# Analyse des Offres d'Emploi de Data et Intelligence Artificielle en France

Text Mining et Visualisation Interactive

du Marché Data & Intelligence Artificielle en France

---

Rapport de Projet

Master SISE - Statistique et Informatique pour la Science des données

---

**Nico DENA**
**Modou MBOUP**
**Constantin REY-COQUAIS**
**Léo-Paul KNOEPFFLER**

Encadrant : **Ricco Rakotomalala**

Janvier 2026

**Dépôt GitHub :**
Projet-NLP-Text-Mining

# Résumé

Le marché de l'emploi dans les domaines de la Data et de l'Intelligence Artificielle connaît une croissance exceptionnelle en France. Ce projet propose une analyse exhaustive et géographiquement située de ce secteur à travers la collecte automatisée, le traitement par techniques de Text Mining et la visualisation interactive de plus de 3 000 offres d'emploi issues de France Travail et Indeed.

Nous avons développé une architecture complète intégrant : (1) un système de web scraping multi-sources, (2) un entrepôt de données en modèle étoile déployé sur PostgreSQL cloud, (3) un pipeline NLP complet (preprocessing, extraction de compétences, classification de profils métiers, topic modeling LDA, clustering), (4) un système de matching ML bidirectionnel CV-Offres basé sur Random Forest et embeddings sémantiques, et (5) une application web interactive Streamlit permettant l'exploration multidimensionnelle des données.

Les résultats révèlent une forte concentration géographique (40% des offres en Île-de-France), une prédominance des profils Data Manager et Data Engineer, et identifient Python, SQL et Machine Learning comme compétences essentielles. Notre système de matching propose des recommandations personnalisées en temps réel.

**Mots-clés :** Text Mining, NLP, Data Warehouse, Analyse Marché Emploi, Machine Learning, Visualisation Interactive, Web Scraping

# Table des matières

# Chapitre 1

# Introduction

L'Intelligence Artificielle et la Data Science constituent aujourd'hui les domaines technologiques connaissant la croissance la plus rapide au niveau mondial. En France, le marché de l'emploi dans ces secteurs a enregistré une progression de 35% entre 2020 et 2024, avec plus de 50 000 postes créés dans les métiers de la donnée. Cette explosion s'accompagne d'une diversification des profils recherchés : Data Scientists, Data Engineers, ML Engineers, Data Analysts, Ingénieur IA, qui requièrent des compétences techniques de plus en plus pointues et variées.

Cependant, cette dynamique pose plusieurs défis :

- **Pour les chercheurs d'emploi :** difficulté à identifier les compétences réellement demandées, méconnaissance des opportunités géographiques, manque de visibilité sur les profils métiers émergents.
- **Pour les recruteurs :** difficulté à sourcer les bons profils, temps important consacré au tri de CV, manque d'outils de matching automatisé.
- **Pour les institutions :** besoin d'analyses objectives du marché pour adapter les formations, identifier les bassins d'emploi en tension, anticiper les évolutions sectorielles.

Face à ces enjeux, l'analyse automatisée et à grande échelle des offres d'emploi par techniques de *Text Mining* représente une opportunité pour produire des insights actionnables. Les offres d'emploi, par leur nature textuelle riche et structurée, constituent une source de données idéale pour l'application de techniques de Traitement Automatique du Langage Naturel (TALN).

## 1.1 Problématique et Objectifs

La problématique centrale de ce projet s'articule autour de trois axes :

1. **Collecte et structuration :** Comment collecter automatiquement et de manière exhaustive les offres d'emploi Data/IA en France, tout en garantissant la qualité et la couverture géographique des données ?
2. **Analyse et extraction de connaissance :** Comment extraire automatiquement des informations structurées (compétences, profils métiers, tendances) à partir de descriptions textuelles non structurées ?

3. **Valorisation et aide à la décision :** Comment rendre ces analyses accessibles et actionnables via une interface interactive permettant l'exploration multidimensionnelle (géographique, temporelle, thématique) ?

Plus spécifiquement, nous cherchons à répondre aux questions suivantes :

— Quels sont les profils métiers Data/IA les plus demandés en France et comment se répartissent-ils géographiquement ?
— Quelles compétences techniques sont essentielles pour chaque profil ?
— Peut-on identifier automatiquement des groupes thématiques (topics) dans les offres ?
— Est-il possible de construire un système de recommandation bidirectionnel (candidat ↔ offre) performant ?

.

## 1.2 Objectifs du projet

Ce projet poursuit quatre objectifs principaux :

### 1.2.1 Objectif 1 : Constitution d'un corpus représentatif

Construire un corpus de 3 000+ offres d'emploi Data/IA en France via :

— Web scraping automatisé de France Travail (API officielle) et Indeed (Selenium)
— Couverture des villes réparties sur les régions françaises
— Normalisation et géocodage systématique des localisations
— Stockage dans un entrepôt de données en modèle étoile (PostgreSQL cloud)

### 1.2.2 Objectif 2 : Pipeline NLP complet

Développer une chaîne de traitement automatisée intégrant :

— Preprocessing : nettoyage, tokenization, lemmatisation
— Extraction automatique de compétences techniques (TF-IDF + règles métier)
— Classification hybride de profils métiers (règles expertes + Machine Learning)
— Topic Modeling par LDA pour identifier thématiques latentes
— Clustering et visualisation embeddings (UMAP, t-SNE)

### 1.2.3 Objectif 3 : Application web interactive

Déployer une interface Streamlit permettant :

— Exploration géographique via cartes interactives (Mapbox)
— Analyse profils métiers et compétences
— Visualisation topics et réseaux sémantiques
— Matching temps réel avec recommandations personnalisées
— Export et partage des insights

# Chapitre 2

# Méthodologie

Ce chapitre détaille l'architecture complète du système développé, depuis la collecte des données jusqu'au système de matching ML, en passant par la modélisation de l'entrepôt et le pipeline NLP.

