MSPR

« Big Data et Analyse de données »

RNCP N°35584

Hamza Benalia - Eryan Delmon - Aymerick Payet - Nabil Bedjih

Introduction

Nous constatons que la popularité des prototypes d'intelligence artificielle ne cesse de d'évoluer de manière positive. Dans ce contexte et étant nous même des experts IT nous avons été sollicité par une start-up fictive, dirigée par Jean-Edouard de la Motte Rouge. Cette société est spécialisée dans le conseil des campagnes électorales.

Notre mission consiste à développer une solution permettant de prédire les résultats des prochaines élections présidentielles. Pour ce faire, nous prévoyons d'utiliser des techniques de machine learning en ayant au préalable sélectionné à l'aide d'une analyse approfondie de nombreux indicateurs clés, tels que la démographie, le taux de chômage et surtout les résultats des quatre dernières élections présidentielle mais aussi législative. L'objectif étant d'analyser les données historiques pour prédire les futures évolutions politiques et offrir un outil performant pour notre client.

La réussite de ce projet réside essentiellement dans nos choix en terme de données, d'indicateur clés, nos connaissances socio-économique mais surtout dans la qualité de nos données finales, il s'articule en deux grandes parties distinctes mais complémentaires :

- Une première phase dédiée à l'analyse des données, comprenant la récupération, le nettoyage et l'harmonisation des données provenant de sources fiables comme data.gouv, France Travaille ou l'INSEE. Il est nécessaire de s'assurer que les données utilisées étaient non seulement officielles, mais aussi conformes aux exigences légales et de qualité pour garantir des prédictions robustes.

- Une seconde phase centrée sur le Machine Learning, qui doit nous permettre de prédire quel parti politique serait le vainqueur lors des prochaines élections présidentielles de 2027. L'algorithme développé se base sur des modèles supervisés, utilisant des données électorales historiques et démographiques pour déterminer les futures tendances.

Notre équipe étant composée de quatre membres, nous avons appliqué une méthodologie professionnelle en appliquant la méthode agile pour la gestion de notre projet. Notamment une découpe du travail organisé en sprints par rapport à nos compétences et disponibilité, des réunions régulièrement, un trello pour valider les tâches, discord et teams pour la communication, google drive pour le partage de fichiers et github pour la mise en commun du code.

1. Choix des KPI

a. Choisir une élection

Nous avons décidé de nous diriger vers les élections présidentielles car cela nous semblait être le plus évident et nous savions qu'il y aurait une quantité suffisante de données, en effet ce sont ces élections qui mobilisent le plus de citoyens. C'est aussi celles qui reflètent le mieux les tendances politiques à l'échelle nationale. Elles offrent des données homogènes, avec un nombre limité de candidats et des enjeux constants. Cela facilite l'entraînement et optimise nos modèles prédictifs.

Pour des raisons de quantité de données et afin d'enrichir notre modèle prédictif, nous avons décidé dans un second temps d'intégrer aussi les données relatives aux élections législatives, en s'assurant de la qualité des données et surtout la normalisation du nom des partis politiques pour garder une bonne cohérence. Pour réaliser cette tâche nous avons décidé d'ajouter les colonnes *Parti_Politique_Candidat 1* et *Parti_Politique_Candidat 2* associé à leurs noms et prénoms ainsi que le *Parti_gagnant* dans chacun des jeux de données, puis nous avons associé chaque candidat à son axe politique (gauche, droite, extrême-gauche et extrême droite).

b. Choisir une zone démographique

Nous nous sommes mis d'accord sur la région Occitanie, vaste et peuplée, elle offre un compromis idéal mêlant une population diversifiée avec des données suffisamment variées et pertinentes pour l'analyse électorale. Ce choix garantit ainsi une étude ciblée, mais représentative des dynamiques électorales françaises.

De plus, nous trouvons pertinent de parler de notre belle région.

c. Choisir des éléments de corrélations

Afin d'enrichir notre analyse prédictive, notre choix s'est porté sur l'intégration des deux éléments de corrélation suivants :

Taux de chômage :

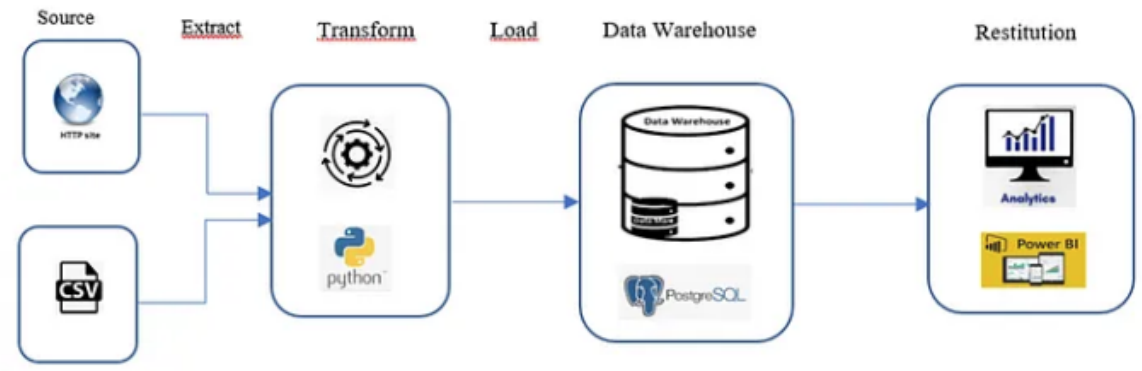
Un fort taux de chômage peut refléter un mécontentement social, économique et orienter les votes vers des candidats proposant des solutions radicales voire moins conventionnelles. En incluant cet indicateur clé dans notre modèle, nous permettrons à notre modèle de machine learning de détecter les liens entre les conditions économiques locales et les résultats électoraux.

Démographie :

Les données démographiques apportent des précisions sur la quantité de population de notre région et cela pour toutes les communes. Ce qui nous permet d'enrichir et d'apporter des précisions supplémentaires sur un grand nombre de nos données. Notamment, en comparant le nombre de votant avec la population totale. Nous développerons davantage cet aspect dans la partie Datavisualisation.

2. Définir une architecture Big Data et Business Intelligence

Dans cette section, nous allons détailler la stratégie complète mise en place pour la collecte, le traitement et l'analyse des données électorales et socio-économiques en suivant les étapes de l'approche ETL (Extract, Transform, Load). Cela nous a permis de construire un pipeline de données efficace, depuis l'acquisition des données jusqu'à leur exploitation dans un contexte de Business Intelligence.



a. Extract (extraction des données)

La première étape consistait à extraire les données à partir de sources fiables. Nos sources principales incluent des jeux de données publics disponibles sur data.gouv.fr pour les données relatives aux élections, l'INSEE pour la démographie, et Francetravail.fr pour le taux de chômage.

Les données comprenaient :

- **Résultats des élections présidentielles**
- **Résultats des élections législatives**
- **Indicateurs socio-économiques** (taux de chômage, population, etc..)

Nous avons téléchargé nos données au format XLS avant de les convertir en CSV comme format principal pour la collecte. Le format CSV est plus léger, simple, universel et mieux adapté à l'automatisation et l'analyse de données massives.

Pour faire cette extraction, nous avons intégré Apache Spark(PySpark et Pandas), qui nous permet de traiter des données volumineuses tout en restant performant.

b. Transform (Transformation des données)

Après l'extraction, nous avons procédé à la transformation des données pour harmoniser les formats et rendre les données exploitables pour notre modèle de prédiction. Plusieurs opérations ont été réalisées :

- **Nettoyage des données** : suppression des valeurs manquantes ou aberrantes, normalisation des formats (dates, nombres) et homogénéisation des noms de colonnes.
- **Filtrage** : nous avons restreint notre analyse aux élections présidentielles de manière ciblée en se concentrant sur la région Occitanie.
- **Enrichissement** : ajout du jeu de données normalisé sur les élections législatives, de nouvelles colonnes comme le taux d'abstention, la répartition des voix par candidat, les partis gagnants.
- **Jointure des données** : nous avons fusionné plusieurs jeux de données (élections, démographie, chômage) pour obtenir une vue complète, en utilisant des clés communes comme le code département et le code commune.

L'utilisation de Pandas et PySpark a permis de gérer ces transformations efficacement, même pour de grands volumes de données. Nous avons également défini des corrélations entre différentes données pour permettre au modèle de détecter les meilleures tendances.

