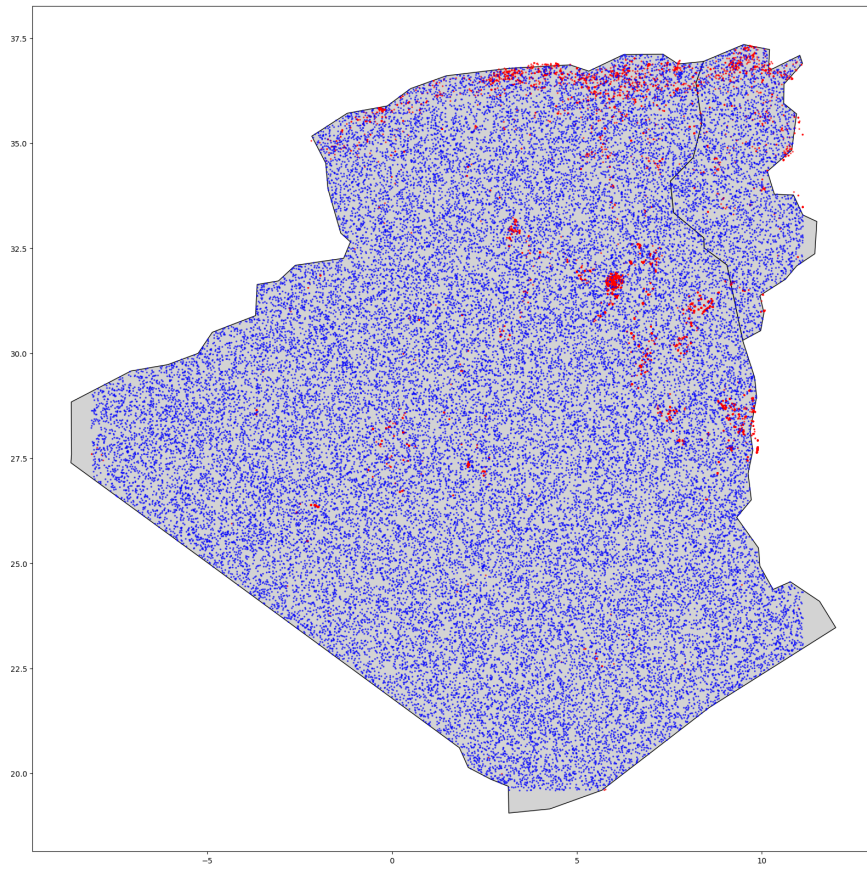


FIGURE 1.20 – dataset fire apres sampling.

1.0.8 Definition Communes

Dans cette section, nous définissons des concepts et des termes communs utilisés dans plusieurs algorithmes d'apprentissage automatique présentés dans ce document.

Distance Euclidienne

La distance euclidienne est une mesure de la distance entre deux points dans un espace euclidien. Elle est définie par la formule suivante pour deux points avec n dimensions $p = (p_1, p_2, \dots, p_n)$ et $q = (q_1, q_2, \dots, q_n)$:

$$d(p, q) = \sqrt{\sum_{i=1}^n (p_i - q_i)^2}$$

Distance de Manhattan

La distance de Manhattan, également connue sous le nom de distance de taxicab ou distance L1, est similairement définie par la formule suivante pour deux points avec n dimensions $p = (p_1, p_2, \dots, p_n)$ et $q = (q_1, q_2, \dots, q_n)$:

$$d(p, q) = \sum_{i=1}^n |p_i - q_i|$$

Distance de Minkowski

La distance de Minkowski est une généralisation des distances euclidienne et de Manhattan. Elle est définie par la formule suivante pour deux points avec n dimensions $p = (p_1, p_2, \dots, p_n)$ et $q = (q_1, q_2, \dots, q_n)$, et un paramètre $r \geq 1$:

$$d(p, q) = \left(\sum_{i=1}^n |p_i - q_i|^r \right)^{\frac{1}{r}}$$

Standardisation

La standardisation, également appelée normalisation Z-score, est une technique de mise à l'échelle des données qui transforme les valeurs d'une variable pour qu'elles aient une moyenne nulle et un écart-type unitaire. La formule de la standardisation pour une valeur x est la suivante :

$$z = \frac{x - \mu}{\sigma}$$

où μ est la moyenne de la variable et σ est son écart-type.

Dummy Encoding

Le Dummy Encoding est une technique d'encodage des variables catégorielles qui transforme chaque catégorie en une variable binaire distincte (0 ou 1). Pour une variable catégorielle avec k catégories, le Dummy Encoding crée k nouvelles variables binaires, où chaque variable indique la présence ou l'absence d'une catégorie spécifique.

Ordinal Encoding

L'Ordinal Encoding est une technique d'encodage des variables catégorielles qui attribue un entier unique à chaque catégorie en fonction de son ordre ou de sa hiérarchie. Contrairement au Dummy Encoding, l'Ordinal Encoding préserve l'ordre des catégories, ce qui peut être utile lorsque les catégories ont une relation intrinsèque (par exemple, "petit", "moyen", "grand" peuvent être encodés comme 0, 1, 2 respectivement), ou quand on souhaite minimiser le nombre de nouvelles variables créées.

F1 Macro

La F1 Macro est une métrique d'évaluation utilisée pour mesurer la performance d'un modèle de classification, en particulier dans les cas de classes déséquilibrées. Elle est calculée en prenant la moyenne non pondérée des scores F1 pour chaque classe. Le score F1 pour une classe est la moyenne harmonique de la précision et du rappel, définie comme suit :

$$F1 = 2 \times \frac{\text{Précision} \times \text{Rappel}}{\text{Précision} + \text{Rappel}}$$

et la F1 Macro est donnée par :

$$F1 \text{ Macro} = \frac{1}{K} \sum_{k=1}^K F1_k$$

ROC AUC

La ROC AUC (Receiver Operating Characteristic - Area Under the Curve) est une métrique d'évaluation utilisée pour mesurer la performance d'un modèle de classification binaire. La courbe ROC trace le taux de vrais positifs (TPR) contre le taux de faux positifs (FPR) à différents seuils de classification. L'aire sous la courbe (AUC) quantifie la capacité du modèle à distinguer entre les classes positives et négatives. Une AUC de 1 indique une classification parfaite, tandis qu'une AUC de 0,5 indique une performance équivalente à un classement aléatoire.

L'entropie

L'entropie est une mesure de l'incertitude ou de la pureté d'un ensemble de données dans le contexte des arbres de décision. Elle est définie par la formule suivante pour un ensemble de données avec c classes :

$$H(D) = - \sum_{i=1}^c p_i \log_2(p_i)$$

ou p_i est la proportion d'instances appartenant à la classe i dans l'ensemble de données D . Une entropie élevée indique une plus grande incertitude, tandis qu'une entropie faible indique une plus grande pureté.

Indice de Gini

L'indice de Gini est une mesure de l'impureté ou de la pureté d'un ensemble de données, souvent utilisée dans les arbres de décision. Il est défini par la formule suivante pour un

ensemble de données avec c classes :

$$G(D) = 1 - \sum_{i=1}^c p_i^2$$

ou p_i est la proportion d'instances appartenant à la classe i dans l'ensemble de données D . Un indice de Gini de 0 indique une pureté totale (toutes les instances appartiennent à une seule classe), tandis qu'un indice de Gini proche de 0,5 indique une impureté maximale (les instances sont réparties uniformément entre les classes).

1.0.9 Knn

Definition

KNN est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. Il fonctionne en trouvant les ' k ' points de données les plus proches dans l'espace des caractéristiques et en utilisant ces points pour prédire la classe ou la valeur de la nouvelle donnée.

Les distances couramment utilisées pour mesurer la proximité entre les points de données incluent la distance euclidienne, la distance de Manhattan et la distance de Minkowski.

Les paramètres clés de l'algorithme KNN incluent :

- **K** : Le nombre de voisins à considérer pour la prédiction.
- **Distance Metric** : La méthode utilisée pour calculer la distance entre les points de données.
- **Weighting** : La manière dont les voisins sont pondérés lors de la prise de décision (par exemple, pondération uniforme ou pondération par distance).

Implemenation from scratch

Voici notre implémentation de l'algorithme KNN en Python avec ces petites modifications :

- Utilisation de la distance carrée $(x-y)^2$ à la place de la distance euclidienne $\sqrt{(x-y)^2}$ pour éviter le calcul de la racine carrée, aussi pour bénéficier du calcul matricielle plus rapide avec numpy.
- Cette modification n'affecte pas le résultat final, car la racine carrée est une fonction monotone croissante, donc l'ordre des distances reste le même. autrement dit, si $d_1 < d_2$, alors $\sqrt{d_1} < \sqrt{d_2}$.

Implemenation KNN from scratch

1.0.10 Normalization et Encodage de données

Comme KNN est basé sur la distance entre les points de données, il est crucial de normaliser les caractéristiques pour garantir que toutes les dimensions contribuent équitablement au calcul de la distance. Les techniques que nous avons utilisées sont :

- Pour la normalisation nous avons utilisé **Standarisation (Z-score)** pour centrer et réduire les données et préserver les distances relatives entre les points.
- Pour l'encodage des caractéristiques categorilaes, nous avons utilisé **Dummy Encoding** pour ne pas introduire d'ordre entre les catégories, sauf pour les caractéristiques avec un grand nombre de catégories uniques comme le **GRIDCODE** où nous avons

utilisé **Ordinal Encoding** pour ne pas introduire beaucoup de nouvelle caractéristiques.