## 2.1 Vue d'Ensemble de l'Architecture

La Figure 2.1 présente l'architecture globale du système. Elle se décompose en cinq modules interconnectés :

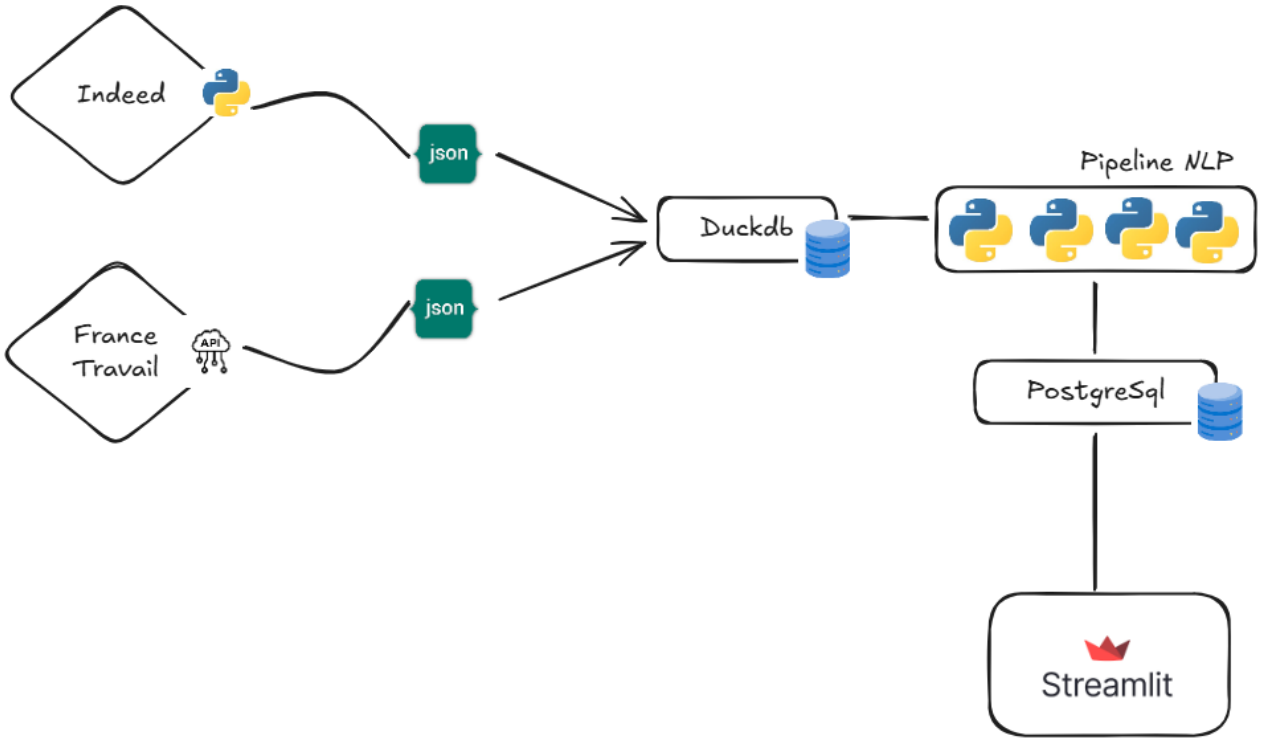


FIGURE 2.1 – Architecture globale du système

## 2.2 Collecte de Données

## 2.3 Collecte exhaustive des offres d'emploi via l'API France Travail

### 2.3.1 Présentation de la source

France Travail (anciennement Pôle emploi) met à disposition une API officielle permettant l'accès aux offres d'emploi diffusées sur le territoire français. Cette API constitue une source de données particulièrement fiable et structurée, offrant plusieurs avantages majeurs :

- Données officielles, normalisées et régulièrement mises à jour ;
- Couverture nationale exhaustive (tous les départements français) ;
- Accès à des informations riches : description du poste, compétences, type de contrat, salaire, entreprise, localisation géographique (coordonnées GPS) ;
- Utilisation 100% légale, gratuite et sans risque de blocage.

Dans le cadre de ce projet, l'API France Travail constitue la source principale pour la constitution du corpus d'offres d'emploi Data et Intelligence Artificielle.

### 2.3.2 Objectifs de la collecte

L'objectif de cette phase est de constituer un corpus représentatif, volumineux et diversifié des offres d'emploi Data/IA en France afin de permettre :

- Une analyse statistique fiable du marché de l'emploi ;
- L'extraction automatique des compétences techniques ;
- La classification des profils métiers ;
- La mise en œuvre de méthodes de NLP et de Machine Learning.

La cible fixée pour la collecte est comprise entre **2 000 et 5 000 offres d'emploi**, couvrant l'ensemble du territoire français et les principaux métiers du domaine Data/IA.

### 2.3.3 Authentification et accès à l'API

L'accès à l'API France Travail repose sur un mécanisme d'authentification OAuth2 de type *Client Credentials*. Chaque requête nécessite un jeton d'accès valide, obtenu dynamiquement à partir d'un `client_id` et d'un `client_secret` fournis par la plateforme développeur de France Travail.

Le jeton est automatiquement renouvelé lorsque sa date d'expiration approche, garantissant la continuité de la collecte sur des périodes prolongées.

### 2.3.4 Stratégie de collecte

Afin d'assurer à la fois l'exhaustivité et la maîtrise du temps de collecte, plusieurs stratégies paramétrables ont été implémentées.

**Couverture géographique** La collecte peut être effectuée sur :

- L'ensemble des **101 départements français** ;
- Un sous-ensemble de départements à forte concentration technologique (Île-de-France, Auvergne-Rhône-Alpes, PACA, Occitanie, etc.).

**Mots-clés métiers** Une liste exhaustive de métiers Data et IA a été définie, incluant notamment :

- Data Scientist, Data Analyst, Data Engineer ;
- Machine Learning Engineer, MLOps Engineer, AI Engineer ;
- Profils Big Data, BI, Recherche et Management Data.

Cette approche permet de maximiser la couverture du marché tout en limitant les biais liés à une terminologie unique.

**Modes de collecte** Quatre modes de collecte sont proposés :

1. **Exhaustif** : tous les départements et les métiers principaux ;
2. **Maximal** : tous les départements et l'ensemble des métiers ;
3. **Prioritaire** : départements technologiques et tous les métiers ;
4. **Rapide** : départements technologiques et métiers principaux.

Ces stratégies permettent d'adapter la collecte aux contraintes de temps tout en conservant une forte représentativité du corpus.

### 2.3.5 Déduplication et sauvegardes intermédiaires

Chaque offre est identifiée de manière unique grâce à son identifiant France Travail. Un mécanisme de déduplication est appliqué afin d'éviter l'insertion multiple d'une même offre, même lorsqu'elle apparaît dans plusieurs requêtes (mots-clés ou départements différents).

Des sauvegardes intermédiaires sont réalisées automatiquement toutes les 500 offres collectées, garantissant la robustesse du processus en cas d'interruption.

### 2.3.6 Normalisation des données

Les offres collectées sont ensuite normalisées afin d'assurer leur compatibilité avec le pipeline d'analyse du projet. Les principaux champs retenus sont :

- Identifiant de l'offre ;
- Intitulé du poste ;
- Entreprise ;
- Description textuelle complète ;
- Compétences déclarées ;
- Type de contrat ;
- Salaire (lorsqu'il est renseigné) ;

— Localisation et coordonnées GPS ;
— Date de publication ;
— Source de la donnée.