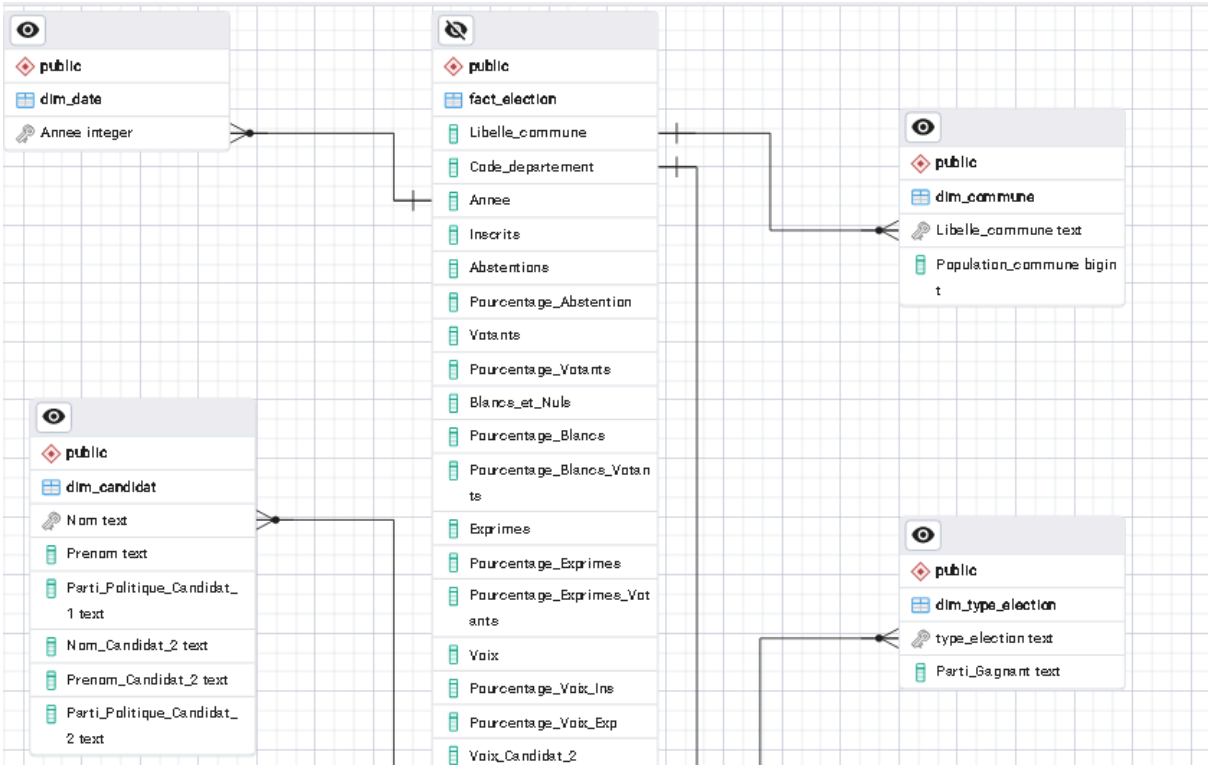
c. Load (Chargement des données)

Initialement, nous avons débuté cette tâche avec HIVE puis, pour poursuivre plus facilement avec un outil que tout le monde maîtrise, nous avons opté pour PostrgreSQL.

La dernière étape de notre ETL consistait à charger les données dans notre entrepôt de données (data warehouse) basé sur PostgreSQL. Cette base de données relationnelle est structurée selon un modèle en étoile, où la table de faits est centrée sur les résultats des élections présidentielles et les tables de dimensions comprennent les données socio-économiques, démographiques, ainsi que les informations sur les candidats.

Ce modèle en étoile facilite les requêtes pour la Business Intelligence et permet d'explorer les données à travers différentes dimensions (années, communes, partis politiques, etc..) Enfin, pour les besoins du Machine Learning, ces données ont été préparées, numérisées et extraites dans des ensembles spécifiques pour entraîner nos modèles de prédiction.

Ce processus ETL nous permet de gérer efficacement l'ensemble du cycle de vie des données, depuis leur collecte jusqu'à leur exploitation à des fins d'analyses prédictives.



Modèle de notre schéma en étoile pour la prédiction des résultats électoraux

README.md | LoadCsvToPostgres.java | fichier_nettoye(2).csv

```java
package fr.epsi.i1cap2024produits.test;
import org.apache.spark.SparkConf;
import org.apache.spark.sql.Dataset;
import org.apache.spark.sql.Row;
import org.apache.spark.sql.SparkSession;
import org.apache.spark.sql.types.DataTypes;

public class LoadCsvToPostgres {  HamzaBenalia
    public static void main(String[] args) {  HamzaBenalia
        // Configuration de Spark
        SparkConf sparkConf = new SparkConf()
                .setAppName("Load CSV to PostgreSQL")
                .setMaster("local[*]");

        // Créer une session Spark
        SparkSession spark = SparkSession.builder()
                .config(sparkConf)
                .getOrCreate();

        // Chemin vers le fichier CSV consolidé
        String filePath = "C:\\Users\\hamza benzy\\Desktop\\Datawhouse\\MSPR_POSTGRES\\src\\main\\resources\\input\\fichier_nettoye(2).csv";

        // Charger les données à partir du fichier CSV
        Dataset<Row> df = spark.read()
                .option("header", "true")
                .option("inferSchema", "true")
                .csv(filePath);
```

README.md | LoadCsvToPostgres.java | fichier_nettoye(2).csv

```java
public class LoadCsvToPostgres {  HamzaBenalia
    public static void main(String[] args) {  HamzaBenalia
        String password = "root";

        // Insérer les données dans la table de faits PostgreSQL
        dfFiltered.write()
                .mode( saveMode: "overwrite")
                .format( source: "jdbc")
                .option("url", jdbcUrl)
                .option("dbtable", "fact_election")
                .option("user", user)
                .option("password", password)
                .save();

        // Vérification des 10 premières lignes
        Dataset<Row> data = spark.read()  DataFrameReader
                .format( source: "jdbc")
                .option("url", jdbcUrl)
                .option("dbtable", "fact_election")
                .option("user", user)
                .option("password", password)
                .load()  Dataset<Row>
                .limit( n: 10);
        data.show();

        // Arrêter la session Spark
        spark.stop();
    }
```

Chargement de nos données dans PostgreSQL via Apache Spark

3. RGPD

Pour notre projet de prédiction des résultats électoraux de 2027 à partir des données historiques, le respect du Règlement Général sur la Protection des Données (RGPD) est primordial.

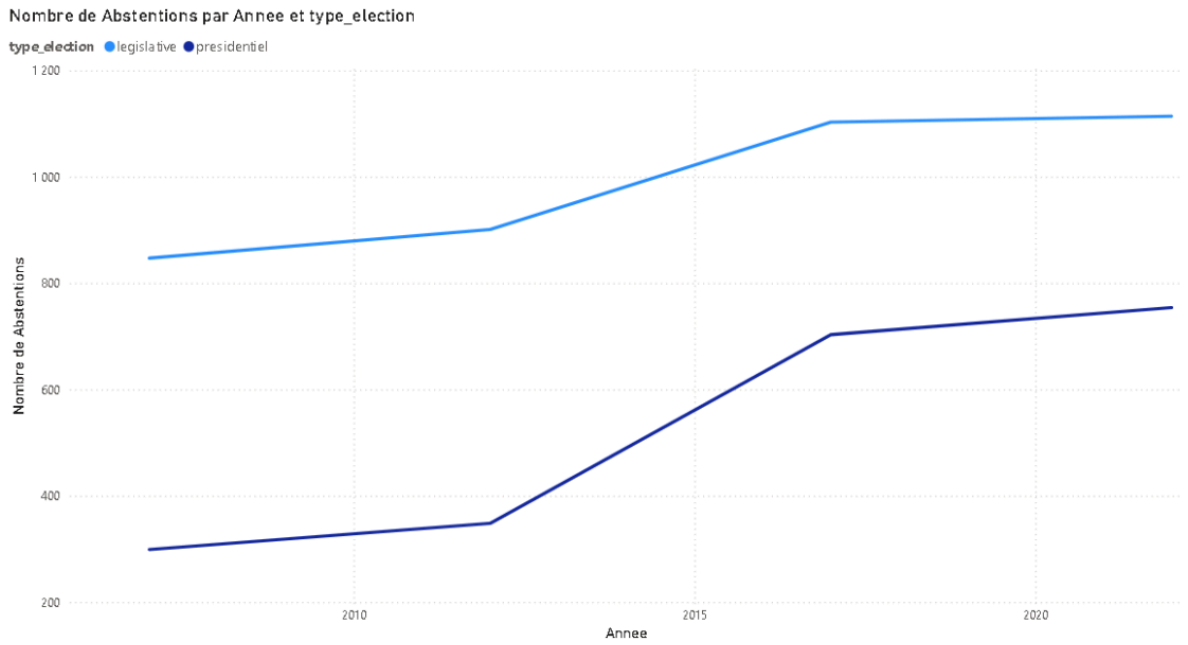
Définition et portée
Le RGPD encadre les droits des individus en matière de confidentialité et leur donne le contrôle de leurs informations personnelles. Les résultats électoraux locaux ont été traités de manière à garantir qu'aucune identification indirecte ne soit possible.

Anonymisation et minimisation des données
Toutes les données ont été soigneusement nettoyées et traitées conformément au principe de minimisation des données du RGPD : seules les informations qui nous semblaient nécessaires à la prévision ont été traitées, sans collecte excessive ou de données non pertinentes.

Transparence et sécurité
Nous avons mis en place des mesures de transparence, notamment la documentation des sources et des procédures de traitement des données, qui se déroulent localement pour minimiser les risques de sécurité. En parallèle, la base de données est sécurisée. Le principe de "privacy by design" a été appliqué dans notre modèle de prédiction, garantissant que la protection des données est intégrée dès la phase de conception. De plus, les principes de "privacy by default" ont également été intégrés, assurant une gestion des données conforme aux exigences du RGPD.

4. Datavisualisation

a. Analyse temporelle des pourcentages d'abstention par type d'élections

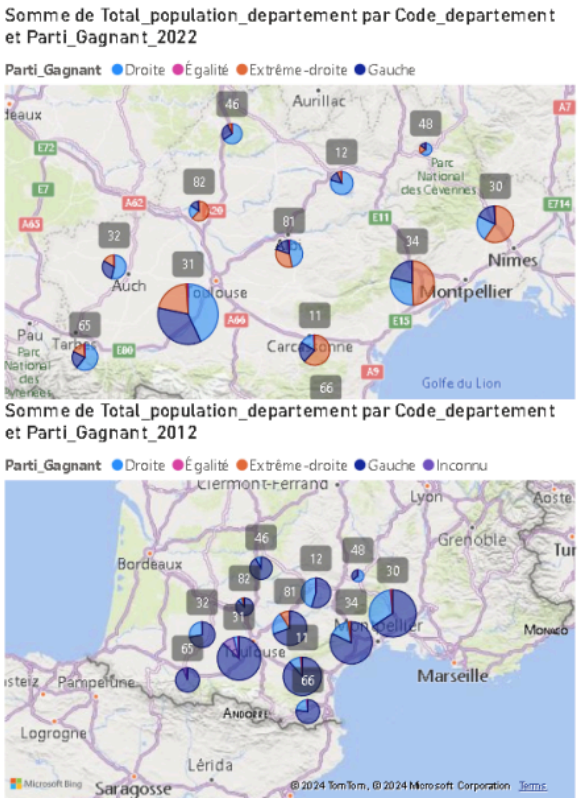


Ce graphique compare les pourcentages d'abstention au fil du temps pour nos deux types d'élections présidentielles et législatives. Voici nos observations :

- **Présidentielles :** Le pourcentage d'abstention est globalement plus élevé pour ce type d'élection, et il suit une tendance à la hausse constante au fil des années.
- **Législatives :** On remarque également une augmentation dans le pourcentage d'abstention, bien que moins prononcée que celle observée pour les présidentielles. Cela pourrait indiquer que l'intérêt ou la participation des électeurs est généralement plus faible pour les élections législatives.