1.0.11 Balancement de données

Comme **KNN** est sensible à la distribution des classes dans les données, nous avons exploré différentes techniques de balancement pour équilibrer les classes, nous avons utilisé un model avec $k = 5$ et une distance pondérée et le pour la comparaison des resultats nous avons utilisé le métrique **ROC AUC**. Les techniques de balancement que nous avons testées sont :

- **Original Data** : Utilisation des données telles quelles, sans modification. Les resultats obtenus sont comme suit :

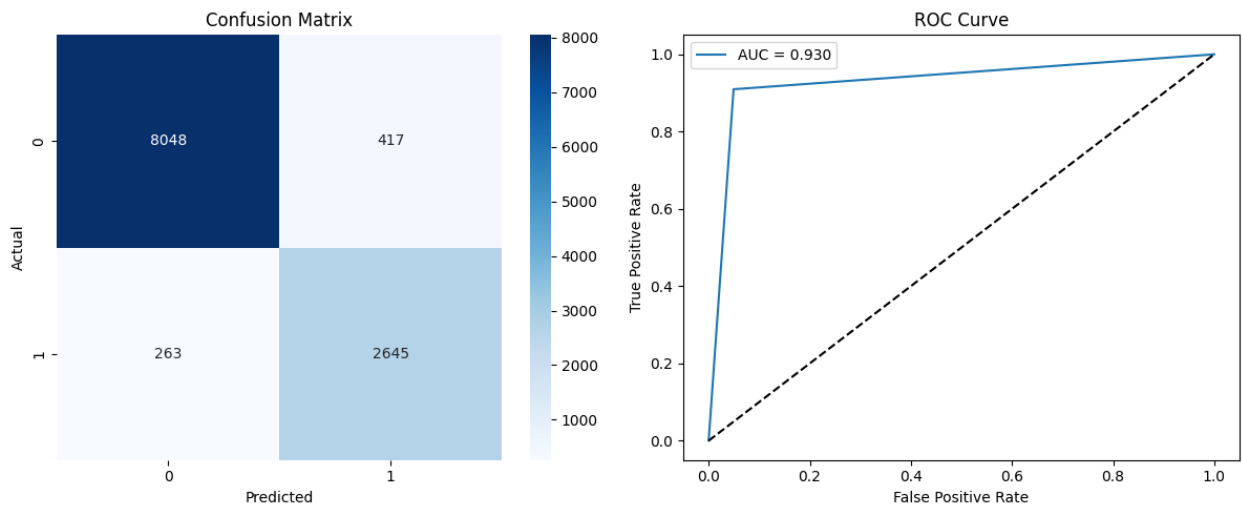


FIGURE 1.21 – KNN avec données originales

- **Random Under Sampling** : Sous échantillonnage aléatoire des données majoritaires pour équilibrer les classes. Les resultats obtenus sont comme suit :

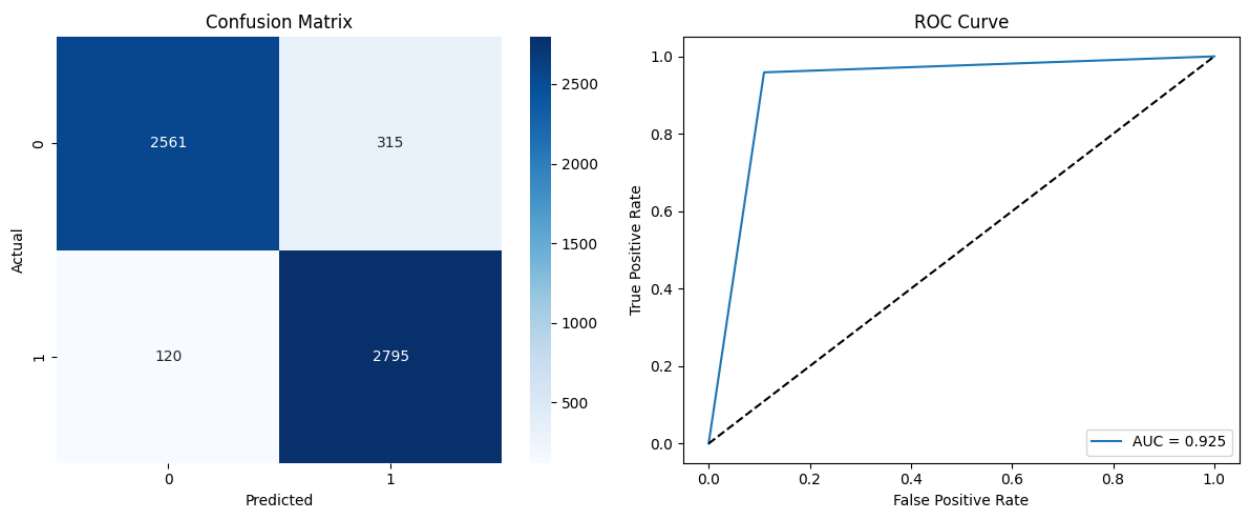


FIGURE 1.22 – KNN avec données random under sampling

- **Ranodom Over Sampling** : Sur échantillonnage aléatoire des données minoritaires pour équilibrer les classes. Les resultats obtenus sont comme suit :

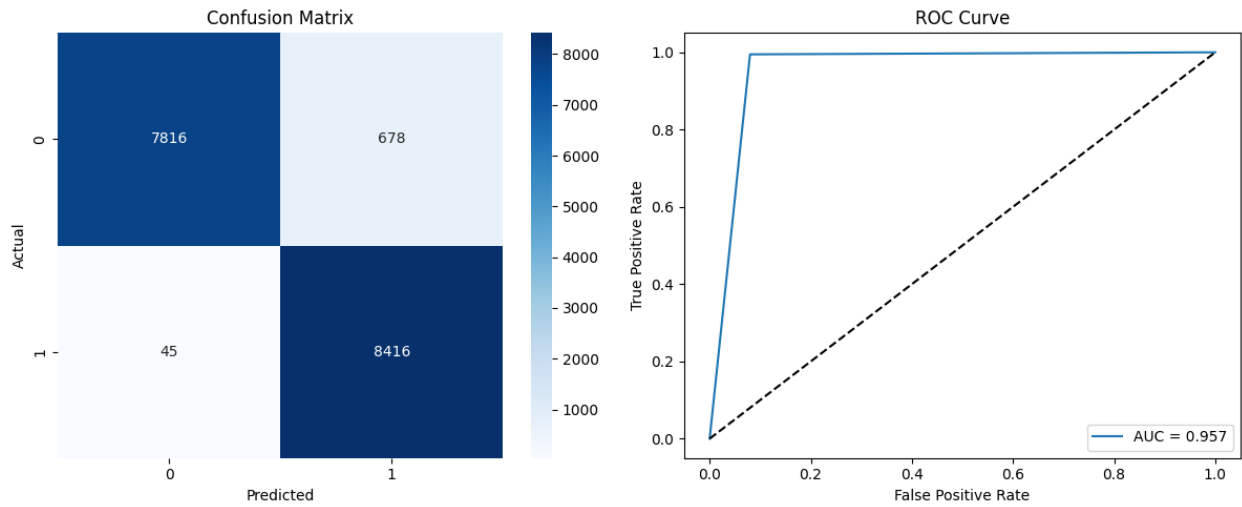


FIGURE 1.23 – KNN avec données random over sampling

- **Smote Over Sampling** : Sur échantillonnage synthétique des données minoritaires pour équilibrer les classes. Les resultats obtenus sont comme suit :

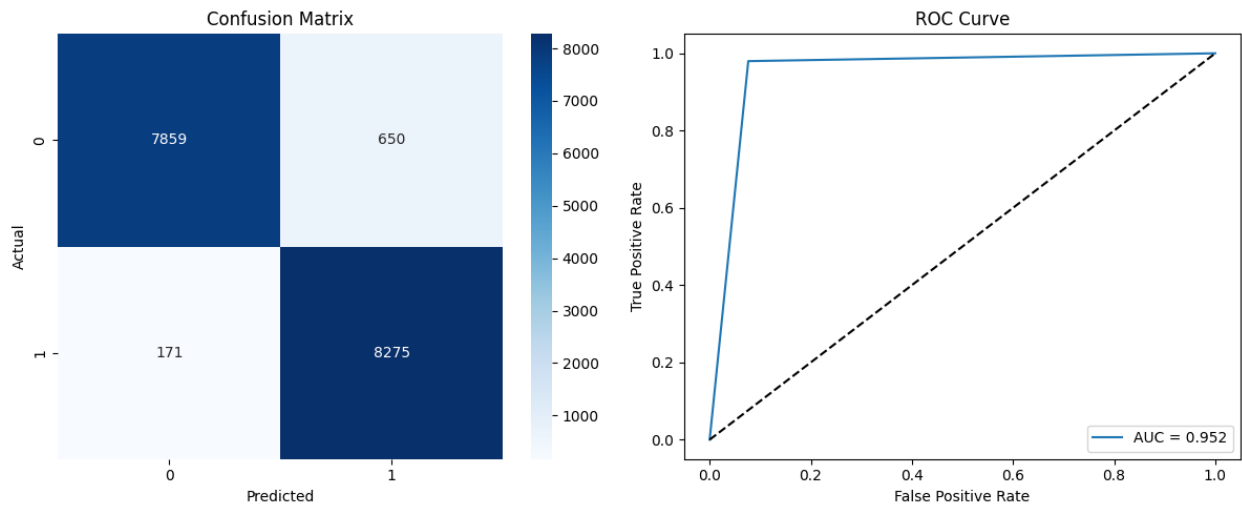


FIGURE 1.24 – KNN avec données smote over sampling

- **Tomek Under Sampling** : Sous échantillonnage des données majoritaires en éliminant les exemples proches des frontières de décision. Les resultats obtenus sont comme suit :