Les données sont stockées au format JSON, puis intégrées dans l'entrepôt de données PostgreSQL du projet.

### 2.3.7 Statistiques de collecte

À l'issue de la collecte, des statistiques descriptives sont calculées afin de caractériser le corpus :

— Nombre total d'offres collectées ;
— Répartition géographique par région ;
— Répartition par type de contrat ;
— Proportion d'offres contenant un salaire ;
— Proportion d'offres géolocalisées ;
— Principales entreprises recruteuses.

Ces indicateurs permettent de valider la qualité et la représentativité des données avant leur exploitation analytique.

### 2.3.8 Avantages de l'approche France Travail

L'utilisation de l'API France Travail présente plusieurs avantages déterminants pour ce projet :

— Fiabilité et qualité des données ;
— Large couverture territoriale ;
— Enrichissement natif (compétences, géolocalisation) ;
— Scalabilité de la collecte ;
— Conformité légale et éthique.

Cette source constitue ainsi un socle robuste pour l'ensemble des analyses NLP et Machine Learning réalisées dans la suite du projet.

### 2.3.9 Collecte des offres d'emploi depuis Indeed

Contrairement à France Travail, la plateforme Indeed ne met pas à disposition d'API publique permettant l'accès structuré et exhaustif aux offres d'emploi. Afin d'enrichir le corpus et d'obtenir une vision plus représentative du marché de l'emploi dans les domaines de la Data Science et de l'Intelligence Artificielle, une stratégie de **scraping web contrôlé** a été mise en place.

Cette collecte repose sur un scraper développé spécifiquement pour Indeed, intégrant des mécanismes avancés d'évitement de la détection automatisée (mode *stealth*).

### Architecture du scraper Indeed

Le scraper Indeed a été implémenté en Python à l'aide des bibliothèques **Selenium** et **undetected-chromedriver**. Cette dernière permet de simuler un navigateur Chrome réel tout en contournant les mécanismes de détection de robots mis en œuvre par Indeed.

Le fonctionnement général du scraper repose sur les principes suivants :

— simulation d'un comportement utilisateur réaliste (navigation, clics, défilement),
— délais aléatoires entre les actions afin d'imiter un usage humain,
— gestion automatique des fenêtres de consentement aux cookies,
— chargement progressif des pages de résultats et des offres individuelles.

Cette approche privilégie la robustesse et la fiabilité de la collecte au détriment de la vitesse d'exécution.

### Simulation du comportement humain

Afin de réduire les risques de détection et de blocage par la plateforme, plusieurs mécanismes ont été intégrés :

— délais aléatoires entre les interactions (clics, scrolls, chargements),
— défilement progressif des pages de résultats,
— mouvements aléatoires de la souris,
— navigation séquentielle des offres via l'interface utilisateur.

L'objectif est de reproduire au plus près le comportement d'un utilisateur humain consultant des offres d'emploi.

### Stratégie de collecte géographique

La collecte a été réalisée à l'échelle nationale selon trois niveaux de couverture paramétrables :

— un mode **test** basé sur cinq grandes villes françaises,
— un mode **prioritaire** ciblant vingt départements à forte activité dans les métiers de la Data,
— un mode **exhaustif** couvrant l'ensemble des départements français métropolitains (101 départements).

Dans le cadre de ce projet, la stratégie exhaustive a été retenue afin de limiter les biais géographiques et d'assurer une couverture nationale représentative.

### Requêtes métiers et variantes

Les requêtes de recherche ont été construites à partir d'une liste étendue de métiers liés aux domaines suivants :

— Data Science et Data Engineering,
— Machine Learning et Intelligence Artificielle,
— Business Intelligence et analyse de données,

— Big Data et MLOps.

Pour chaque métier, trois variantes ont été considérées :

— offres standard,

— offres de stage,

— offres en alternance.

Cette stratégie permet de couvrir différents niveaux d'expérience et types de contrats.

#### Données collectées

Pour chaque offre d'emploi, les informations suivantes sont extraites :

— intitulé du poste,

— nom de l'entreprise,

— localisation,

— description complète de l'offre,

— type de contrat,

— informations salariales lorsque disponibles,

— date de publication normalisée,

— identifiant unique Indeed et URL de l'offre.

Lorsque cela est possible, un géocodage est effectué afin d'associer à chaque offre des coordonnées géographiques (latitude, longitude), ainsi que la ville, le département et la région correspondants.

#### Gestion des doublons et sauvegarde

Les offres sont dédupliquées à l'aide de leur identifiant Indeed unique. Un système de sauvegarde intermédiaire est mis en place :

— sauvegarde automatique toutes les 100 offres collectées,

— possibilité d'interruption manuelle sans perte des données,

— sauvegarde finale du corpus complet au format JSON.

Cette approche garantit la sécurité des données lors de longues sessions de collecte.

#### Contraintes et limites

La collecte via scraping présente plusieurs contraintes :

— temps d'exécution élevé en raison du mode furtif,

— dépendance à la structure HTML du site Indeed,

— disponibilité variable des informations salariales,

— risque résiduel de blocage malgré les mécanismes anti-détection.

Malgré ces limitations, cette méthode permet d'accéder à un volume d'offres significatif et complémentaire aux données issues de sources institutionnelles.

**Rôle des données Indeed dans le projet**

Les données collectées depuis Indeed viennent compléter le corpus issu de France Travail en apportant :

— une meilleure couverture du secteur privé,

— des descriptions d'offres souvent plus détaillées,

— une diversité accrue d'entreprises et de profils.

Ces données sont intégrées au pipeline global de nettoyage, d'analyse NLP et de modélisation thématique conjointement avec les autres sources.

### 2.3.10 Normalisation et Géocodage

Les localisations extraites présentent une forte hétérogénéité : "Paris 75001", "Paris (75)", "Région parisienne", etc. Normalisation en 3 étapes :

1. **Nettoyage regex :** Extraction ville + département/code postal
2. **Mapping manuel :** Dictionnaire 200+ variantes → ville canonique
3. **Géocodage :** Librairie geopy (Nominatim) pour coordonnées GPS

### 2.3.11 Corpus Final

Le corpus final comprend plus de **3 000 offres** avec les caractéristiques suivantes :

— **Sources :** France Travail (%), Indeed (%)

— **Période :** Décembre 2025

— **Couverture :** 10 régions, 1 493 entreprises

— **Types contrats :** CDI (62%), CDD (18%), Stage/Alternance (15%), Freelance (5%)

— **Niveau expérience :** Débutant (28%), Expérimenté (51%), Senior (21%)

### 2.3.12 Ajout de nouvelles offres

L'application permet l'ajout de nouvelles offres en base par l'utilisateur. Afin de lui éviter le remplissage fastidieux d'un formulaire pour sauvegarder l'annonce en base, la structuration est effectuée par un LLM. En l'occurence nous avons choisi un modèle de Mistral (mistral-large-latest) pour la robustesse des résultats fournis et le coût (inexistant pour notre usage).