Cela nous permet de constater une tendance haussière quel que soit le type d'élection, ce qui peut indiquer un désengagement croissant de la population.

b. Somme totale de la population départementale par parti gagnant

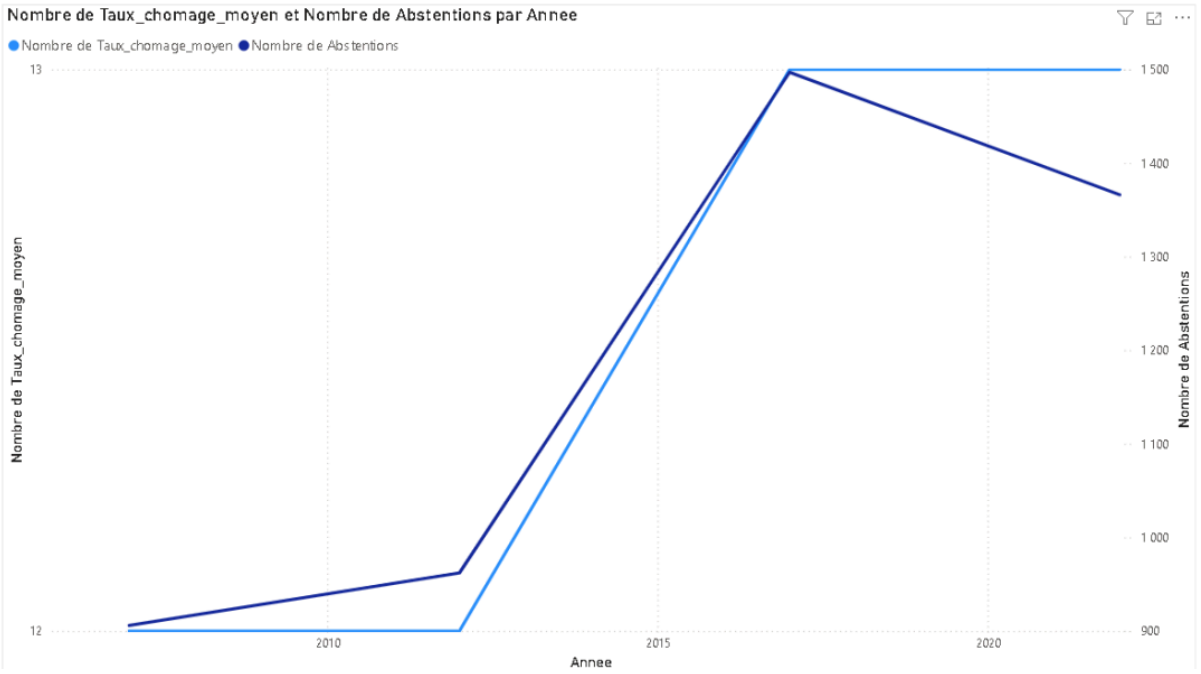


Ce graphique de carte, zoomé sur l'occitanie, utilise des diagrammes circulaires pour représenter la répartition des partis gagnants dans les différents départements, et associe cela à la population totale du département. Voici nos observations :

- **Répartition des partis :** Les départements sont colorés en fonction des partis gagnants (Droite, Gauche, Extrême-droite, etc.), et il semble qu'il y ait une grande variation géographique dans les résultats des élections.
- **Population :** Les tailles des secteurs des diagrammes circulaires donnent une idée de la population relative de chaque département, ce qui permet de visualiser la proportion des populations qui ont soutenu chaque parti.

Cette carte permet de voir la majorité des partis politiques dans les départements ainsi que l'importance démographique des électeurs qui soutiennent chaque parti. Cela donne une bonne perspective sur le poids électoral des partis dans tous les départements de la région.

c. Analyse des tendances de l'évolution du taux de chômage et de l'abstention

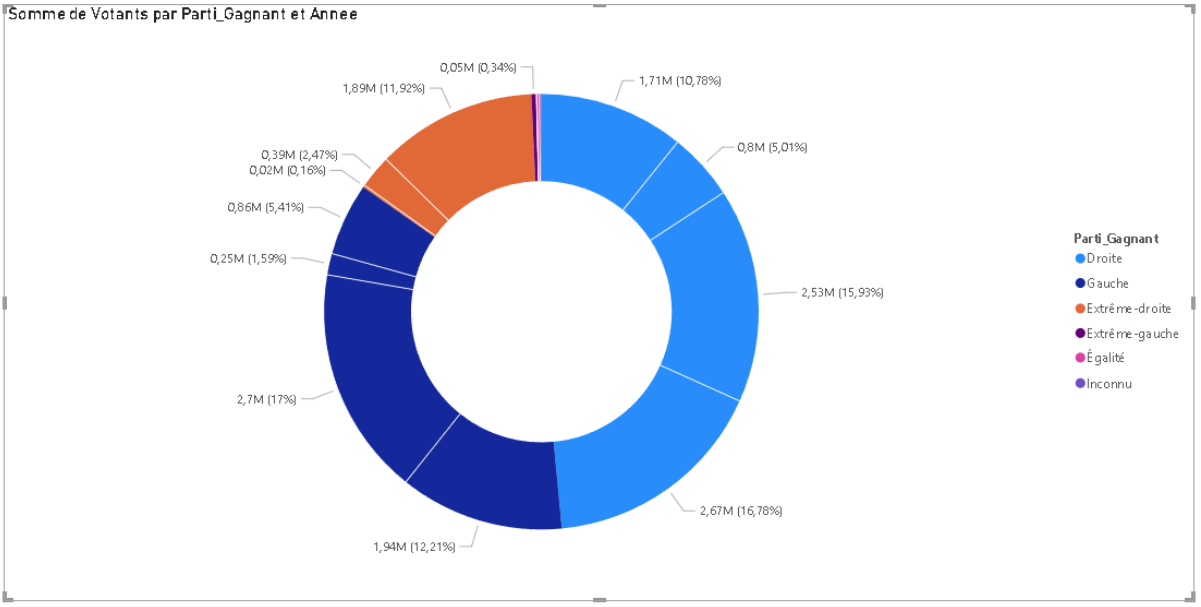


Ce graphique montre une corrélation entre le taux de chômage moyen et le nombre d'abstentions sur les 15 ans entre 2007 et 2022. Voici nos observations :

- **Taux de chômage :** Le taux de chômage semble augmenter au fil du temps, suivant une tendance similaire au nombre d'abstentions.
- **Abstention :** Comme pour le taux de chômage, l'abstention augmente aussi, ce qui pourrait suggérer une relation entre les deux facteurs.

Ce graphique suggère une corrélation potentielle entre le chômage et l'abstention. Plus le chômage est élevé, plus les électeurs sont susceptibles de s'abstenir de voter, ce qui pourrait être le signe d'une désillusion politique accrue parmi les personnes au chômage.

Ce graphique en anneau montre la répartition des votants par parti gagnant et par année, exprimée en millions de votants et en pourcentage.



Parti Dominant :

- La Droite (en bleu) est majoritaire, représentant plusieurs segments significatifs du graphique avec des contributions importantes comme 2,7M (17%), 2,67M (16,78%). Ce parti a obtenu une majorité significative des voix sur les années représentées.

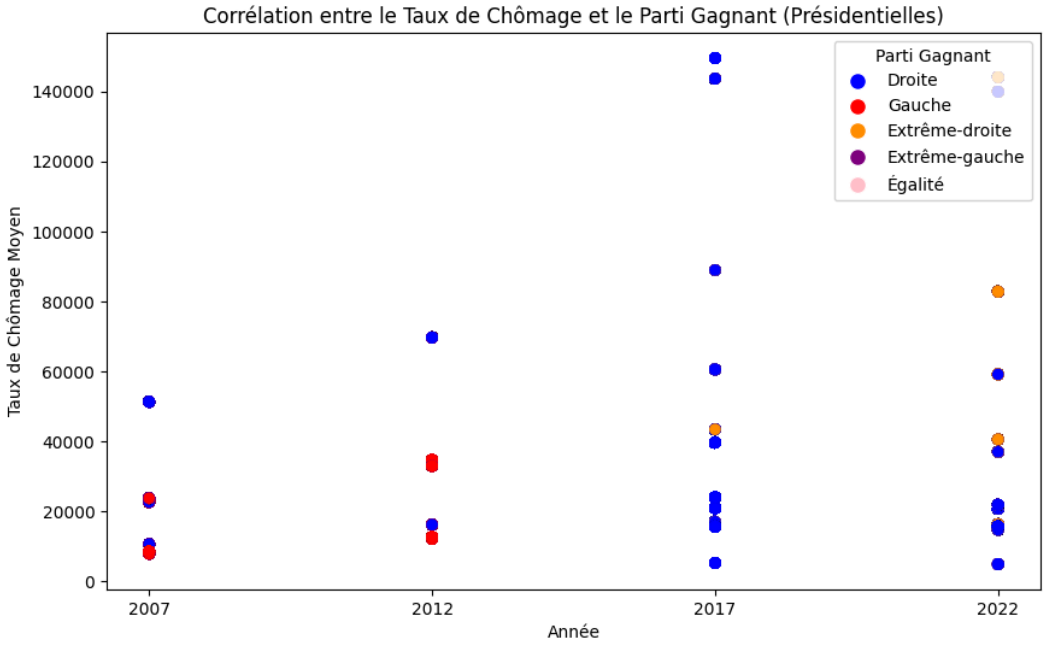
Gauche et Extrême-droite :

- La Gauche (en bleu clair) occupe également une part importante, comme les 1,94M (12,21%) et 2,53M (15,93%). L'Extrême-droite (en orange) a également une contribution notable avec une part de 1,71M (10,78%), bien que cela reste inférieur aux deux principaux partis, la Droite et la Gauche.