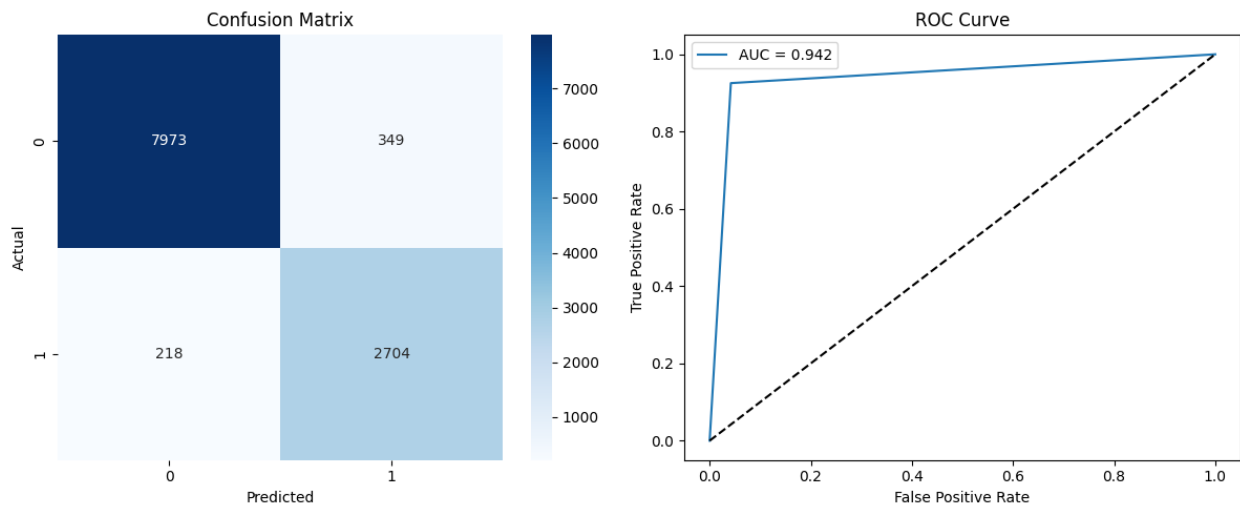


FIGURE 1.25 – KNN avec données tomes under sampling

À la fin la meilleure technique de balancement pour KNN est **Random Over Sampling** avec les resultats suivants :

- Best sampling method : Random Over-Sampling with ROC AUC : 0.9574
- New dataset sizes : Train=67819, Test=16955, Full=84774
- Class distribution : class 0 : 33893, class 1 : 33926

1.0.12 Réglage des paramètres

Pour le réglage des paramètres de l'algorithme KNN, nous avons utilisé la validation croisée (k-folds avec $k = 3$) avec une recherche d'optimisation bayésienne de 30 itérations, la plage des paramètres testés est la suivante :

- **n_neighbors** : [1, 10]
- **weights** : ['uniform', 'distance'] indiquant si tous les voisins ont le même poids ou si les voisins plus proches ont un poids plus élevé.
- **metric** : ['euclidean', 'manhattan'] différentes méthodes pour calculer la distance entre les points de données.

Les meilleurs paramètres trouvés sont sont illustrés dans Table 1.5.

1.0.13 Réduction de dimensionnalité

Comme KNN peut être affecté par la malédiction de la dimensionnalité, nous avons appliqué une réduction de dimensionnalité en utilisant l'analyse en composantes principales (PCA) pour réduire le nombre de caractéristiques tout en conservant 90% de la variance des données. Après l'application de PCA, le nombre de caractéristiques est passé de 40 à 16. Nous avons ensuite répété le processus de réglage des paramètres avec les mêmes plages de valeurs. Les meilleurs paramètres trouvés après la réduction de dimensionnalité sont exactement les mêmes que ceux trouvés avant la réduction de dimensionnalité, les résultat sont illustrés dans Table 1.5

1.0.14 Comparaison de resultats

Les resultats finaux obtenus avec KNN après le réglage des paramètres et l'utilisation de la technique de balancement Random Over Sampling sont les suivants :

Réglage de paramètres

Configuration	n_neighbors	weights	metric
Sans réduction de dimensionnalité	1	uniform	Manhattan
Avec réduction de dimensionnalité	1	uniform	Manhattan

TABLE 1.5 – Comparaison des performances avec et sans réduction de dimensionnalité

Résultats finaux

Configuration	Accuracy	Precision	Recall	F1 Score
Sans réduction de dimensionnalité	0.9706	0.9528	0.9902	0.9711
Avec réduction de dimensionnalité	0.9694	0.9509	0.9898	0.9700

TABLE 1.6 – Comparaison des performances avec et sans réduction de dimensionnalité

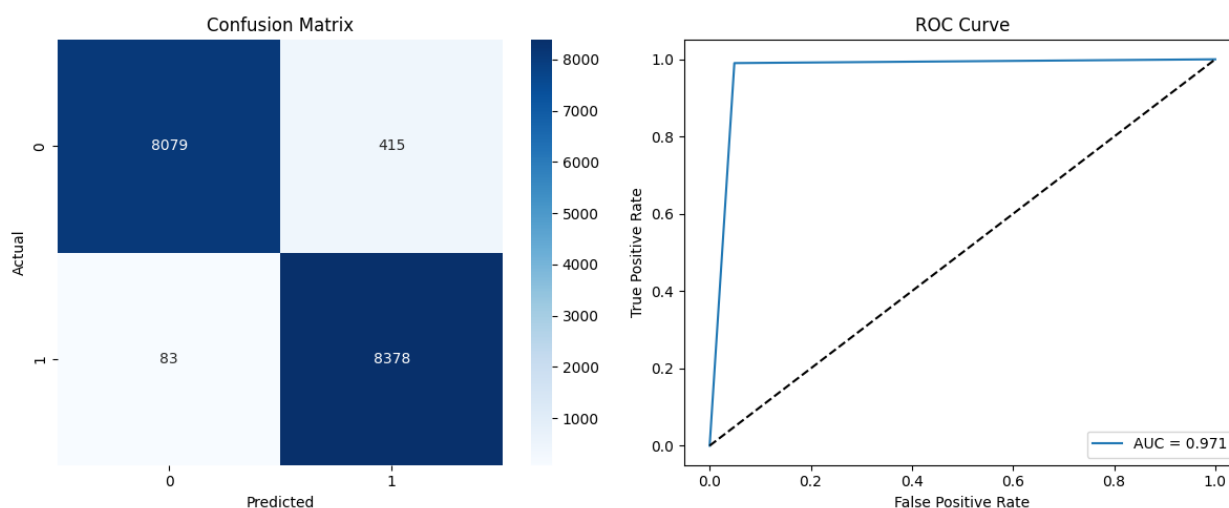


FIGURE 1.26 – KNN - Final Results with no PCA

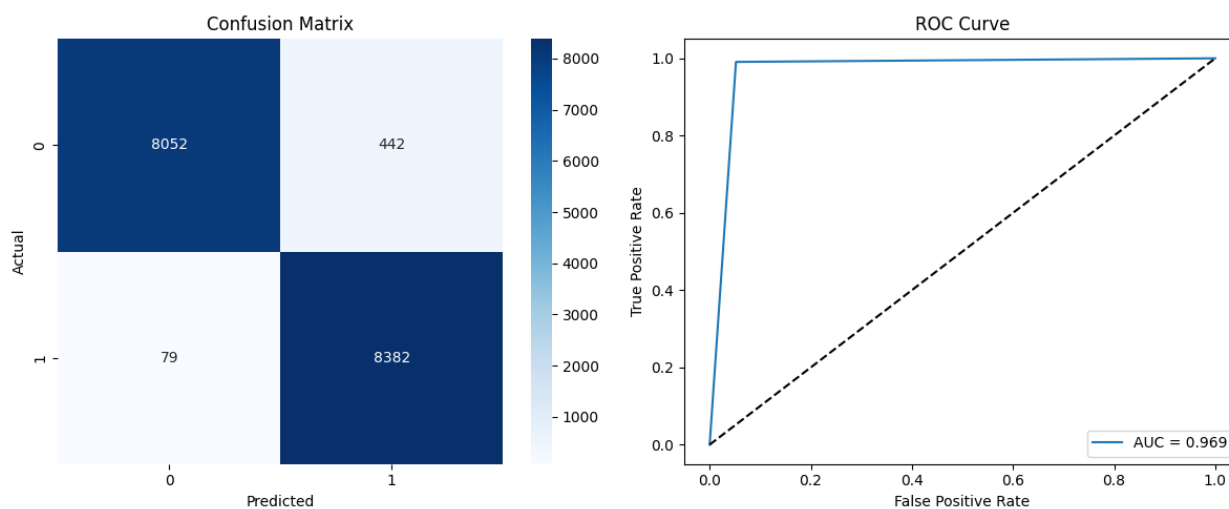


FIGURE 1.27 – KNN - Final Results with PCA

Nous observons que la réduction de dimensionnalité n'a pas eu un impact significatif sur les performances de l'algorithme KNN dans ce cas, mais elle a permis de réduire significativement le temps de calcul et les ressources nécessaires pour l'entraînement et la prédiction du modèle.