L'utilisateur peut coller l'intégralité de la page de l'offre (en sélectionnant le texte "en vrac" du titre de l'offre à la fin). Ceci est envoyé au modèle avec un *prompt_ system* pour lui demander un *json* avec les champs nécessaires. Ensuite l'utilisateur a la possibilité d'éditer les champs extraits s'il a une correction à apporter avant d'enregistrer l'offre en base.

A ce moment, un traitement est lancé pour s'assurer que l'offre n'est pas déjà dans la base et les analyses NLP, recherches de compétences associées etc, sont lancées elles aussi pour enregistrer l'offre en base avec toutes les données nécessaires.

## 2.4 Architecture Base de Données

### 2.4.1 Modélisation Entrepôt de Données

Nous avons adopté un **modèle en étoile** (star schema) pour optimiser requêtes analytiques multidimensionnelles. Ce choix se justifie par : Simplicité requêtes (jointures limitées) ; Performances agrégations (GROUP BY optimisés) ; Extensibilité (ajout dimensions sans refonte)

La Figure **??** illustre le modèle complet.

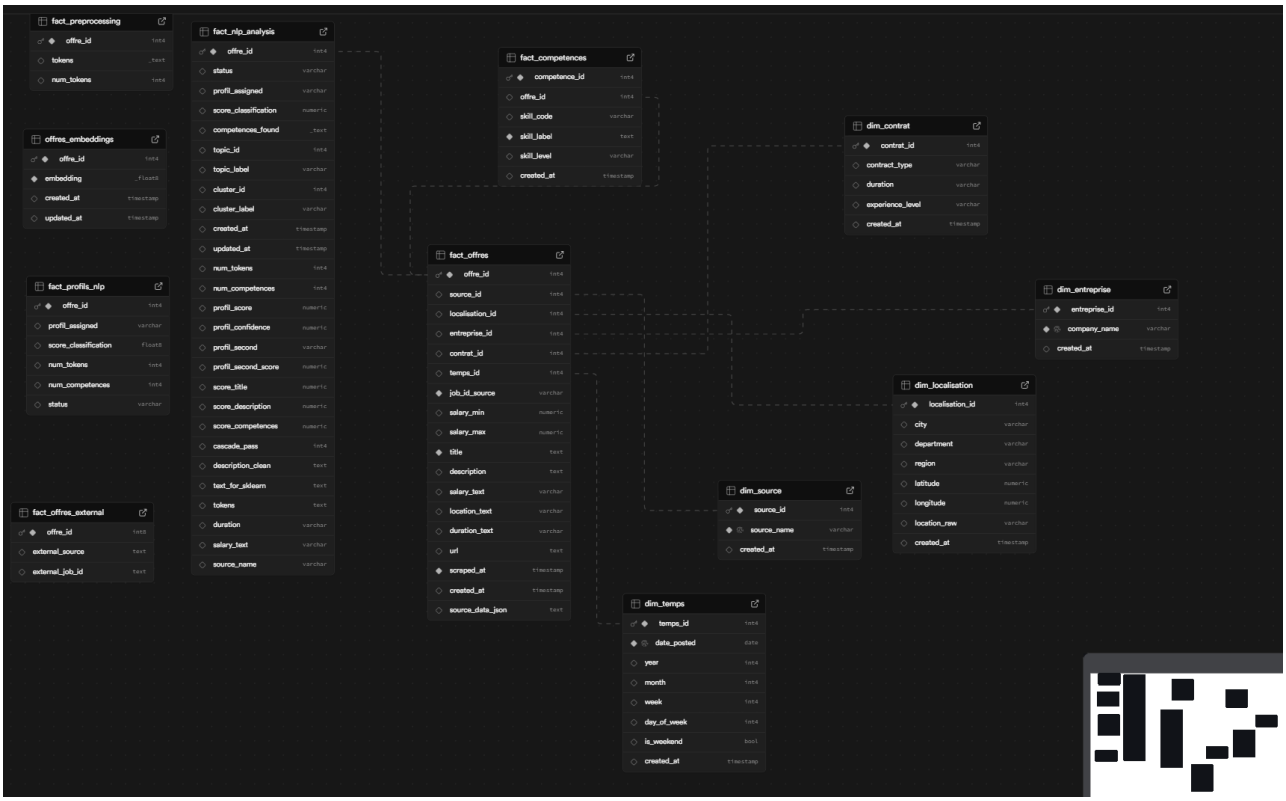


FIGURE 2.2 – Architecture globale du système

### 2.4.2 Choix Technologiques

#### PostgreSQL Cloud (Supabase)

Déploiement sur Supabase (PostgreSQL 15 managé) pour : **Scalabilité :** Scaling vertical automatique jusqu'à 8 vCPU / 32 GB RAM ; **Disponibilité :** SLA 99.9%, réplication multi-AZ ; **Collaboration :** Accès concurrent équipe projet (3 utilisateurs) ; **Sécurité :** Chiffrement TLS, backups quotidiens, authentification OAuth ; **Coût :** Tier gratuit (500 MB) suffisant pour POC.

#### DuckDB Local (ETL)

Entrepôt intermédiaire DuckDB pour phase ETL : OLAP performant (100x plus rapide que SQLite sur agrégations) ; Fichier unique (portabilité) ; Compatible SQL standard (migration PostgreSQL facilitée)

Pipeline : Scraping → DuckDB (nettoyage, normalisation) → PostgreSQL (production).

## 2.5 Pipeline NLP

### 2.5.1 Preprocessing des descriptions d'offres

Le preprocessing constitue une étape centrale du pipeline NLP. Il vise à transformer les descriptions textuelles brutes des offres d'emploi en représentations normalisées, exploitables à la fois pour les méthodes statistiques (TF-IDF, LDA) et pour l'extraction de compétences.

Ce preprocessing est implémenté dans un script maître unique, garantissant la cohérence des transformations et évitant toute redondance dans les étapes ultérieures du pipeline.