Ce graphique en anneau montre clairement la dominance de la Droite dans les élections, avec la Gauche et l'Extrême-droite occupant des positions importantes mais minoritaires. Cette visualisation met en lumière les partis politiques les plus influents au cours des années représentées, la Droite et la Gauche étant les deux forces politiques les plus compétitives.

Le graphique montre la corrélation entre le taux de chômage moyen et le parti gagnant pour les élections présidentielles des années 2007, 2012, 2017, et 2022.

Il révèle une corrélation notable entre un taux de chômage élevé et la victoire de la Droite, particulièrement en 2017 et 2022. La Gauche remporte principalement des départements avec des taux de chômage plus bas en 2007 et 2012, tandis que l'Extrême-droite gagne en influence en 2022 dans des départements à taux de chômage modérés. Le graphique reflète l'impact potentiel du chômage sur les choix électoraux, avec une tendance pour les départements les plus touchés à voter pour la Droite ou l'Extrême-droite dans les récentes élections.



Années et Distribution :

- Le graphique est divisé par année, de 2007 à 2022, avec chaque point représentant un département et son taux de chômage moyen. Les couleurs des points indiquent le parti gagnant dans chaque département, les partis représentés étant : Droite (bleu), Gauche (rouge), Extrême-droite (orange), Extrême-gauche (violet), et Égalité (rose).

Taux de Chômage et Partis :

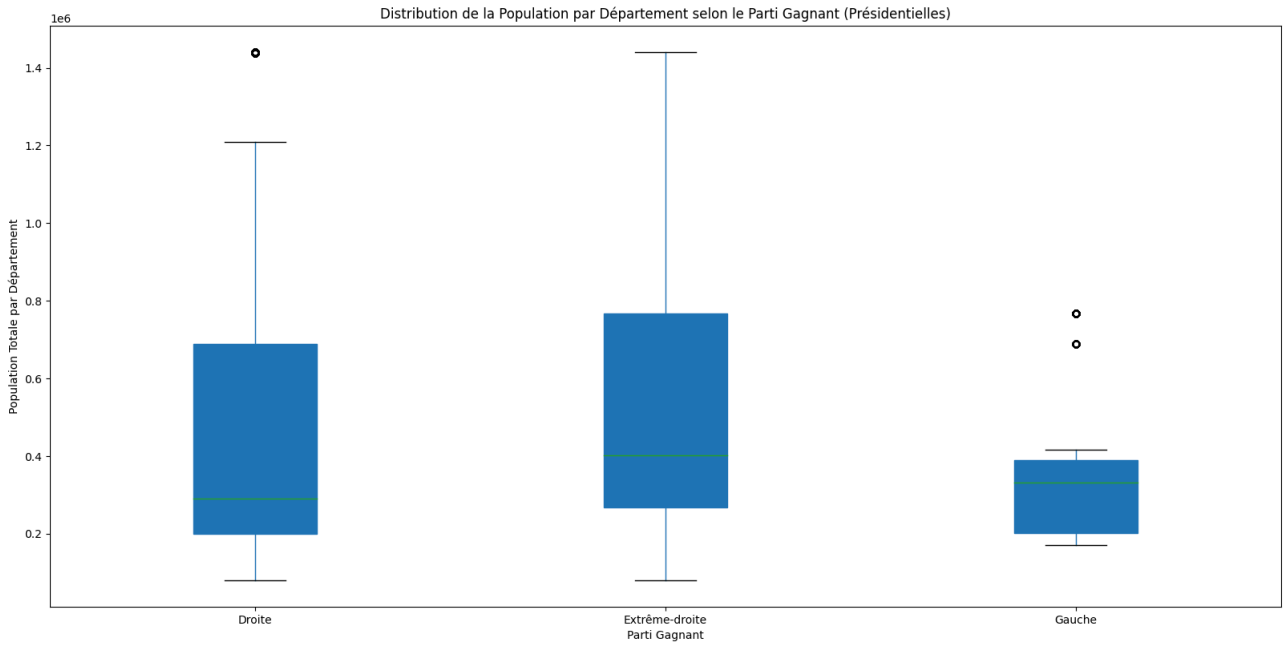
- Droite (bleu) : Domine les élections dans les départements avec des taux de chômage moyens élevés, particulièrement visibles en 2017 et 2022, où on voit plusieurs points bleus à des taux de chômage élevés (60 000 à 140 000).
- Gauche (rouge) : En 2007 et 2012, la Gauche a remporté des départements avec des taux de chômage modérés (entre 20 000 et 40 000), mais elle est beaucoup moins présente en 2017 et 2022.

- Extrême-droite (orange) : Devient plus présente en 2022, remportant des départements avec des taux de chômage variés. Cela suggère une montée en puissance de ce parti dans des régions touchées par un chômage significatif.
- Extrême-gauche (violet) et Égalité (rose) : ont une présence négligeable et ne remportent aucun département.

Corrélation :

- Le graphique suggère une corrélation entre un taux de chômage élevé et la victoire de la Droite, surtout visible en 2017 et 2022. Cependant, on ne voit pas de tendance claire pour la Gauche, qui semble gagner dans des départements à des taux de chômage plus modérés.
- La montée de l'Extrême-droite dans les départements avec des taux de chômage moyens en 2022 pourrait indiquer un changement dans les tendances électorales en réponse à des défis économiques.

Ce graphique ci-dessous est un boxplot qui montre la distribution de la population par département en fonction du parti gagnant pour les élections présidentielles. Les partis représentés ici sont la Droite, l'Extrême-droite, et la Gauche.



Ce boxplot met en évidence les différences démographiques entre les départements qui soutiennent les différents partis politiques, en particulier la dominance de la Droite dans des départements à forte population, et la Gauche dans des départements plus modestes en taille.

Parti Droite :

- La médiane de la population des départements où la Droite a gagné se situe autour de 250 000 habitants.
- La boîte s'étend de 200 000 à environ 600 000, ce qui montre une forte variabilité dans la population des départements gagnés par la Droite.
- Il y a un outlier au-dessus de 1,4 million, ce qui signifie qu'il y a au moins un département exceptionnellement grand en termes de population où la Droite a gagné.

Parti Extrême-droite :

- La médiane de la population des départements où l'Extrême-droite a gagné est similaire à celle de la Droite, aux alentours de 250 000 habitants.
- La boîte pour l'Extrême-droite montre une distribution plus homogène avec moins de variabilité par rapport à la Droite, et s'étend de 200 000 à environ 600 000.
- Aucune valeur aberrante n'est visible pour ce parti, ce qui signifie que la population des départements où l'Extrême-droite a gagné est plus consistante.

Parti Gauche :

- La population des départements où la Gauche a gagné est notablement plus faible. La médiane se situe autour de 150 000 habitants.
- La boîte montre une faible variabilité, s'étendant d'environ 100 000 à 200 000 habitants.
- Il y a quelques outliers au-dessus de 250 000, ce qui signifie que dans certains départements, la Gauche a gagné malgré une population plus importante, mais cela reste une exception.

Droite :

- Gagne principalement dans les départements avec des populations très variables, allant de départements moyens à très peuplés.

Extrême-droite :

- Remporte des départements avec une population plus homogène, dans une tranche similaire à celle de la Droite, mais avec moins de variabilité.

Gauche :

- A tendance à gagner dans des départements avec des populations plus petites, autour de 150 000 habitants, avec moins de variabilité par rapport aux autres partis.

5. Machine Learning

L'apprentissage supervisé est une technique de Machine Learning où le modèle est entraîné sur un ensemble de données étiquetées, c'est-à-dire que chaque exemple d'entraînement est associé à une sortie correcte (label). Le but est d'apprendre une fonction qui, à partir de nouvelles données non vues, puisse prédire la sortie correcte avec un haut degré de précision. Dans notre cas, nous utilisons des résultats d'élections passées et d'autres données de corrélations pour entraîner le modèle à prédire le parti gagnant lors d'une prochaine élection.

Random_Forest.py > ...

```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.ensemble import RandomForestClassifier
from sklearn.impute import SimpleImputer
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sqlalchemy import create_engine

# Connexion à la base de données PostgreSQL
DATABASE_TYPE = 'postgresql'
DBAPI = 'psycopg2'
ENDPOINT = 'localhost'  # Remplace par l'URL de ta BDD
USER = 'postgres'
PASSWORD = 'root'
PORT = 5432
DATABASE = 'dbtest'

engine = create_engine(f"{DATABASE_TYPE}+{DBAPI}://{USER}:{PASSWORD}@{ENDPOINT}:{PORT}/{DATABASE}")

# Charger les données depuis PostgreSQL
try:
    df = pd.read_sql("SELECT * FROM fact_election", engine)
    print("Données chargées avec succès.")
except Exception as e:
    print(f"Erreur lors du chargement des données: {e}")
```

Cette première partie du code se concentre sur la connexion avec notre base de données PostgreSQL où les données d'élection sont stockées. Elle utilise le package SQLAlchemy pour créer un moteur de connexion à la base de données et charger les données de la table *Fact_Election* dans un DataFrame Pandas. Ce processus permet de transférer les données d'une base relationnelle vers l'environnement de traitement, où elles peuvent être analysées et utilisées pour le modèle de Machine Learning. La gestion des erreurs permet de détecter et d'afficher tout problème potentiel pendant le chargement des données.