1.0.15 Decision Tree

Definition

Decision Tree est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. Il fonctionne en divisant récursivement l'espace des caractéristiques en sous-ensembles basés sur des conditions de seuil sur les caractéristiques, formant ainsi une structure arborescente où chaque nœud interne représente une condition de décision et chaque feuille représente une prédiction de classe ou de valeur.

Les critères couramment utilisés pour mesurer la qualité des divisions incluent l'indice de Gini, le gain d'information (entropie) et la réduction de la variance.

Les paramètres clés de l'algorithme Decision Tree incluent :

- **Max Depth** : La profondeur maximale de l'arbre, généralement utilisé pour contrôler la complexité de l'arbre et éviter le surapprentissage.
- **Min Samples Split** : Le nombre minimum d'échantillons requis pour diviser un nœud, généralement fixé à la valeur minimale de 2.
- **Min Samples Leaf** : Le nombre minimum d'échantillons requis pour être à une feuille, généralement fixé à la valeur minimale de 1.
- **Criterion** : La fonction utilisée pour mesurer la qualité des divisions (par exemple, Gini, Entropie).

Implemenation from scratch

Voici notre implémentation de l'algorithme Decision Tree en Python.

Implemenation Decision Tree from scratch

1.0.16 Normalization et Encodage de données

Decision Tree n'est pas basé sur la distance entre les points de données, donc la normalisation des caractéristiques n'est pas nécessaire dans ce cas. Cependant, l'encodage des caractéristiques catégorielles est important pour permettre à l'algorithme de traiter correctement ces données. Les techniques que nous avons utilisées sont :

- Pour l'encodage des caractéristiques categorilaes, nous avons utilisé **Ordinal Encoding** pour ne pas introduire beaucoup de nouvelle caractéristiques ainsi que pour préserver un order entre les valeurs pour permettre à l'algorithme de mieux séparer les données.

1.0.17 Balancement de données

L'algorithme de Decision Tree est sensible à la distribution des classes dans les données comme KNN, nous avons donc fait la meme exploration de différentes techniques de balancement pour équilibrer les classes, nous avons utilisé un model avec **criterion='gini'** et le reste des paramètres pas defaut, et le pour la comparaison des resultats nous avons utilisé le métrique **ROC AUC**. Les techniques de balancement que nous avons testées sont :

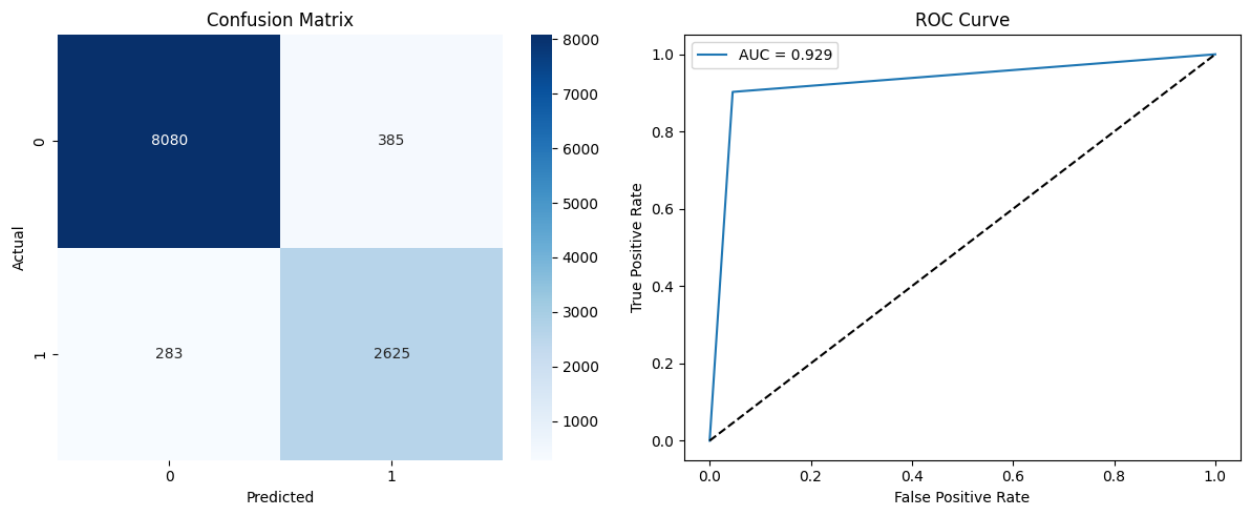


FIGURE 1.28 – Decision Tree avec données originales

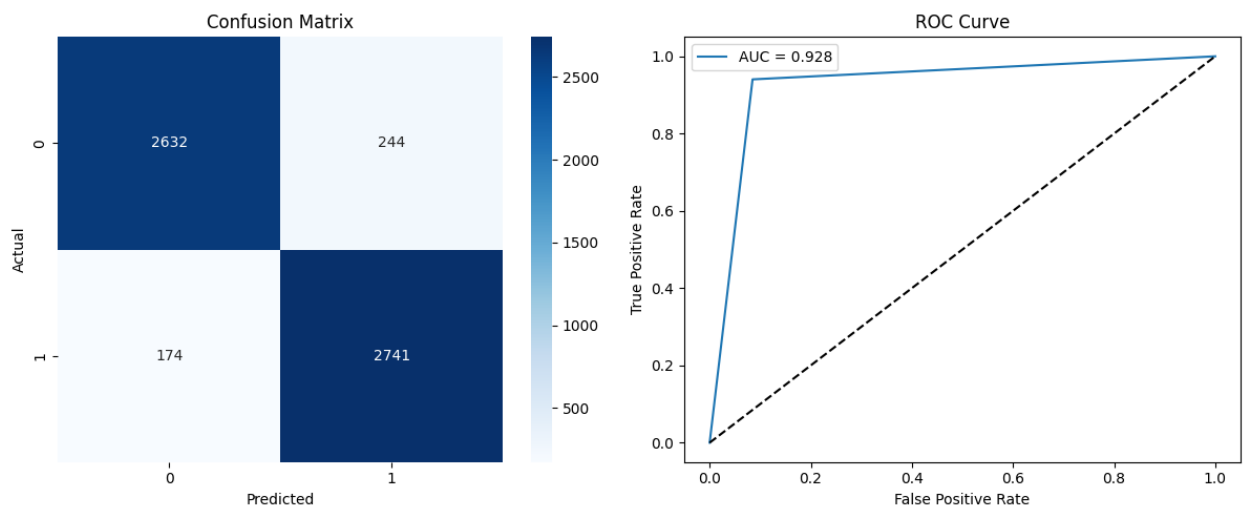


FIGURE 1.29 – Decision Tree avec données random under sampling

- **Original Data :**
- **Random Under Sampling :**
- **Ranodom Over Sampling :**
- **Smote Over Sampling :**
- **Tomek Under Sampling :**

À la fin la meilleure technique de balancement pour Decision Tree est **Random Over Sampling** avec les resultats suivants :

- Best sampling method : Random Over-Sampling with ROC AUC : 0.9725
- New dataset sizes : Train=67819, Test=16955, Full=84774
- Class distribution : class 0 : 33893, class 1 : 33926

1.0.18 Réglage des paramètres

Pour le réglage des paramètres de l'algorithme Decision Tree, nous avons utilisé la validation croisée (k-folds avec $k = 5$) avec une recherche d'optimisation bayésienne de 50 itérations, la plage des paramètres testés est la suivante :