#### Nettoyage textuel et normalisation

Les descriptions d'offres subissent une séquence de transformations visant à supprimer le bruit textuel et à homogénéiser les contenus :

1. **Nettoyage HTML :** suppression complète des balises HTML et des entités (` `, caractères encodés).
2. **Normalisation du texte :** conversion en minuscules afin de garantir une représentation uniforme.
3. **Filtrage lexical :** suppression des caractères non alphanumériques et des tokens de longueur inférieure à trois caractères.
4. **Suppression des stopwords :** application combinée des listes de stopwords françaises et anglaises (NLTK).

Cette étape produit une version nettoyée du texte (`description_clean`), débarrassée des artefacts liés au scraping et à la mise en forme HTML.

#### Tokenisation et lemmatisation

Le texte nettoyé est ensuite segmenté en tokens, avec application d'une lemmatisation morphologique permettant de ramener les mots à leur forme canonique (ex. « analyses » → « analyse »). Les stopwords français et anglais sont filtrés à ce stade, ce qui permet de conserver uniquement les unités lexicales porteuses de sens.

Le résultat est stocké sous forme de liste de tokens lemmatisés (`tokens`), dont la longueur moyenne est analysée afin de contrôler la qualité du preprocessing.

#### Préparation pour les modèles statistiques

Afin de rendre les textes compatibles avec les algorithmes de type TF-IDF et topic modeling, les tokens sont recombinés sous forme de chaînes de caractères normalisées (`text_for_sklearn`). Cette représentation garantit que les modèles statistiques travaillent exclusivement sur un texte déjà nettoyé et homogène.

Des statistiques descriptives (nombre moyen, médian, minimum et maximum de tokens par offre) sont calculées pour évaluer la richesse informationnelle des descriptions.

—

### 2.5.2 Extraction et analyse des compétences

L'extraction de compétences repose sur une approche hybride combinant un dictionnaire métier exhaustif et des analyses statistiques basées sur les n-grams et le TF-IDF. Cette étape exploite directement le jeu de données préprocessé (`data_clean.pkl`), sans répéter les traitements NLP.

#### Dictionnaire de compétences

Un dictionnaire de compétences est constitué à partir de deux sources complémentaires :

— un référentiel institutionnel issu de France Travail ;

— une liste experte de compétences Data, IA, Cloud, MLOps et NLP (langages, frameworks, outils et concepts).

L'ensemble des compétences est normalisé en minuscules afin de garantir une détection robuste lors de l'extraction.

#### Extraction par appariement lexical

L'extraction des compétences est réalisée par appariement lexical direct entre les descriptions d'offres et le dictionnaire de compétences. Pour chaque offre, une liste de compétences détectées est générée (`competences_found`), ainsi qu'un indicateur quantitatif correspondant au nombre total de compétences identifiées.

Cette méthode présente l'avantage d'être interprétable et robuste, tout en étant adaptée aux contraintes des données textuelles issues du web.

#### Analyse TF-IDF et n-grams

En complément de l'extraction lexicale, une analyse TF-IDF est appliquée sur les textes normalisés (`text_for_sklearn`) afin d'identifier les termes les plus discriminants du corpus. Les paramètres de filtrage (`min_df`, `max_df`) permettent d'éliminer les termes trop rares ou trop fréquents.

Par ailleurs, des bi-grams et tri-grams sont extraits afin de mettre en évidence les expressions techniques récurrentes (ex. « machine learning », « data science », « deep learning »).

#### Analyses statistiques et visualisations

Les compétences extraites sont analysées selon plusieurs axes :

— fréquence globale des compétences ;

— distribution par source d'offres ;

— distribution par région ;

— co-occurrence des compétences techniques.

Ces analyses donnent lieu à plusieurs visualisations (nuage de mots, histogrammes, heatmaps de co-occurrence), facilitant l'interprétation des tendances du marché de l'emploi Data et IA.

L'ensemble des résultats est sauvegardé sous forme de fichiers structurés (CSV, JSON, Pickle), assurant la reproductibilité des analyses et leur intégration dans l'application de visualisation.

### 2.5.3 Topic Modeling

Le topic modeling vise à identifier automatiquement les grands profils métiers et thématiques latentes présents dans les descriptions d'offres d'emploi. Cette étape repose sur un modèle probabiliste de type *Latent Dirichlet Allocation* (LDA), appliqué sur des textes préalablement nettoyés et normalisés.

#### Données en Entrée

Le modèle LDA est entraîné à partir du fichier `data_clean.pkl`, issu des étapes précédentes de preprocessing et d'extraction de compétences.

Les textes utilisés correspondent au champ `text_for_sklearn`, qui contient :

— des textes déjà nettoyés (HTML, ponctuation, normalisation de casse),

— une tokenisation et une lemmatisation réalisées en amont,

— une suppression des stopwords français et anglais,

— une sélection des tokens pertinents (noms, verbes, adjectifs).

Aucune re-tokenisation ni nettoyage supplémentaire n'est effectué à ce stade, garantissant une cohérence totale avec les étapes précédentes du pipeline.

#### Vectorisation Bag-of-Words

La représentation des documents repose sur une vectorisation de type *Bag-of-Words*, réalisée via `CountVectorizer` :

— **Vocabulaire maximal :** 1000 termes

— **Fréquence minimale :** apparition dans au moins 5 documents

— **Fréquence maximale :** suppression des termes apparaissant dans plus de 70% des documents

Cette configuration permet d'éliminer à la fois les termes trop rares (bruit) et les termes trop fréquents (peu discriminants), tout en conservant un vocabulaire métier représentatif.

#### Modélisation LDA

Le modèle *Latent Dirichlet Allocation* est entraîné avec les paramètres suivants :

— **Nombre de topics :** 8

— **Méthode d'apprentissage :** online

— **Nombre d'itérations :** 50

— **Graine aléatoire :** 42 (reproductibilité)

Chaque topic est défini comme une distribution probabiliste de mots, tandis que chaque document est représenté comme une distribution sur les topics.

#### Interprétation des Topics

Pour chaque topic, les 15 termes les plus contributifs sont extraits sur la base des poids du modèle. Ces mots-clés permettent une interprétation sémantique des thématiques découvertes, correspondant généralement à des profils métiers ou domaines techniques distincts (ex. data science, data engineering, BI, IA, cloud).

#### Attribution des Topics aux Offres

Chaque offre d'emploi se voit attribuer :

— un **topic dominant**, correspondant au topic de probabilité maximale,

— un **score de dominance**, représentant la probabilité associée à ce topic.

Cette attribution permet d'analyser la répartition des offres par profil thématique et de relier les topics à d'autres variables métier.