Cette section du code initialise tout le nécessaire afin d'interagir avec la base de données, charger les données, les manipuler, puis entraîner le modèle de machine learning.

a. Quelques détails sur les bibliothèques

Pandas (pd) : Cette bibliothèque permet de manipuler et d'analyser des données tabulaires, comme des fichiers CSV, nous l'avons utilisée pour le traitement de données structurées.

Numpy (np) : C'est une bibliothèque pour effectuer des opérations mathématiques et des manipulations sur des tableaux multi-dimensionnels. Très utile pour les algorithmes de machine learning.

Matplotlib (plt) et Seaborn (sns) : Utilisées pour la visualisation des données, nous avons utilisé ces bibliothèques afin de créer des graphiques et des diagrammes pour mieux comprendre la corrélation entre les données.

Scikit-learn (train_test_split, LabelEncoder, StandardScaler, RandomForestClassifier, SimpleImputer, accuracy_score, classification_report, confusion_matrix) :

- train_test_split : Divise les données en ensembles d'entraînement et de test.
- LabelEncoder : Transforme les valeurs catégoriques en valeurs numériques.
- StandardScaler : Normalise les données en les centrant et en les mettant à l'échelle.
- RandomForestClassifier : Implémente l'algorithme Random Forest pour la classification des données.
- SimpleImputer : Permet de gérer les valeurs manquantes en remplissant par une statistique (comme la médiane ici).
- accuracy_score, classification_report, confusion_matrix : Ces fonctions servent à évaluer la performance du modèle de machine learning.

Cette section effectue une première analyse des données pour évaluer leur qualité. Le code commence par afficher un aperçu des premières lignes du DataFrame pour s'assurer que les données ont été chargées correctement. Ensuite, il génère des statistiques descriptives qui sont sauvegardées dans un fichier CSV, permettant une vue d'ensemble des valeurs moyennes, des écarts-types, etc. Après cette analyse, les valeurs manquantes sont identifiées et traitées. Les données contenant des labels indésirables, comme "Inconnu" ou "Extrême-gauche", sont filtrées. Enfin, les valeurs manquantes des colonnes numériques sont imputées avec la médiane, une stratégie que nous avons utilisée pour éviter de biaiser les distributions lors du traitement de données manquantes.

```
30  # Étape 1 : Analyse de la qualité des données
31  print("Aperçu des données :")
32  print(df.head())
33
34  # Statistiques descriptives
35  statistiques_descriptives = df.describe(include='all')
36  statistiques_descriptives.to_csv('statistiques_descriptives.csv', index=False)
37  print("Statistiques descriptives enregistrées dans 'statistiques_descriptives.csv'.")
38
39  # Vérifier les valeurs manquantes
40  valeurs_manquantes = df.isnull().sum()
41  print("Valeurs manquantes par colonne :")
42  print(valeurs_manquantes)
43
44  df = df[(df['Parti_Gagnant'] != 'Inconnu') & (df['Parti_Gagnant'] != 'Extrême-gauche') & (df['Parti_Gagnant'] !=
45
46  # Remplir les valeurs manquantes avec la médiane pour les colonnes numériques
47  imputer = SimpleImputer(strategy='median')
48  df[df.select_dtypes(include=[np.number]).columns] = imputer.fit_transform(df.select_dtypes(include=[np.number]))
49
50  # Visualisation de la matrice de corrélation
51  numerical_df = df.select_dtypes(include=[np.number])
52  plt.figure(figsize=(10, 8))
53  sns.heatmap(numerical_df.corr(), annot=True, cmap='coolwarm')
54  plt.title("Matrice de corrélation")
55  plt.savefig('correlation_matrix.png')
56  plt.show()
57  print("Matrice de corrélation enregistrée dans 'correlation_matrix.png'.")
```

Cette étape consiste à analyser la qualité de tes données et à les préparer pour l'entraînement du modèle. Elle inclut des actions importantes telles que :

- La gestion des valeurs manquantes,
- Le filtrage des données non pertinentes,
- La visualisation des corrélations pour mieux comprendre les relations entre les variables.

L'objectif est d'obtenir un ensemble de données propre et prêt à être utilisé pour le machine learning.

Cette section prépare les données pour l'entraînement du modèle de Machine Learning. Tout d'abord, les caractéristiques "features" utilisées pour prédire le gagnant sont extraites en excluant la colonne cible *Parti_Gagnant*. Ensuite, les données sont normalisées à l'aide de la classe StandardScaler pour s'assurer que toutes les variables sont sur la même échelle. Le jeu de données est ensuite divisé en deux ensembles, un ensemble d'entraînement pour ajuster le modèle (60%) et un ensemble de tests (40%) pour évaluer ses performances.

```
# Étape 2 : Préparation des données pour l'entrainement
features = numerical_df.columns.drop('Parti_Gagnant', errors='ignore')
X = df[features]
y = df['Parti_Gagnant']

# Normaliser les données
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Séparer les ensembles d'entrainement et de test
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.4, random_state=42)

# Étape 3 : Entrainer le modèle
model = RandomForestClassifier(class_weight='balanced', random_state=42, n_estimators=100)  # Utiliser Random For
model.fit(X_train, y_train)

# Étape 4 : Évaluer les performances
y_pred = model.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy}")

classification_rep = classification_report(y_test, y_pred, zero_division=0)
print("Classification Report:")
print(classification_rep)

confusion_mat = confusion_matrix(y_test, y_pred)
print("Confusion Matrix:")
print(confusion_mat)
```

La répartition 60/40 offre un bon compromis entre un entraînement suffisant du modèle et une bonne évaluation de ses performances sur un ensemble de test représentatif. Cela garantit une meilleure capacité de prédiction future tout en réduisant les risques d'overfitting et en assurant une bonne généralisation.

Cette étape couvre l'entraînement du modèle de random forest et l'évaluation de ses performances. Le processus inclut :

- La séparation des données en ensembles d'entraînement et de test,
- La normalisation des données pour rendre les features comparables,
- L'entraînement du modèle sur les données d'entraînement,
- L'évaluation des performances sur les données de test.

Cette section clôt le processus d'entraînement en sauvegardant à la fois les résultats des performances (exactitude, matrice de confusion, etc.) dans un fichier texte et les prédictions du modèle dans un fichier CSV. Cela facilite la comparaison des prédictions avec les vraies valeurs et permet une analyse post-modélisation plus approfondie.

```python
# Enregistrer les performances du modèle
with open('random_forest_performance.txt', 'w') as f:
    f.write(f"Accuracy: {accuracy}\n\n")
    f.write("Classification Report:\n")
    f.write(classification_rep)
    f.write("\nConfusion Matrix:\n")
    f.write(str(confusion_mat))
print("Les performances du modèle ont été enregistrées dans 'random_forest_performance.txt'.")

# Étape 5 : Sauvegarder les prédictions
# Encoder la colonne 'Parti_Gagnant' avec toutes les classes possibles
le_gagnant = LabelEncoder()
le_gagnant.fit(df['Parti_Gagnant'])  # Ajuster le LabelEncoder

# S'assurer que y_test et y_pred sont bien des entiers
y_test = y_test.astype(int)
y_pred = y_pred.astype(int)

# Inverse transform des prédictions
y_test_inverse = le_gagnant.inverse_transform(y_test)
y_pred_inverse = le_gagnant.inverse_transform(y_pred)

# Sauvegarder les prédictions
output_df = pd.DataFrame({
    'Vraie_valeur': y_test_inverse,
    'Prediction': y_pred_inverse
})
output_df.to_csv('predictions_random_forest.csv', index=False)
print("Les prédictions ont été enregistrées dans 'predictions_random_forest.csv'.")

```

Résultat après exécution du script :

```
Matrice de corrélation enregistrée dans 'correlation_matrix.png'.
Accuracy: 0.9357094133697135
Classification Report:
               precision    recall  f1-score   support

       Droite       0.92      0.96      0.94      5838
Extrême-droite       0.95      0.92      0.93      1485
       Gauche       0.95      0.91      0.93      4405

     accuracy                           0.94     11728
    macro avg       0.94      0.93      0.94     11728
 weighted avg       0.94      0.94      0.94     11728

Confusion Matrix:
[[5585   68  185]
 [ 106 1365   14]
 [ 376    5 4024]]
```

b. Matrice de corrélation

Le code commence par l'affichage et l'enregistrement de la matrice de corrélation, qui permet de visualiser les relations entre les différentes variables numériques dans le jeu de données. Cette matrice est essentielle pour comprendre les liens entre les variables et pour identifier les éventuelles variables redondantes ou fortement corrélées qui pourraient influencer le modèle.

c. Précision globale du modèle (Accuracy)

L'accuracy, ou précision, est une manière simple de juger les performances d'un modèle de classification. Elle correspond à la proportion des prédictions correctes par rapport au total des prédictions effectuées.dans notre cas, avec un accuracy de 93.6%, cela signifie qu'il a fait les bonnes prédictions dans 93.6% des cas sur les données de test.

Le rapport de classification détaille les performances du modèle pour les trois principales classes politiques : Droite, Extrême-droite, et Gauche.