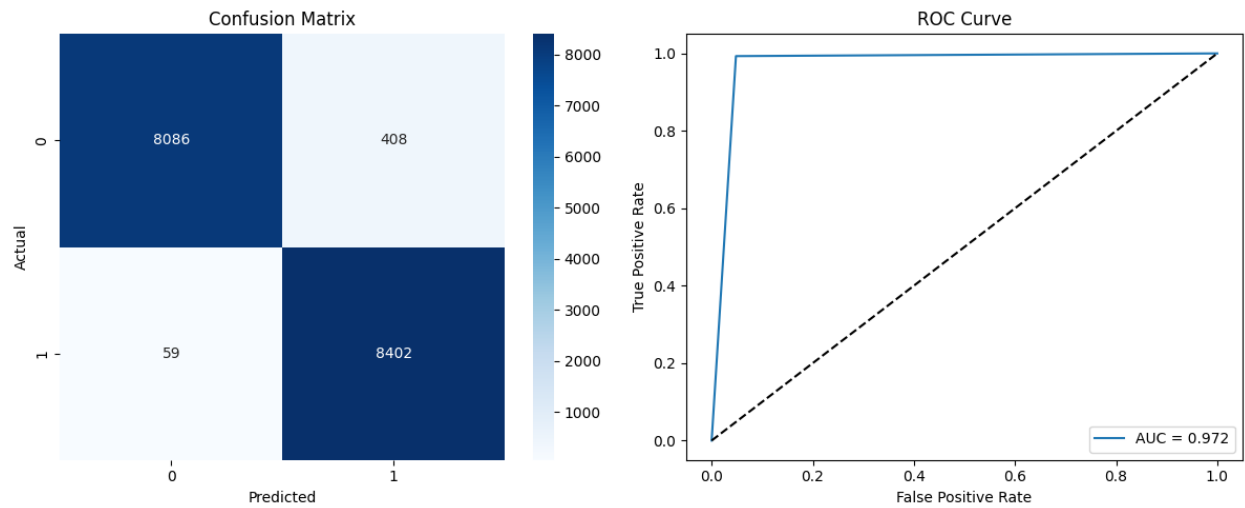


FIGURE 1.30 – Decision Tree avec données random over sampling

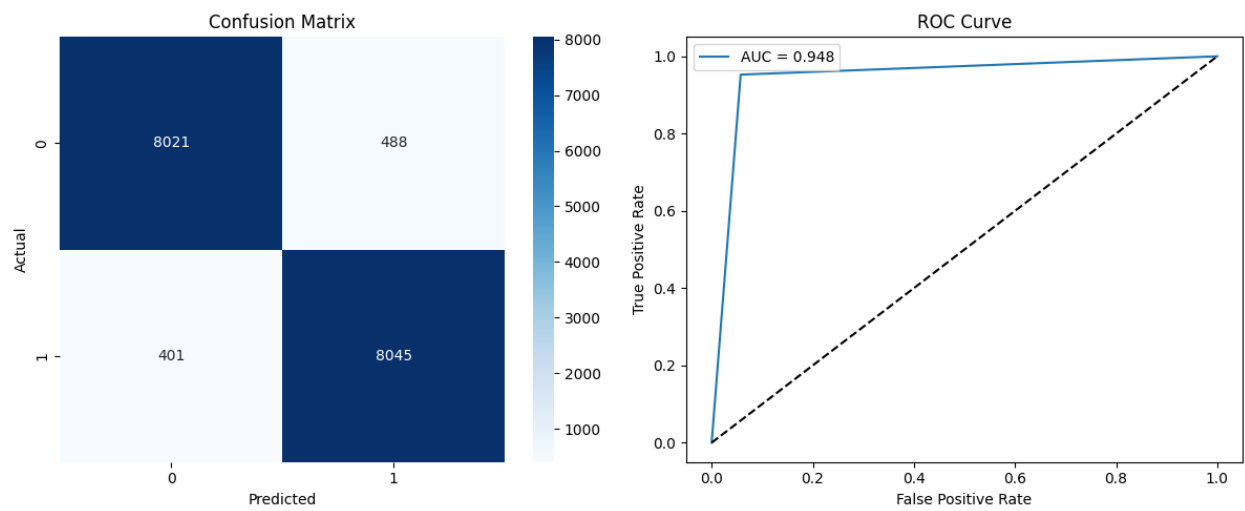


FIGURE 1.31 – Decision Tree avec données smote over sampling

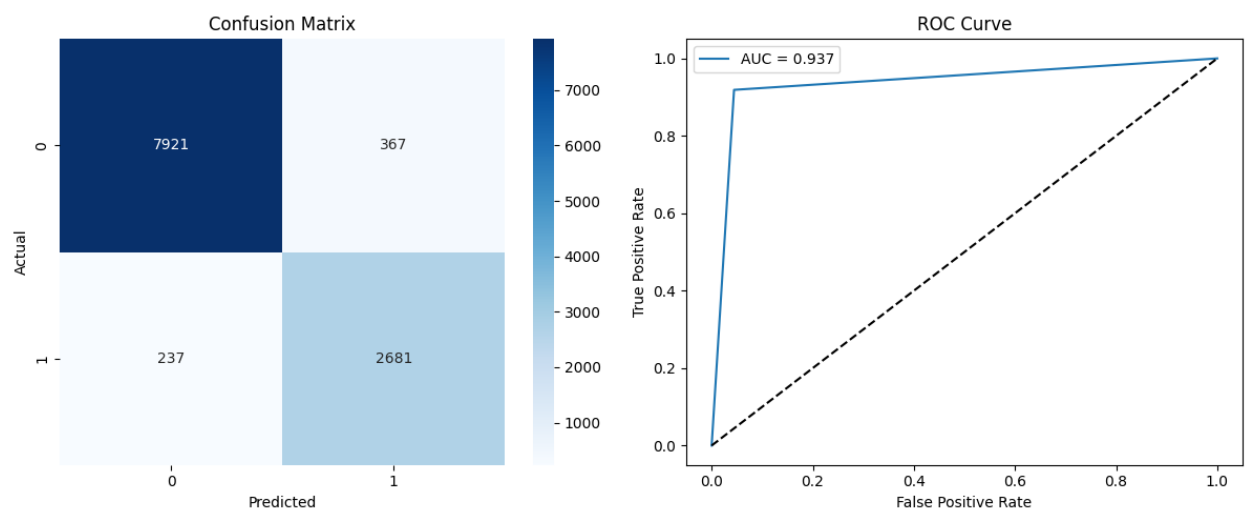


FIGURE 1.32 – Decision Tree avec données tokek under sampling

-
- **max_depth** : [1, 50] la profondeur maximale de l'arbre
 - **min_samples_split** : [2, 50] le nombre minimum d'échantillons requis pour diviser un nœud interne
 - **min_samples_leaf** : [1, 50] le nombre minimum d'échantillons requis pour être à une feuille
 - **max_features** : [None, "sqrt", "log2"] le nombre de caractéristiques à considérer lors de la recherche de la meilleure division, None signifie que toutes les caractéristiques sont utilisées
 - **criterion** : ["gini", "entropy"] la fonction pour mesurer la qualité d'une division

Les meilleurs paramètres trouvés sont illustrés dans Table ?? . On remarque que tous les paramètres sont à leurs valeurs minimales sauf max depth qui est limité à 39 pour éviter le surapprentissage.

1.0.19 Réduction de dimensionnalité

Decision Tree peut être affecté par la malédiction de la dimensionnalité comm KNN, nous avons donc appliqué une réduction de dimensionnalité en utilisant l'analyse en composantes principales (PCA) pour réduire le nombre de caractéristiques tout en conservant 90% de la variance des données. Après l'application de PCA, le nombre de caractéristiques est passé de 38 à 1. Nous avons ensuite répété le processus de réglage des paramètres avec les mêmes plages de valeurs. Les meilleurs paramètres trouvés après la réduction de dimensionnalité sont exactement les mêmes que ceux trouvés avant la réduction de dimensionnalité et sont illustrés dans Table ?? .

1.0.20 Comparaison de resultats

Les resultats finaux obtenus avec KNN après le réglage des paramètres et l'utilisation de la technique de balancement Random Over Sampling sont les suivants :

Configuration	Criterion	Max Depth	Min Samples Split	Min Sample
Sans réduction de dimensionnalité	gini	39	2	1
Avec réduction de dimensionnalité	gini	50	2	1

TABLE 1.7 – Comparaison des performances avec et sans réduction de dimensionnalité

Configuration	Accuracy	Precision	Recall	F1 Score
Sans réduction de dimensionnalité	0.9706	0.9528	0.9902	0.9711
Avec réduction de dimensionnalité	0.9694	0.9509	0.9898	0.9700

TABLE 1.8 – Comparaison des performances avec et sans réduction de dimensionnalité

Nous observons que la réduction de dimensionnalité n'a pas eu un impact significatif sur les performances de l'algorithme Decision Tree dans ce cas, mais elle a permis de réduire significativement le temps de calcul et les ressources nécessaires pour l'entraînement et la prédiction du modèle en réduisant tous les caractéristiques du dataset en une seule composante.