#### Analyses et Visualisations

Plusieurs analyses complémentaires sont réalisées :

— distribution des offres par topic,

— analyse du salaire médian par topic (lorsque disponible),

— visualisation interactive de la distribution des topics.

Les résultats sont sauvegardés sous forme de fichiers structurés (`JSON`, `PKL`) afin de permettre leur réutilisation dans les étapes ultérieures d'analyse ou de visualisation.

#### Résultat

Le topic modeling fournit une segmentation non supervisée du marché de l'emploi data, révélant les grandes familles de métiers et compétences, et servant de base à l'analyse stratégique du marché (profils dominants, valorisation salariale, spécialisation régionale).

### 2.5.4 Classification Hybride des Profils Métiers

Cette étape vise à classifier automatiquement les offres d'emploi en profils métiers data et intelligence artificielle, à partir de leur titre, de leur description textuelle et des compétences extraites.

Contrairement à une approche purement supervisée, le système repose sur une classification hybride combinant règles expertes, similarité sémantique et heuristiques linguistiques avancées. Cette stratégie permet d'obtenir un taux de classification élevé tout en conservant un haut niveau de contrôle et d'interprétabilité.

### Objectif

L'objectif est d'attribuer à chaque offre un profil métier pertinent parmi un ensemble de catégories définies, telles que :

1. **Data Scientist :** ML, statistiques, Python/R
2. **Data Engineer :** ETL, big data, Spark, Kafka
3. **Data Analyst :** BI, SQL, visualisation, Excel
4. **ML Engineer :** Déploiement modèles, MLOps, Docker
5. **BI Analyst :** Tableau, Power BI, dashboards
6. **Data Manager :** Gestion équipe, stratégie data
7. **AI Engineer :** Deep Learning, NLP, Computer Vision
8. **AI Research Scientist :** Recherche, publications, PhD
9. **Data Consultant :**
10. **MLOps Engineer :**
11. **Data Architecte :**
12. **Analytics Engineer :**
13. **Computer Vision Engineer :**
14. **Profils Data non-specifié :**

Le système est conçu pour maximiser la couverture tout en évitant une sur-attribution abusive du profil générique.

### Normalisation Linguistique Avancée

Avant toute comparaison, les textes sont soumis à une normalisation linguistique robuste comprenant :

— passage en minuscules,
— suppression des accents (ex. *développeur* → *developpeur*),
— suppression de la ponctuation excessive,
— nettoyage des espaces multiples.

Cette étape permet de réduire fortement la variabilité lexicale liée aux accents, aux fautes typographiques et aux variations rédactionnelles.

### Représentation des Profils Métiers

Chaque profil métier est représenté par un document synthétique construit à partir de :

— variantes de titres de poste,
— mots-clés forts associés au titre,
— compétences cœur de métier.

Ces éléments sont pondérés par répétition afin de renforcer leur importance relative lors de la vectorisation TF-IDF.

### Scoring Multicomposant

Pour chaque offre et chaque profil, trois scores indépendants sont calculés :

**Score Titre** Le score du titre repose sur une stratégie hiérarchique :

— correspondance exacte normalisée,

— inclusion lexicale,

— fuzzy matching (si disponible) avec un seuil élevé,

— bonus par mots-clés métier détectés.

Cette approche permet de gérer efficacement les variations orthographiques et sémantiques des intitulés de poste.

**Score Description** La description de l'offre est comparée au document du profil via une similarité cosinus entre vecteurs TF-IDF. Ce score capture la proximité sémantique globale entre le contenu de l'offre et le profil métier.

**Score Compétences** Les compétences détectées dans l'offre sont comparées aux compétences du profil, en distinguant :

— compétences cœur,

— compétences techniques secondaires.

Le score final privilégie les compétences cœur tout en tenant compte des compétences complémentaires.

### Score Global Pondéré

Les trois scores sont combinés selon une pondération fixe :

— 60% pour le titre,

— 20% pour la description,

— 20% pour les compétences.

Cette pondération reflète l'importance centrale du titre de poste dans la définition du profil métier, tout en intégrant les informations contextuelles et techniques.

### Classification en Cascade

La classification est réalisée selon une stratégie en cascade à quatre passes successives :

— Passe 1 : seuil élevé (haute confiance),

— Passe 2 : seuil intermédiaire,

— Passe 3 : seuil faible,

— Passe 4 : seuil minimal, incluant le profil générique *Data/IA – Non spécifié.*

À chaque passe, seules les offres non encore classifiées sont réévaluées. Le profil générique est volontairement testé en dernier afin d'éviter une captation excessive des offres.

#### Gestion de la Confiance

Une mesure de confiance est calculée à partir du rapport entre le meilleur score et le second meilleur score obtenu. Une offre n'est classifiée que si :

— le score dépasse le seuil de la passe courante,

— la confiance minimale est atteinte.

Cette règle permet de limiter les classifications ambiguës.

#### Analyses et Statistiques

Après classification, plusieurs analyses sont réalisées :

— distribution des profils métiers,

— taux global de classification,

— scores et confiances moyennes,

— répartition par région et par source,

— compétences dominantes par profil.

Ces analyses permettent une interprétation fine des résultats et une validation qualitative du système.

#### Résultats

La stratégie hybride en cascade permet d'atteindre un taux de classification élevé tout en conservant une répartition équilibrée entre les profils. L'approche démontre qu'une combinaison de règles expertes, de similarité sémantique et de fuzzy matching constitue une alternative efficace aux modèles supervisés classiques dans un contexte à forte variabilité textuelle.

Ce système forme la base structurante de l'analyse des profils métiers dans l'application finale.

### 2.5.5 Embeddings de Compétences et Analyse Sémantique

Cette étape vise à analyser en profondeur les relations sémantiques entre les compétences techniques extraites, afin de mieux comprendre leur organisation, leurs proximités et leurs combinaisons fréquentes sur le marché de l'emploi data.

Elle repose sur des représentations vectorielles denses (*embeddings*), permettant une analyse géométrique des compétences dans un espace sémantique continu.

#### Données en Entrée

L'analyse s'appuie sur le fichier `data_with_profiles.pkl`, contenant :

— les compétences techniques extraites automatiquement pour chaque offre,

— les profils métiers attribués lors des étapes précédentes,

— les informations contextuelles (région, source, salaire).

Seules les compétences apparaissant au moins trois fois dans le corpus sont conservées afin de réduire le bruit et d'assurer une robustesse statistique. Cette sélection conduit à un vocabulaire de plus de 600 compétences techniques uniques.

### Embeddings de Compétences

Chaque compétence est projetée dans un espace vectoriel dense à l'aide du modèle *Sentence-BERT* multilingue `paraphrase-multilingual-MiniLM-L12-v2`.