- **Précision** : Le modèle prédit correctement les cas positifs pour chaque classe avec des scores de 0.92 pour Droite, 0.95 pour Extrême-droite, et 0.95 pour Gauche.
- **Rappel** : Il identifie bien les cas effectivement positifs, avec un rappel de 0.96 pour Droite, 0.92 pour Extrême-droite, et 0.91 pour Gauche.
- **F1-Score** : Une bonne balance entre précision et rappel avec des scores F1 élevés : Droite (0.94), Extrême-droite (0.93), et Gauche (0.93).

Le modèle montre ainsi un excellent équilibre dans la classification des trois groupes.

En examinant la **matrice de confusion** :

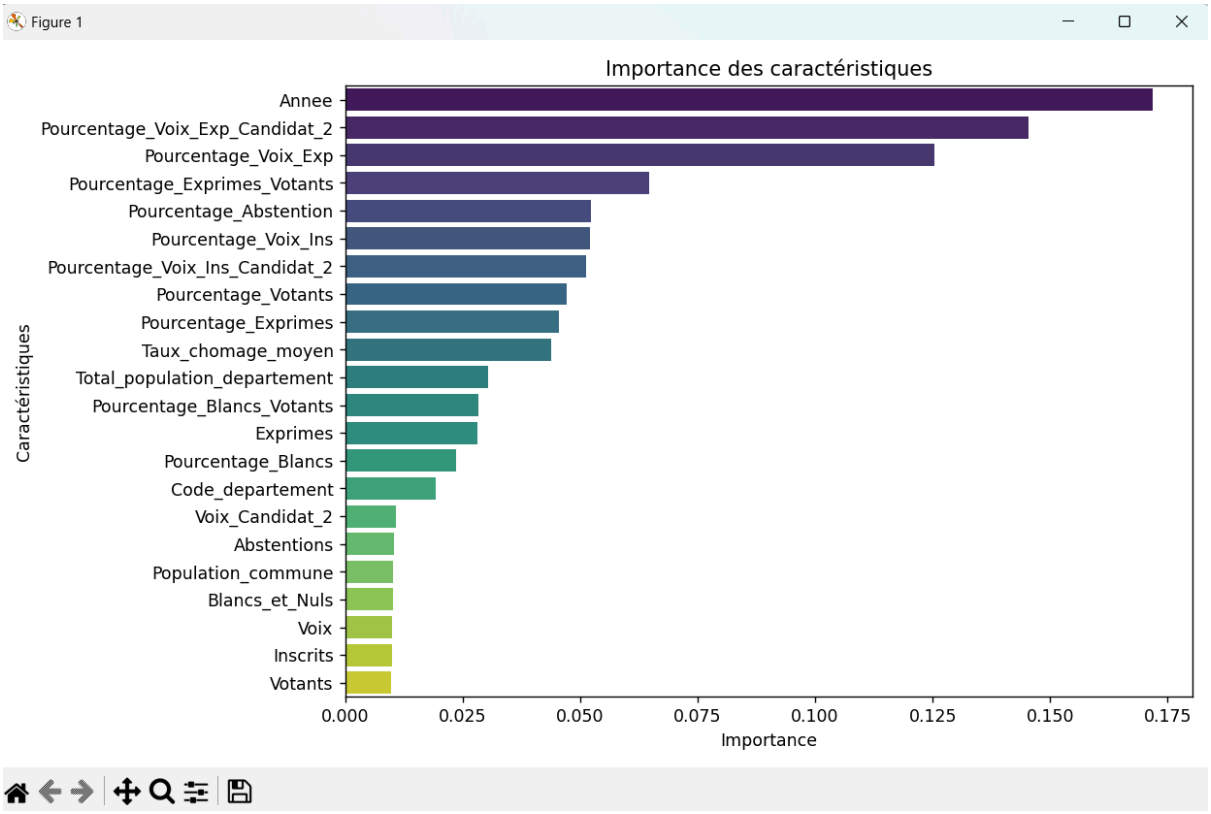
- Pour **Droite**, sur 5838 cas, 5585 ont été correctement prédits, avec quelques confusions avec Gauche (185) et Extrême-droite (68).
- Pour **Extrême-droite**, sur 1485 cas, 1365 ont été correctement prédits, avec (106) confusions avec Droite et (14) avec Gauche.
- Pour **Gauche**, sur 4405 cas, 4024 ont été correctement prédits, avec des confusions principalement avec Droite (376) et très peu avec Extrême-droite (5).

Cela indique que le modèle gère globalement bien la classification, avec quelques confusions entre Droite et Gauche.

d. Conclusion des résultats

Les résultats montrent que le modèle Random Forest est efficace pour prédire les résultats électoraux en Occitanie, avec une précision globale de 93.6%. Il présente un excellent équilibre entre précision et rappel pour les trois classes principales (Droite, Extrême-droite, Gauche). La matrice de confusion révèle que les principales confusions se produisent entre les classes de Droite et Gauche, ce qui est compréhensible étant donné leurs proximités dans certaines régions.

D'après le graphique ci-dessous, on remarque que l'année de l'élection ("Année") est le facteur le plus déterminant dans la prédiction des résultats électoraux. Ensuite viennent des variables directement liées aux votes, comme le *Pourcentage_Voix_Exp_Candidat_2* et le *Pourcentage_Voix_Exp*, qui mesurent la proportion de voix exprimées pour chaque candidat. Des indicateurs comme le *Pourcentage_Abstention* et le *Pourcentage_Exprimes_Votants* montrent aussi une forte influence, mettant en avant l'importance du taux de participation et des votes valides.



Cependant, il est surprenant de voir que des facteurs socio-économiques comme le taux de chômage et la population ont un poids relativement faible dans ce modèle. Ces variables sont généralement perçues comme influentes sur les comportements électoraux, mais ici, elles semblent moins corrélées aux résultats prédits.

Cela s'explique par la nature des données que nous avons utilisées et le type de modèle appliqué. Le Random Forest a tendance à privilégier les variables qui ont une relation directe avec l'objectif à prédire, en l'occurrence le résultat de l'élection. Des variables comme les pourcentages de voix ou d'abstention, qui reflètent immédiatement le comportement électoral, prennent donc plus d'importance. Les facteurs comme le taux de chômage peuvent avoir un impact plus subtil ou se faire sentir sur une plus longue période. Cela pourrait expliquer pourquoi, dans ce modèle, ils semblent avoir moins d'influence immédiate sur les résultats.

Cela ne signifie pas que ces variables sont sans importance. Elles pourraient jouer un rôle plus important dans d'autres types de modèles, comme par exemple avec XGBoost (voir annexe)

Conclusion

Au terme de ce projet, nous avons réussi à mettre en place une **architecture de Big Data et Business Intelligence**, combinant des données **électorales et socio-économiques** afin de prédire les résultats des **élections présidentielles** en **Occitanie**.

En suivant une approche **ETL** complète, nous avons extrait, transformé et chargé des données provenant de sources fiables et officielles, tout en nous assurant de la **qualité** et de **l'intégrité** des nos données. La création d'un modèle **prédictif supervisé**, notamment avec le modèle Random Forest, nous a permis d'obtenir des résultats prometteurs avec une précision de **93,6 %**.

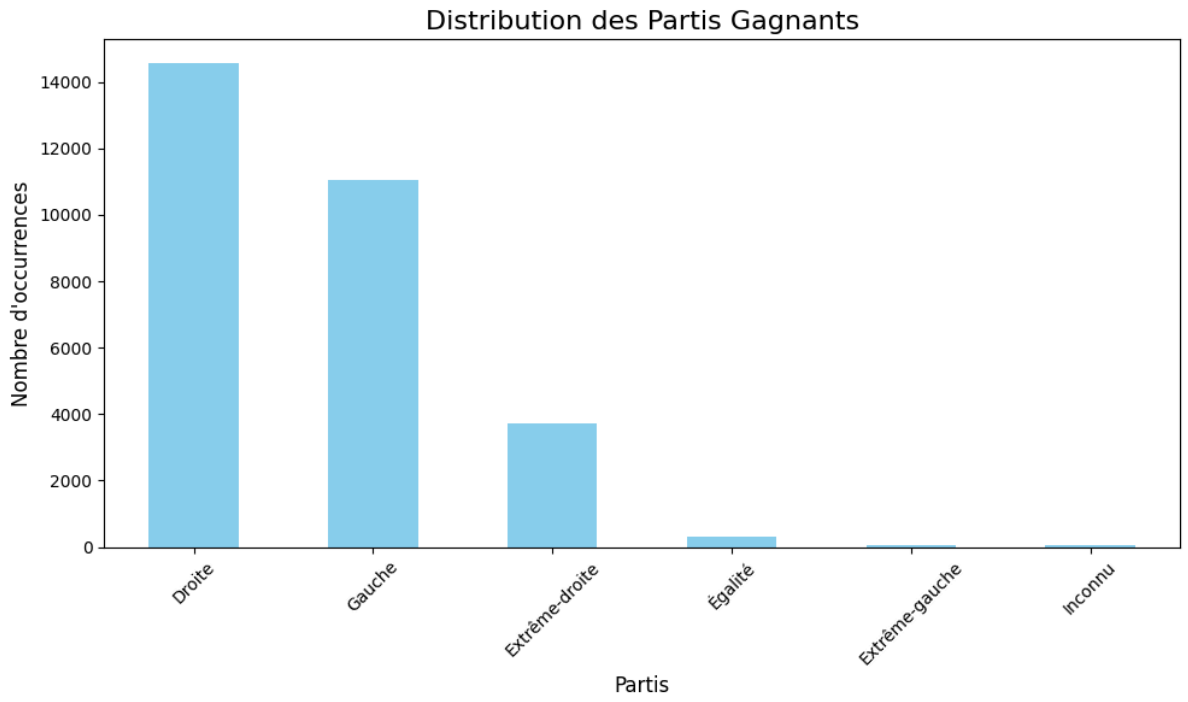
L'**analyse des données** et les résultats obtenus révèlent des corrélations intéressantes, notamment l'importance des variables liées aux **votes exprimés** et au **taux de participation**, ainsi que le rôle **moindre mais significatif des facteurs socio-économiques**, comme le **taux de chômage**. Ce projet a montré que bien que des variables comme l'abstention soient immédiatement corrélées aux résultats, d'autres facteurs contextuels pourraient être plus influents dans des **modèles alternatifs**.