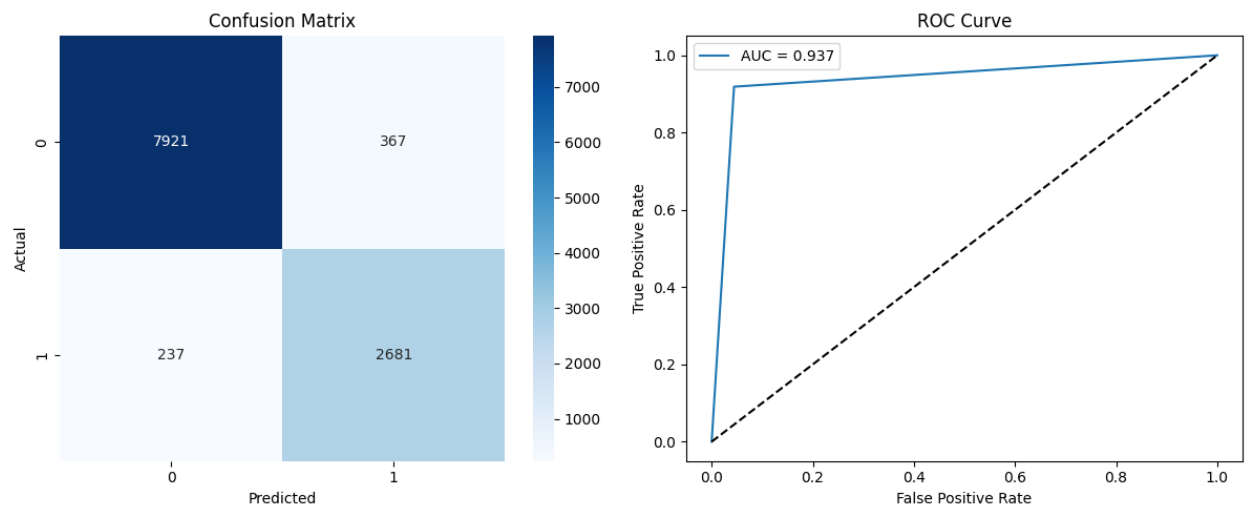


FIGURE 1.33 – Decision Tree - Final Results with no PCA

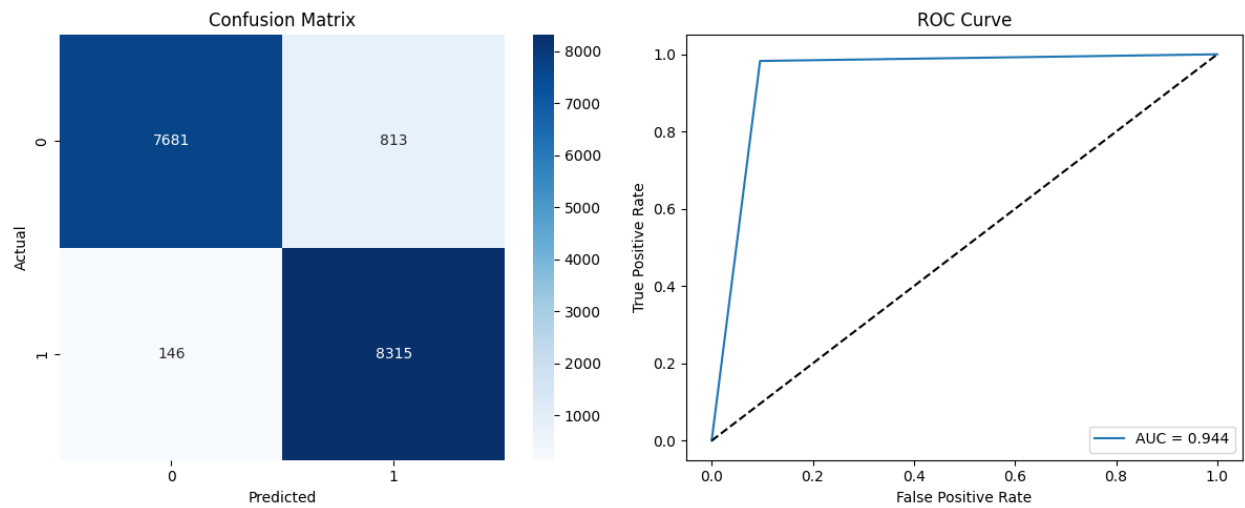


FIGURE 1.34 – Decision Tree - Final Results with PCA

1.0.21 Random Forest

Definition

Random Forest est un algorithme d'apprentissage supervisé utilisé pour la classification et la régression. Il fonctionne en combinant plusieurs arbres de décision construits sur des sous-échantillons aléatoires des données, et en agrégeant leurs prédictions pour améliorer la précision et réduire le surapprentissage. Chaque arbre de décision est construit en sélectionnant aléatoirement un sous-ensemble des caractéristiques à chaque nœud, ce qui introduit de la diversité parmi les arbres et améliore la robustesse du modèle global.

Les critères utilisés pour mesurer la qualité des divisions incluent sans les meme que celle d'arbre de decision comme Gini, le gain d'information (entropie) et la réduction de la variance.

Les paramètres clés de l'algorithme Random Forest incluent :

- tous les paramètres de l'arbre de décision, tels que :
 - **Max Depth** : La profondeur maximale de l'arbre, généralement utilisé pour contrôler la complexité de l'arbre et éviter le surapprentissage.
 - **Min Samples Split** : Le nombre minimum d'échantillons requis pour diviser un nœud, généralement fixé à la valeur minimale de 2.
 - **Min Samples Leaf** : Le nombre minimum d'échantillons requis pour être à une feuille, généralement fixé à la valeur minimale de 1.
 - **Criterion** : La fonction utilisée pour mesurer la qualité des divisions (par exemple, Gini, Entropie).
- **N Estimators** : Le nombre d'arbres dans la forêt, varie selon le dataset.

Implemenation from scratch

Voici notre implémentation de l'algorithme Random Forest en Python.

Implemenation Random Forest from scratch

1.0.22 Normalization et Encodage de données

Puisque Random Forest est basé sur des arbres de décision, nous avons appliqué les mêmes techniques de normalisation et d'encodage que pour l'algorithme d'arbre de décision :

- Pour l'encodage des caractéristiques categoriales, nous avons utilisé **Ordinal Encoding** pour ne pas introduire beaucoup de nouvelle caractéristiques ainsi que pour préserver un order entre les valeurs pour permettre à l'algorithme de mieux séparer les données.

1.0.23 Balancement de données

Pour le balancement de classes comme avec les arbres de décision, nous avons utilisé un model de Random Forest de base avec les paramètres importants fixés à 50 pour **n_estimators** et "gini" pour **criterion** et nous avons utilisé la métrique **ROC AUC** pour l'évaluation. Les techniques de balancement que nous avons testées sont :

- **Original Data** :

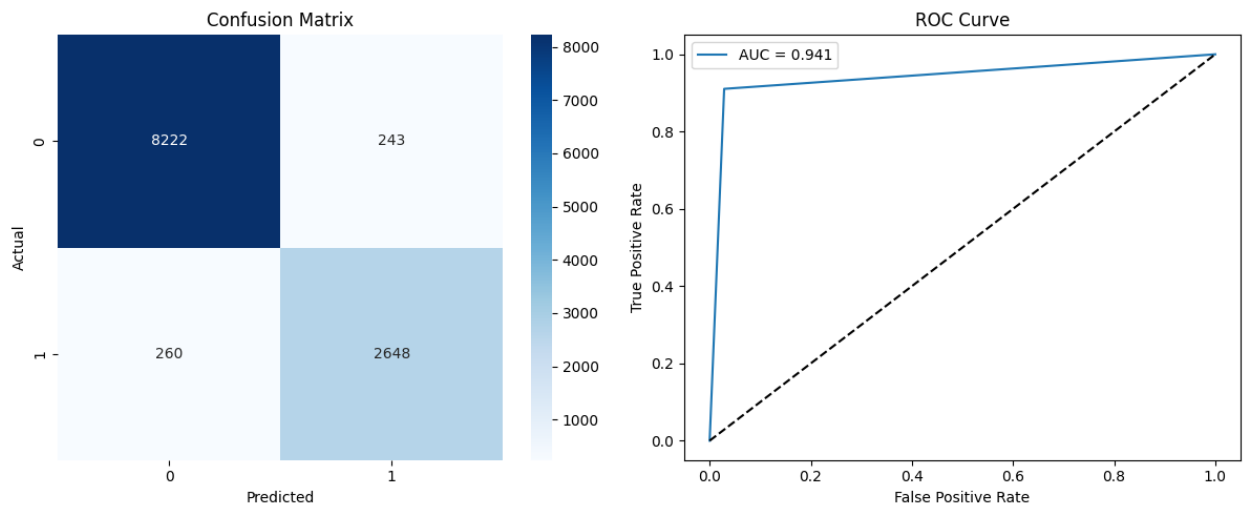


FIGURE 1.35 – Random Forest avec données originales

— Random Under Sampling :

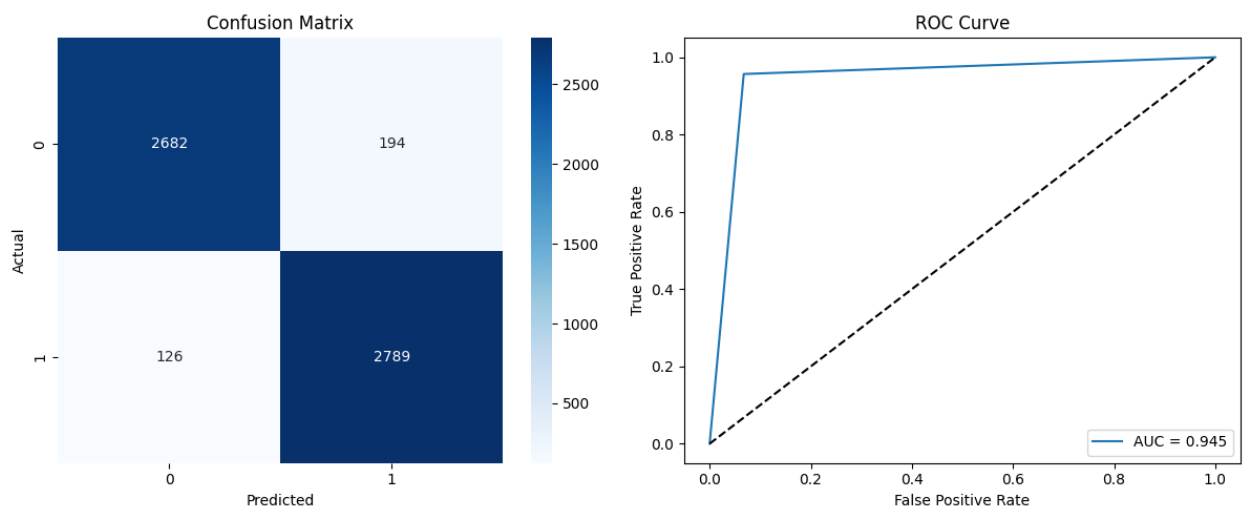


FIGURE 1.36 – Random Forest avec données random under sampling

— Random Over Sampling :