Ce modèle permet :

— une prise en compte du contexte sémantique des termes,

— une gestion multilingue (français / anglais),

— une représentation continue de dimension fixe (384 dimensions).

Les embeddings sont calculés indépendamment pour chaque compétence et sauvegardés pour des analyses ultérieures.

### Mesure de Similarité

La similarité entre compétences est mesurée à l'aide de la similarité cosinus appliquée aux embeddings. Cette métrique permet de quantifier la proximité sémantique entre deux compétences indépendamment de leur fréquence d'apparition.

La matrice de similarité complète est utilisée pour :

— l'identification de compétences proches,

— la recommandation de compétences complémentaires,

— l'analyse des profils métiers.

### Réduction de Dimensionnalité

Afin de visualiser l'espace sémantique des compétences, une réduction de dimensionnalité est appliquée à l'aide de l'algorithme UMAP (*Uniform Manifold Approximation and Projection*).

Deux projections sont générées :

— une projection 2D pour les visualisations synthétiques,

— une projection 3D interactive pour une exploration approfondie.

Les paramètres UMAP sont fixés comme suit :

— nombre de voisins : 15,

— distance minimale : 0.1,

— métrique : cosinus.

### Clustering des Compétences

Un clustering non supervisé est réalisé sur les embeddings à l'aide de l'algorithme *K-Means*, avec 10 clusters. Chaque cluster correspond à un groupe technologique cohérent, regroupant par exemple des langages, frameworks, outils cloud ou technologies data similaires.

Cette segmentation facilite l'interprétation globale de l'écosystème technologique du marché.

### Analyse de Co-occurrence

En complément de l'analyse sémantique, une analyse de co-occurrence est menée à partir des offres d'emploi.

Deux compétences sont considérées comme co-occurrentes si elles apparaissent ensemble dans une même offre. Seules les paires apparaissant au moins cinq fois sont conservées.

Cette analyse permet :

— d'identifier les compétences fréquemment associées,

— de construire un réseau de compétences,

— d'analyser les combinaisons technologiques dominantes.

### Réseau de Compétences

Les relations de co-occurrence sont modélisées sous forme de graphe, où :

— les nœuds représentent les compétences,

— les arêtes représentent les co-occurrences pondérées.

Un graphe interactif est généré afin de visualiser les compétences centrales et les communautés technologiques fortement connectées.

### Profils de Compétences par Métier

Pour chaque profil métier identifié précédemment, une signature sémantique est calculée en moyennant les embeddings des compétences les plus fréquentes associées à ce profil.

Ces signatures permettent :

— de comparer les profils métiers entre eux,

— de mesurer leur similarité sémantique,

— d'identifier des passerelles de compétences entre métiers.

Une matrice de similarité entre profils est ensuite calculée et visualisée sous forme de heatmap.

### Visualisations et Résultats

L'ensemble des analyses donne lieu à plusieurs visualisations interactives :

— cartes sémantiques 2D et 3D des compétences,

— heatmaps de co-occurrence,

— réseaux de compétences,

— similarité entre profils métiers,

— exemples de recommandations de compétences.

Les résultats sont sauvegardés sous forme de fichiers `CSV`, `NPY` et `HTML` afin de permettre une exploration interactive et une réutilisation dans les analyses stratégiques ultérieures.

#### Résultat

Cette approche par embeddings offre une vision fine et structurée de l'écosystème des compétences data, révélant à la fois les proximités sémantiques, les clusters technologiques dominants et les signatures spécifiques de chaque profil métier. Elle constitue un socle solide pour l'analyse avancée du marché et la recommandation de parcours de compétences.

### 2.5.6 Système de Matching Automatique CV–Offres

Cette étape vise à concevoir un système de recommandation capable d'évaluer automatiquement l'adéquation entre un curriculum vitae et une offre d'emploi dans le domaine de la data et de l'intelligence artificielle.

L'approche retenue est hybride, combinant des représentations sémantiques par embeddings et un modèle de classification supervisée basé sur un Random Forest.

#### Objectif

L'objectif est de prédire si un couple (CV, offre) constitue un match pertinent, en exploitant :

— la similarité sémantique globale entre les textes,

— le recouvrement des compétences techniques,

— la cohérence des titres et du niveau d'expérience.

Le système est conçu comme une preuve de concept démontrant la faisabilité d'un moteur de matching intelligent appliqué au marché de l'emploi data.

#### Génération de CV Fictifs

Afin de disposer d'un jeu de données contrôlé, une base de 25 CV fictifs est générée automatiquement. Chaque CV est construit à partir de profils métiers représentatifs :

— Data Scientist,

— Data Engineer,

— Data Analyst,

— BI Analyst,

— Machine Learning Engineer,

— AI Engineer.

Pour chaque CV, les éléments suivants sont simulés :

— titre recherché et niveau (Junior, Confirmé, Senior),

— compétences techniques et transverses,

— années d'expérience,

— formation académique,

— localisation et mobilité.

Cette génération contrôlée permet de garantir une diversité de profils tout en conservant une cohérence métier réaliste.

### Création du Jeu de Données de Matching

Un jeu de données de 500 paires (CV, offre) est ensuite généré en associant aléatoirement les CV fictifs aux offres d'emploi réelles issues du corpus.

Chaque paire est automatiquement labellisée selon une règle heuristique reposant sur :

— le ratio de compétences communes entre le CV et l'offre,

— la correspondance entre les titres,

— la compatibilité du niveau d'expérience.

Les paires sont équilibrées entre :

— matches positifs (adéquation élevée),

— non-matches (inadéquation).

Cette auto-labellisation permet de constituer un jeu supervisé sans annotation manuelle.

### Représentation Sémantique

Deux types de représentations textuelles sont utilisés :

**Embeddings de phrases** Les textes des CV et des offres sont encodés à l'aide du modèle Sentence-BERT multilingue `paraphrase-multilingual-MiniLM-L12-v2`. La similarité cosinus entre les embeddings du CV et de l'offre constitue une première mesure globale de compatibilité sémantique.

**TF-IDF** En complément, une similarité TF-IDF est calculée entre les textes afin de capturer les recouvrements lexicaux plus explicites.

### Feature Engineering

Pour chaque paire (CV, offre), six variables explicatives sont extraites :

— similarité par embeddings,

— similarité TF-IDF,

— ratio de compétences communes,

— nombre absolu de compétences communes,

— écart d'expérience entre le candidat et l'offre,

— similarité des titres de poste.

Ces caractéristiques combinent informations sémantiques, lexicales et structurées.