Enfin, nous avons démontré qu'il est possible de **prédire les tendances politiques futures** à travers une **analyse approfondie des données historiques**. Ce travail pose les bases pour des développements futurs(features), notamment en explorant d'autres modèles de machine learning, tels que XGBoost, qui pourraient renforcer encore la précision des prédictions.

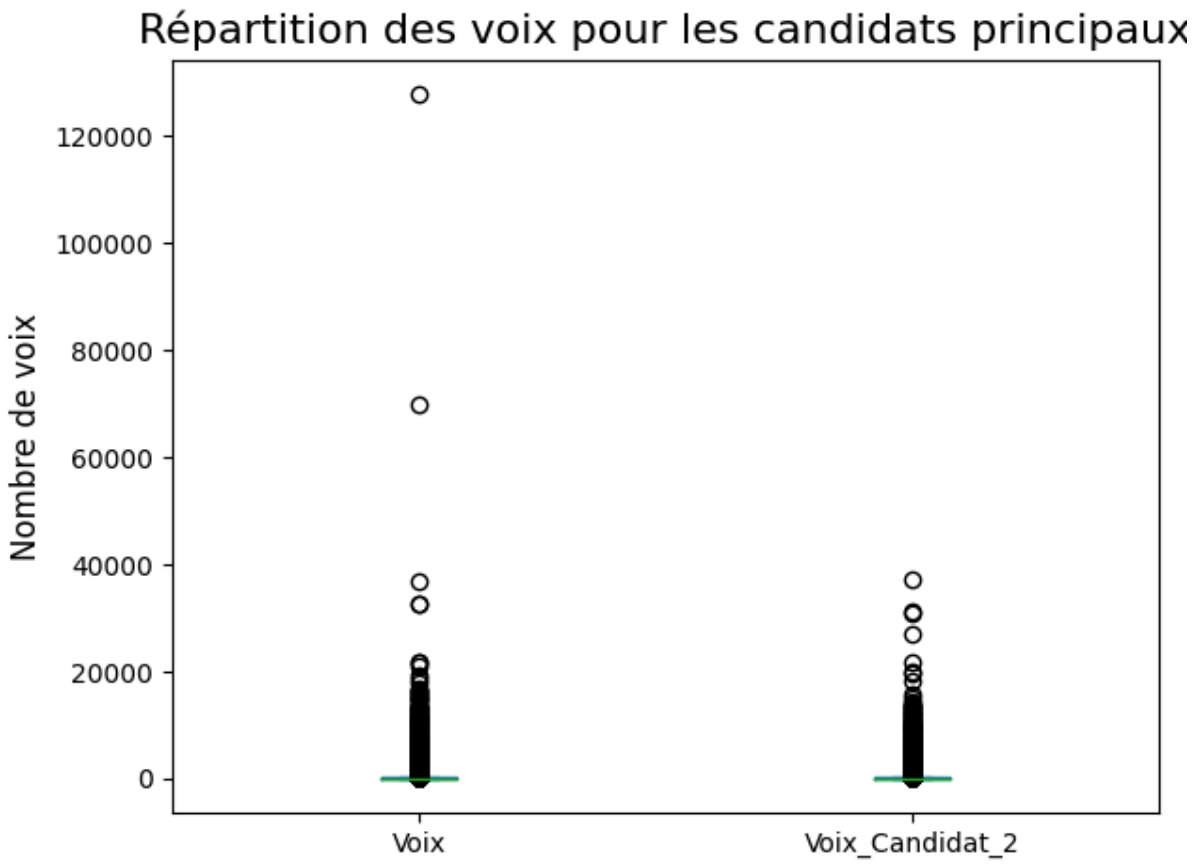
Notre **approche agile et collaborative** a été un facteur clé de succès, nous permettant de répondre aux exigences de notre client tout en restant flexibles face aux défis rencontrés.

Ainsi, ce projet constitue une étape importante dans la compréhension des dynamiques électorales, et offre des perspectives intéressantes pour des analyses prédictives plus poussées à l'avenir.

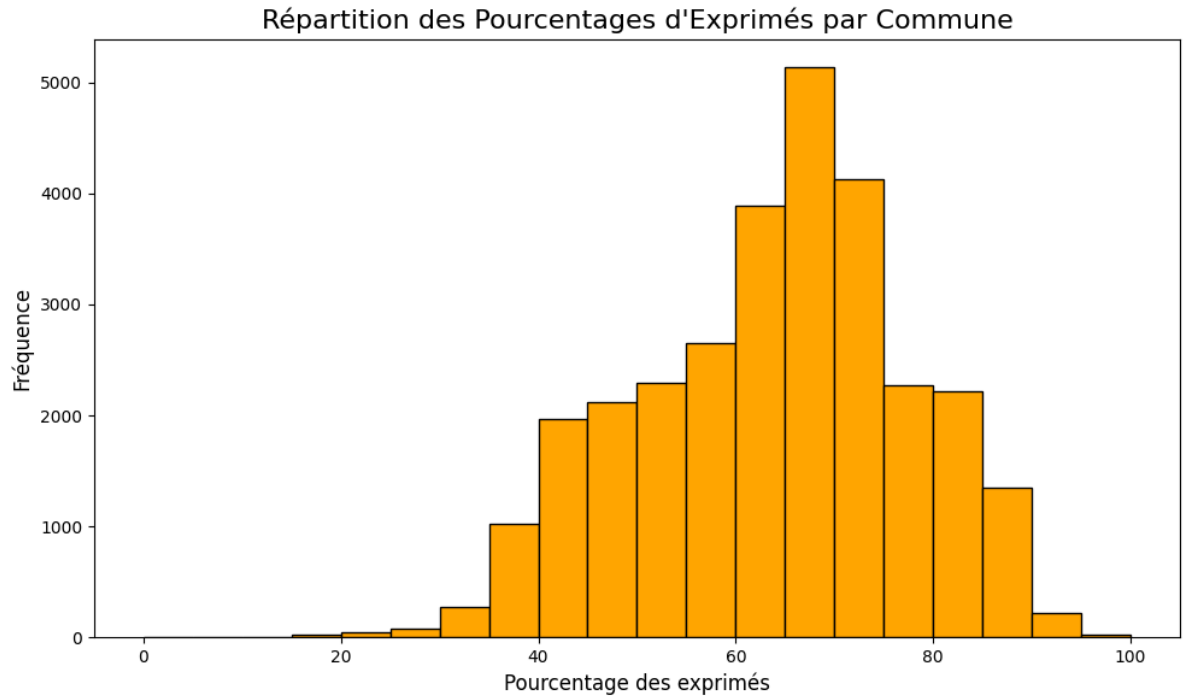
ANNEXE



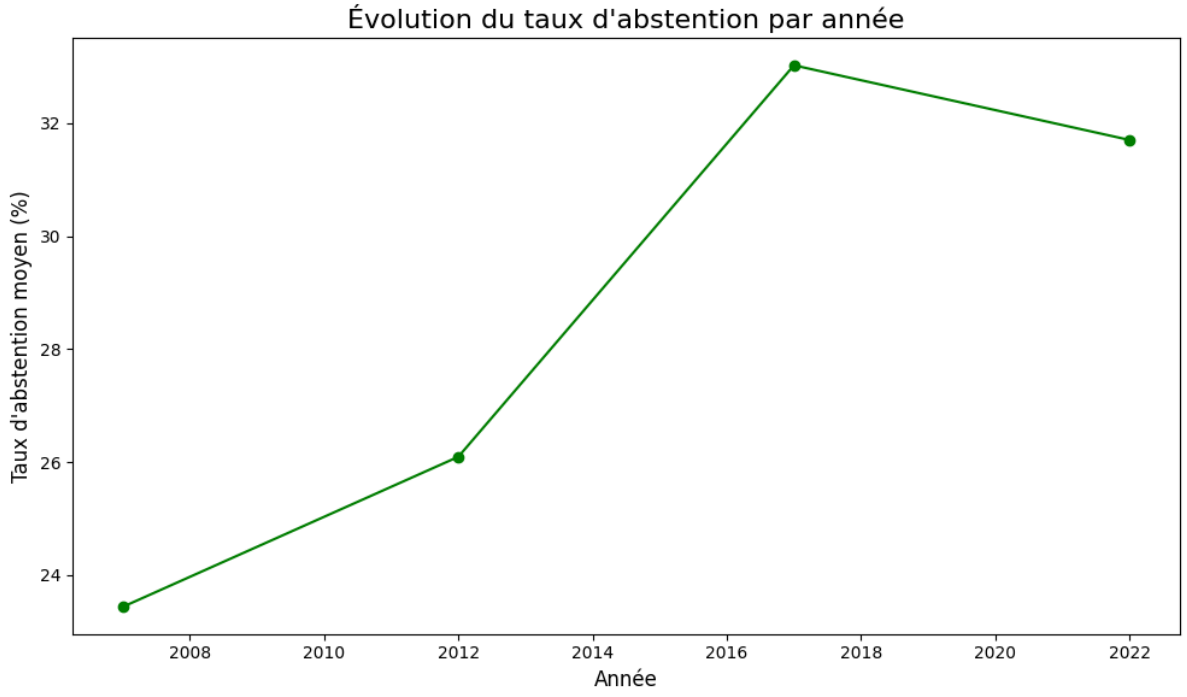
- Synthèse : Ce graphique illustre clairement que les deux principaux partis politiques sont Droite et Gauche, avec une nette avance pour la Droite.