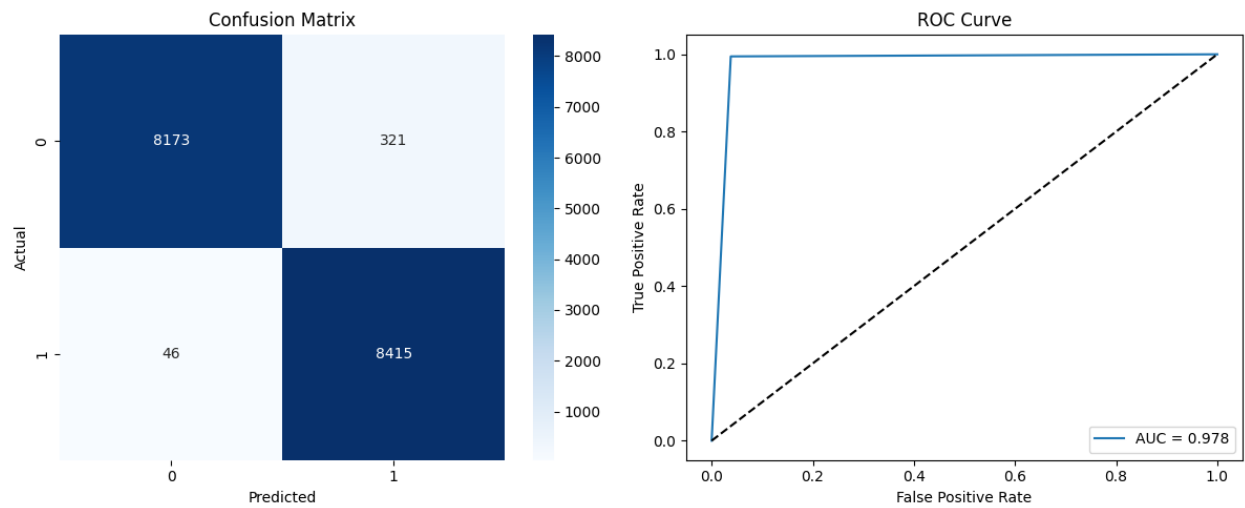


FIGURE 1.37 – Random Forest avec données random over sampling

— **Smote Over Sampling :**

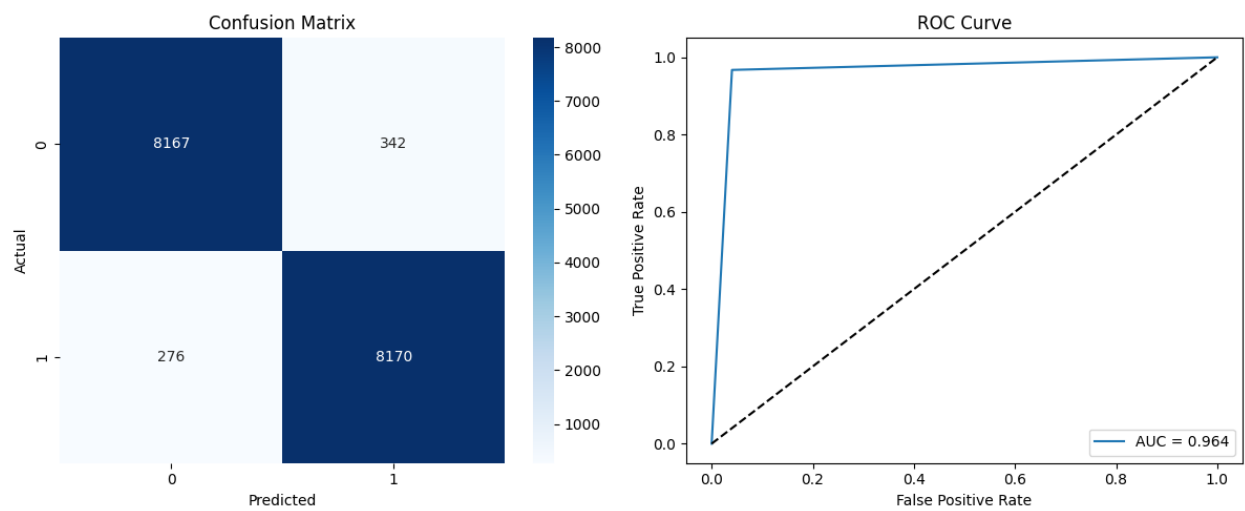


FIGURE 1.38 – Random Forest avec données smote over sampling

— **Tomek Under Sampling :**

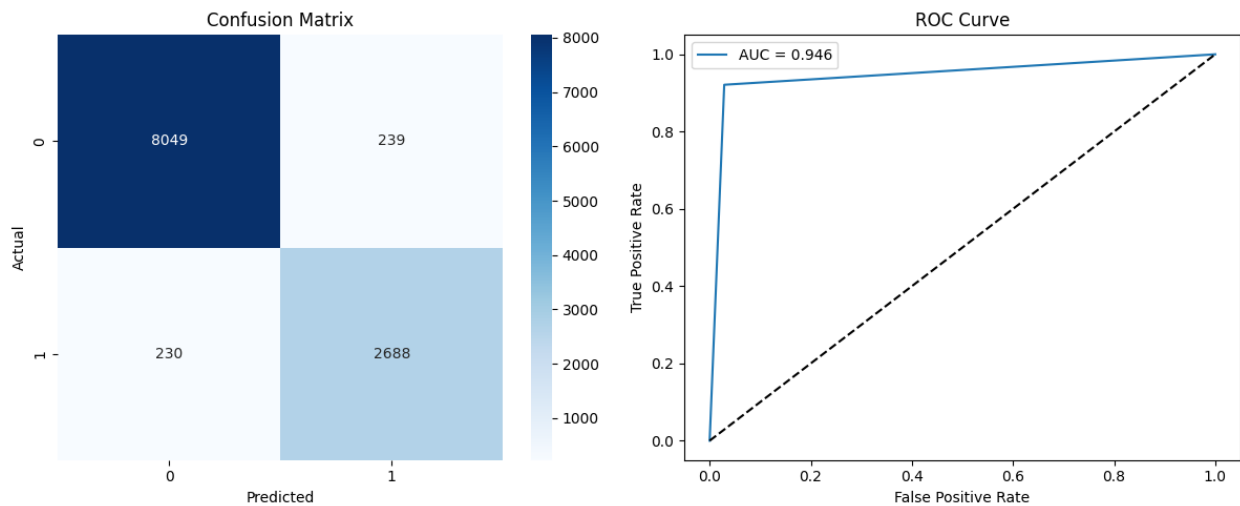


FIGURE 1.39 – Random Forest avec données tomek under sampling

À la fin la meilleure technique de balancement pour Random Forest est **Random Over Sampling** avec les resultats suivants :

- Best sampling method : Random Over-Sampling with ROC AUC : 0.9784
- New dataset sizes : Train=67819, Test=16955, Full=84774
- Class distribution : class 0 : 33893, class 1 : 33926

1.0.24 Réglage des paramètres

Pour le réglage des paramètres de l'algorithme Random Forest, nous avons utilisé la validation croisée (k-folds avec $k = 2$) avec une recherche de grille, la plage des paramètres testés est la suivante :

- **max_depth**, **min_samples_split**, **min_samples_leaf** : les memes valeurs optimales que pour l'arbre de décision
- **n_estimators** : [20, 30, 50, 100] le nombre d'arbres dans la forêt
- **max_features** : [None, "sqrt", "log2"] le nombre de caractéristiques à considérer lors de la recherche de la meilleure division pour chaque arbre, None signifie que toutes les caractéristiques sont utilisées
- **criterion** : ["gini", "entropy"] la fonction pour mesurer la qualité d'une division dans chaque arbre

Voici le diagramme d'importance des caractéristiques pour le modèle Random Forest avec les meilleurs paramètres trouvés :

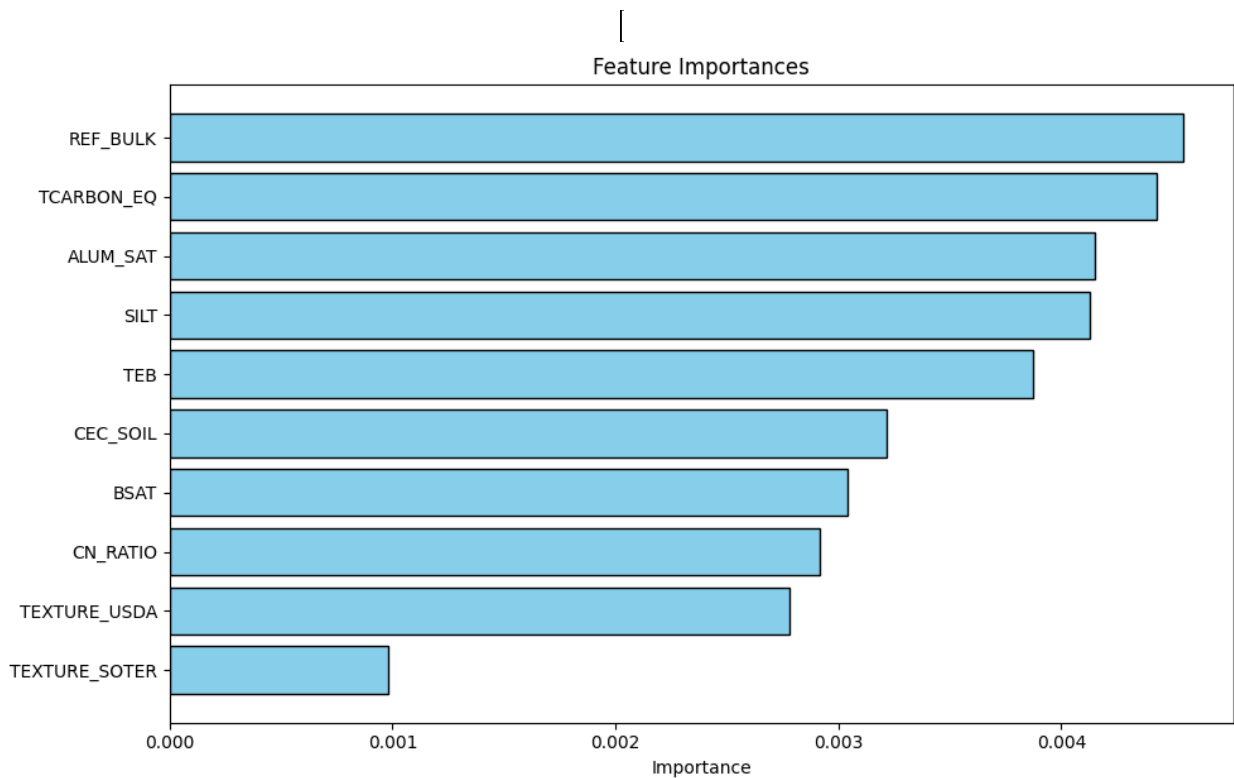


FIGURE 1.40 – Random Forest - Feature Importance

Les meilleurs paramètres trouvés sont sont illustrés dans Table ??.

1.0.25 Réduction de dimensionnalité

Comme toujours, Random Forest peut bénéficier de la réduction de dimensionnalité puisque il est basé sur des arbres de décision. Nous avons utilisé l'analyse en composantes principales (PCA) pour réduire la dimensionnalité des données avec une variance expliquée de 90%. Nous avons eu le meme résultat que celui des arbres de decision (car le nombre de caractéristiques est le meme grace au meme type d'encodage). Après avoir appliqué PCA, le nombre de caractéristiques est passé de 38 à 1. Nous avons ensuite répété le processus de réglage des paramètres avec les mêmes plages de valeurs. Les meilleurs paramètres trouvés après la réduction de dimensionnalité sont également illustrés dans Table ??.

1.0.26 Comparaison de resultats

Les resultats finaux obtenus avec KNN après le réglage des paramètres et l'utilisation de la technique de balancement Random Over Sampling sont les suivants :

Configuration	Criterion	N Estimators	Max Features
Sans réduction de dimensionnalité	gini	100	log2
Avec réduction de dimensionnalité	gini	100	log2

TABLE 1.9 – Comparaison des performances avec et sans réduction de dimensionnalité

Configuration	Accuracy	Precision	Recall	F1 Score
Sans réduction de dimensionnalité	0.9985	0.9969	1	0.9985
Avec réduction de dimensionnalité	0.9434	0.9108	0.9829	0.9455

TABLE 1.10 – Comparaison des performances avec et sans réduction de dimensionnalité

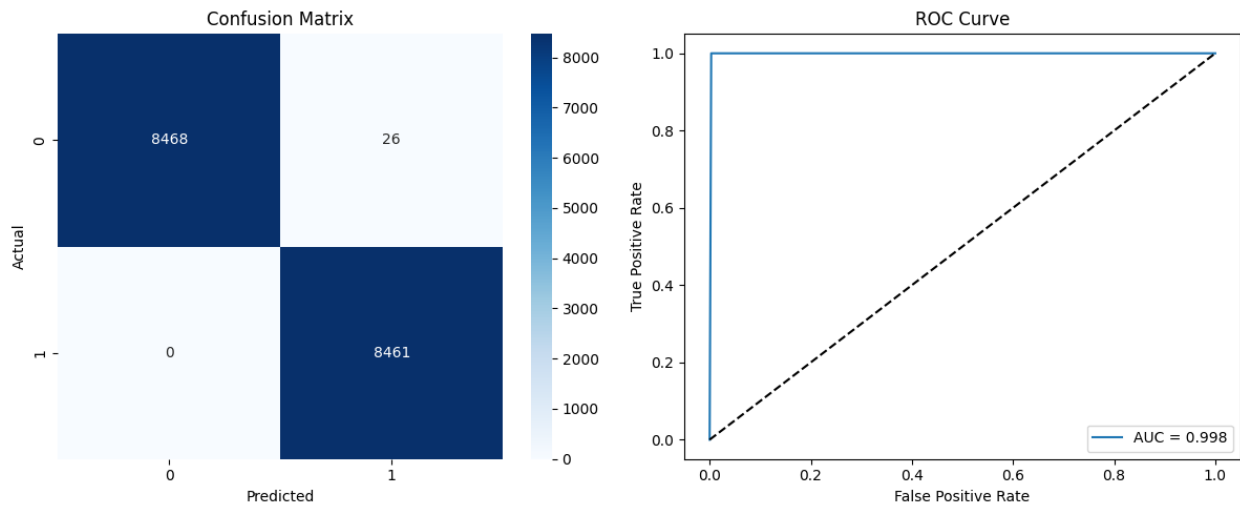


FIGURE 1.41 – Random Forest - Final Results with no PCA

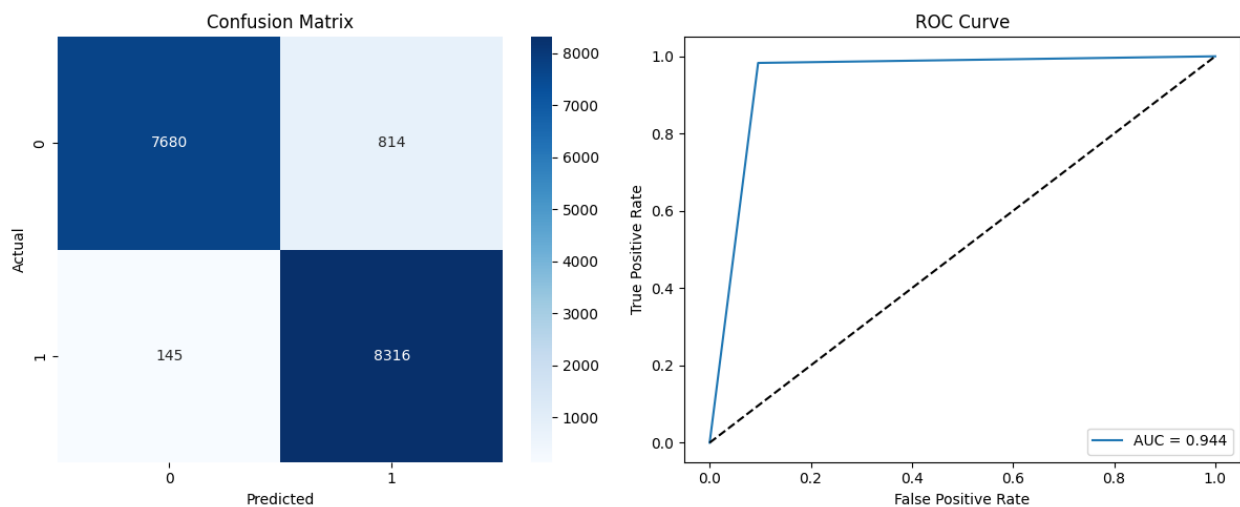


FIGURE 1.42 – Random Forest - Final Results with PCA

On observe que la réduction de dimensionnalité a eu un impact négatif sur les performances de l'algorithme Random Forest dans ce cas, contrairement à l'arbre de décision. Cependant, elle a permis de réduire significativement le temps de calcul et les ressources nécessaires pour l'entraînement et la prédiction du modèle en réduisant tous les caractéristiques du dataset en une seule composante. On remarque aussi que les paramètres optimaux restent les mêmes avant et après la réduction de dimensionnalité.

Chapitre 2

Conclusion

Ce projet a permis de poser les bases d'un système robuste de prédiction des risques d'incendie pour la région Algérie-Tunisie. La première phase cruciale de prétraitement et d'intégration a permis d'harmoniser des données complexes, allant de la texture du sol à 20 cm de profondeur (couche D1 de la base HWSD) jusqu'aux variables climatiques de World-Clim. Le recours à des techniques avancées comme les KDTrees a assuré une fusion spatiale précise malgré des grilles de coordonnées divergentes. L'application future des modèles de Supervised Learning et d'Unsupervised Learning permettra de transformer ces données brutes en une cartographie prédictive opérationnelle. En identifiant les zones de vulnérabilité et les schémas naturels de regroupement des feux, ce travail fournit des perspectives interprétables essentielles pour la conception de systèmes d'alerte précoce et de plans de prévention des feux de forêt. En conclusion, la maîtrise du cycle complet de la donnée, de l'extraction à la modélisation "from scratch", renforce la fiabilité des outils de surveillance environnementale pour ces deux pays