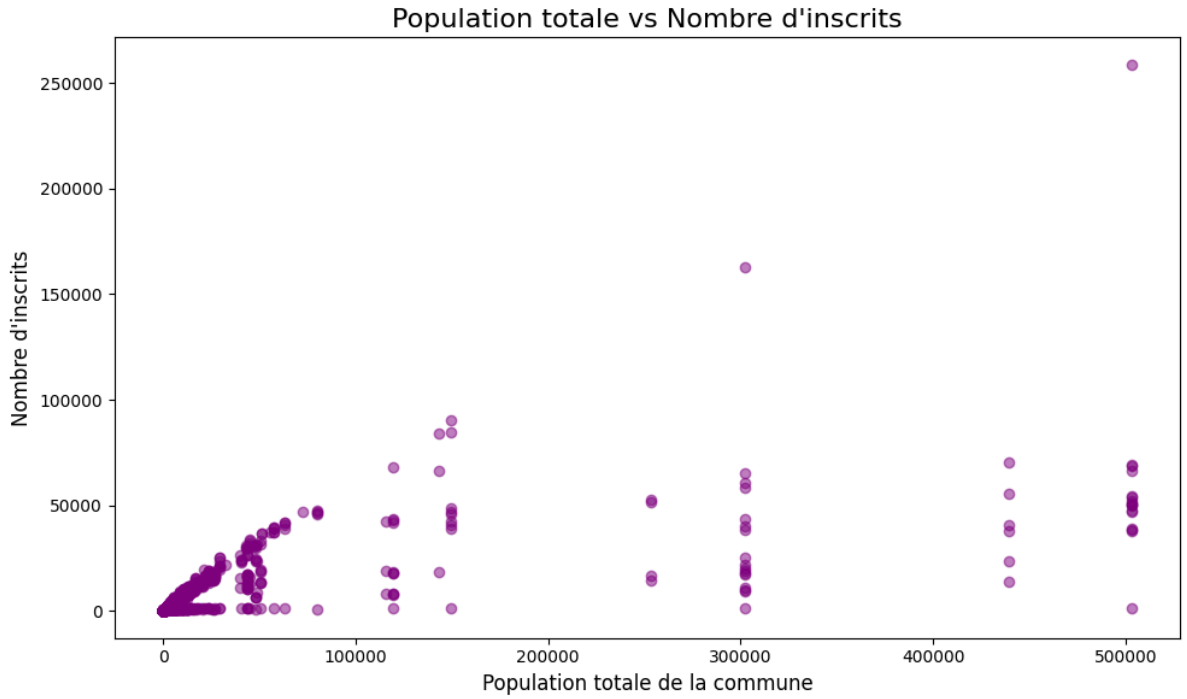
- Synthèse : La distribution des votes est fortement concentrée autour de valeurs basses, mais certaines communes peuvent être des exceptions notables.



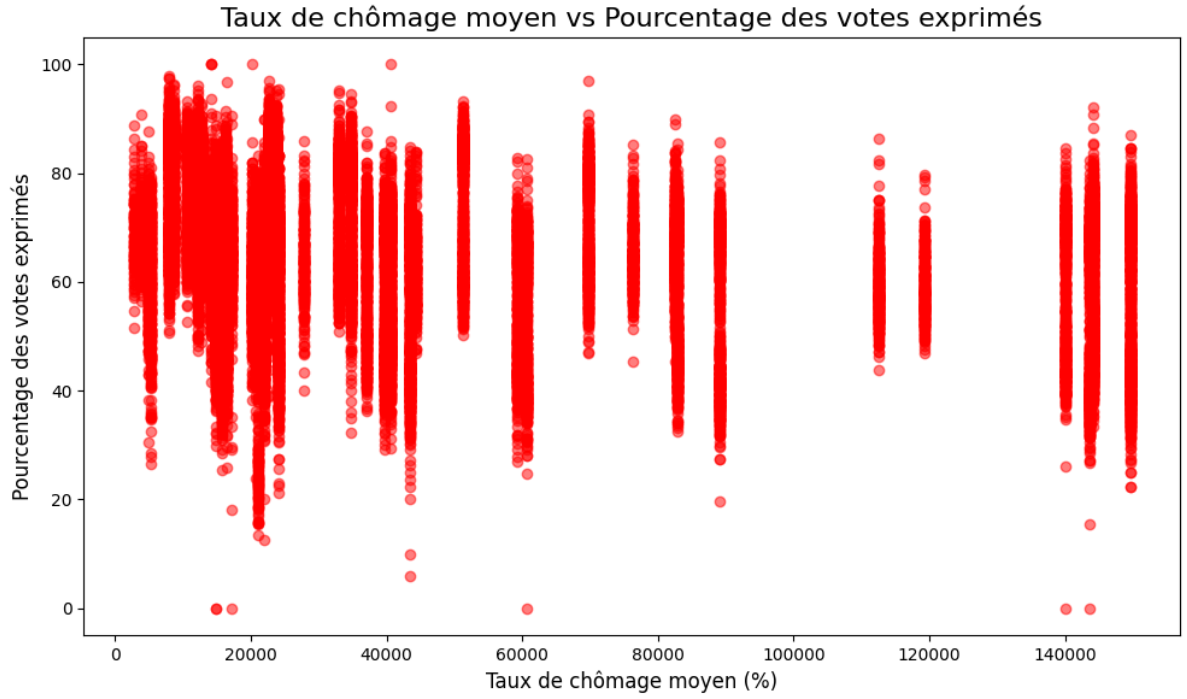
- Synthèse : Ce graphique montre que la plupart des communes ont un taux de participation modéré à élevé, ce qui peut être indicatif d'un engagement électoral stable dans la majorité des régions.



- Synthèse : Ce graphique met en lumière une tendance croissante à l'abstention sur une période de 14 ans, avec un pic en 2017, suggérant un désengagement politique accru au fil du temps.

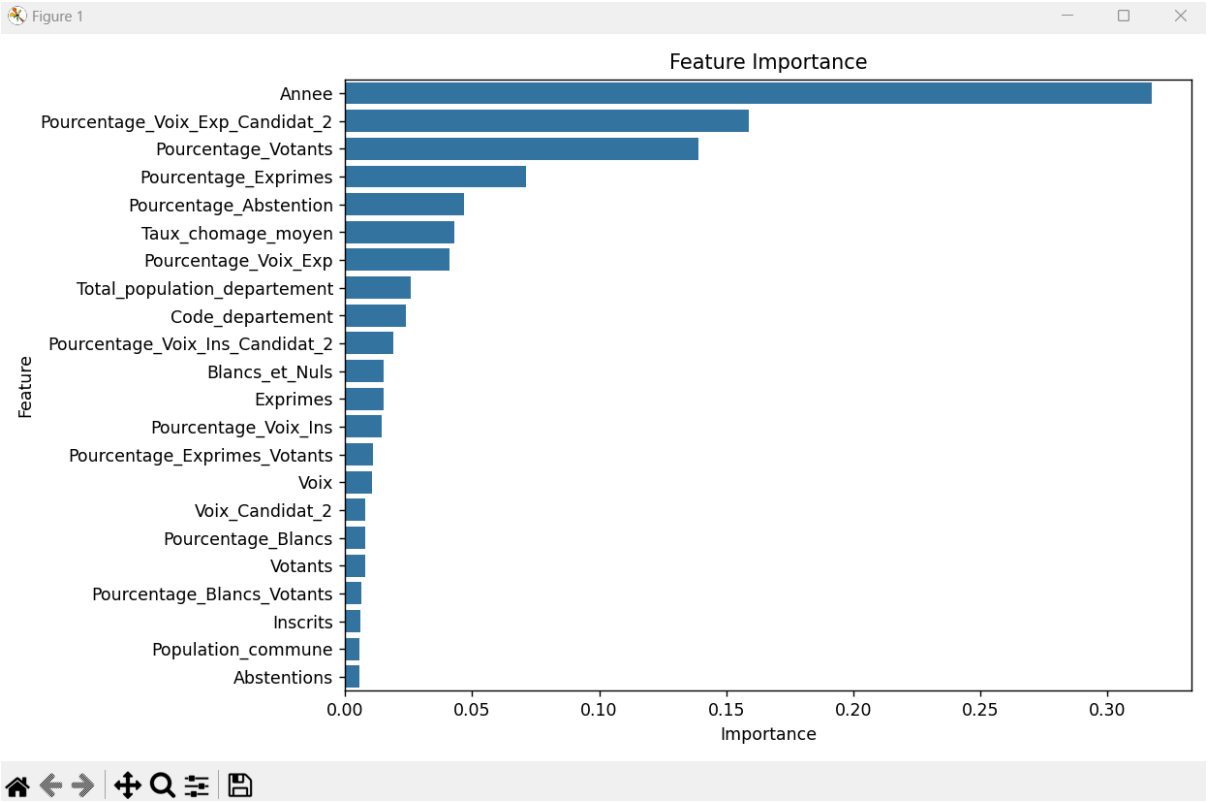


- Synthèse : La plupart des communes avec une petite population ont un nombre d'inscrits proche de la population totale, alors que pour les plus grandes communes, la proportion d'inscrits semble varier de façon plus importante.



- Synthèse : L'analyse suggère que le taux de chômage ne semble pas avoir un impact direct sur le pourcentage des votes exprimés.

Essaie avec XGBoost pour la Feature Importance



Source pour la récolte de nos données :

Elections présidentielle et législatives :
https://www.data.gouv.fr/fr/pages/donnees-des-elections/

Chômages :
https://statistiques.pole-emploi.org/stmt/defm?ff=E&fj=07,09,11,12,30,32,31,65,34,46,48,66,81,82&pp=202101-202112&ss=1

Démographie :

https://www.insee.fr/fr/information/3544265/