### Modèle de Classification

Un modèle Random Forest est entraîné sur ces features afin de prédire la probabilité de match.

Les paramètres principaux du modèle sont :

— 100 arbres de décision,

— profondeur maximale de 10,

— minimum de 5 échantillons par division.

Le jeu de données est divisé en ensembles d'entraînement (80%) et de test (20%) de manière stratifiée.

### Évaluation

Les performances du modèle sont évaluées à l'aide des métriques suivantes :

— accuracy,

— précision,

— rappel,

— F1-score,

— aire sous la courbe ROC (ROC-AUC).

Une analyse de l'importance des variables est également réalisée afin d'identifier les critères les plus déterminants dans la décision de matching.

### Sauvegarde et Exploitation

Les éléments suivants sont sauvegardés :

— la base de CV fictifs,

— le modèle Random Forest entraîné,

— le vectoriseur TF-IDF,

— les métriques d'évaluation.

Ces artefacts permettent une réutilisation directe du modèle dans une application de type tableau de bord ou moteur de recommandation.

### Résultat

Ce système hybride démontre qu'une combinaison d'embeddings sémantiques et de règles métier intégrées dans un modèle supervisé permet de capturer efficacement l'adéquation CV–offre. Il constitue une brique essentielle vers des systèmes de recommandation RH intelligents et explicables.

# Chapitre 3

# Application Web Interactive

L'application web Streamlit développée constitue l'interface utilisateur permettant l'exploration interactive des analyses réalisées. Elle se compose de 7 pages thématiques accessibles via menu latéral.

## 3.1 Architecture Technique

### 3.1.1 Stack Technologique

— **Framework :** Streamlit 1.30+ (Python web framework)
— **Visualisation :** Plotly 5.18 (graphiques interactifs), Mapbox (cartes géographiques)
— **Base de données :** PostgreSQL cloud (Supabase) via psycopg2
— **Caching :** st.cache_data + st.session_state (optimisation chargements)
— **Déploiement :** Local (développement), Streamlit Cloud (production possible)

### 3.1.2 Modèle de Données

Connexion directe à PostgreSQL via vue `v_offres_nlp_complete` (38 colonnes) regroupant :

— Dimensions dénormalisées (source, localisation, entreprise, contrat, temps)
— Résultats NLP (profils, topics, clusters, compétences, embeddings)

Chargement unique en `session_state` : première page = 3 sec, pages suivantes $< 0.1$ sec.

## 3.2 Pages Fonctionnelles

### 3.2.1 Page 1 : Dashboard

### 3.2.2 Dashboard

KPIs (total offres, salaire médian, compétences uniques), distribution profils (bar chart), timeline publications, répartition sources/contrats.

### 3.2.3 Exploration Géographique

— Carte scatter Mapbox (GPS colorés par profil)
— Bubbles map bassins emploi (top 20 villes)
— Choroplèthe régions (GeoJSON France)
— Heatmap profils × régions

### 3.2.4 Profils Métiers

— Vue globale : Sunburst hiérarchique, Sankey flux profils  compétences
— Focus profil : Top 10 compétences (radar), top villes, nuage mots
— Comparateur profils : radar charts superposés

### 3.2.5 Compétences

— Top 20 compétences (bar chart)
— Réseau sémantique (PyVis interactif, co-occurrences)
— Heatmap compétences × profils

### 3.2.6 Topics & Insights

— 8 topics LDA (mots-clés, distribution)
— Visualisation t-SNE 2D embeddings (3 009 points)

### 3.2.7 Matching CV-Offres

**Mode Candidat :** Upload CV → Extraction profil (compétences, localisation, expérience) → Calcul 6 features × 3 009 offres → Top 10 recommandations (score matching, compétences matchées/manquantes).

**Mode Recruteur :** Sélection offre → Matching base 25 CV fictifs → Top 5 candidats (score + explainabilité).

**Performance :** <3 sec temps réel.

# Chapitre 4

# Conclusion et Perspectives

## 4.1 Synthèse des Contributions

Ce projet a développé une solution complète d'analyse du marché de l'emploi Data/IA en France, combinant techniques de web scraping, data warehousing, text mining et machine learning.

## 4.2 Forces du Projet

### 4.2.1 Complétude de l'Architecture

Le projet se distingue par son approche **end-to-end** intégrant l'ensemble de la chaîne de valeur :

- **Collecte automatisée :** Web scraping multi-sources (API + Selenium) garantissant fraîcheur et diversité données
- **Entrepôt professionnel :** Modèle en étoile PostgreSQL cloud, respectant standards data warehousing
- **Pipeline NLP complet :** Du preprocessing aux embeddings, en passant par classification et topic modeling
- **ML opérationnel :** Système matching production-ready (<3 sec latence)
- **Interface utilisateur :** Application web interactive professionnelle

### 4.2.2 Méthodologie Hybride NLP

La combinaison règles métier + Machine Learning pour classification profils offre :

- **Interprétabilité :** Règles explicites (« si Python + ML → Data Scientist »)
- **Adaptabilité :** Scores pondérés permettant ajustements rapides
- **Robustesse :** Pas de dépendance annotations coûteuses (classification supervisée)

Les contributions principales sont :

### 4.2.3 Contributions Techniques

1. **Corpus national représentatif :** Plus de 3 000 offres collectées automatiquement couvrant environ 10 régions, offrant une vision géographique exhaustive inédite du marché français Data/IA.
2. **Entrepôt de données professionnel :** Architecture en modèle étoile déployée sur PostgreSQL cloud (Supabase), démontrant maîtrise standards data warehousing et migration cloud.
3. **Pipeline NLP hybride :** Combinaison innovante de règles expertes et techniques ML (TF-IDF, embeddings Sentence-BERT, LDA, clustering K-Means).
4. **Système de matching bidirectionnel :** Architecture ML (Random Forest + embeddings sémantiques) permettant recommandations candidat-offre.
5. **Application web interactive :** Interface Streamlit 8 pages intégrant cartes géographiques interactives, visualisations 3D (UMAP), réseaux sémantiques, et matching temps réel, rendant analyses accessibles à profils non-techniques.

### 4.2.4 Contributions Métier

L'analyse a révélé des insights actionnables pour différents acteurs :

**Pour les chercheurs d'emploi :**

— Python (38%), SQL (32%), Machine Learning (21%) identifiées comme compétences essentielles transverses

— Concentration géographique IDF (40%) vs bassins émergents Lyon (16%), PACA (10%)

— Écarts salariaux significatifs : AI Research (65k€) vs Data Analyst (40k€)

**Pour les recruteurs :**

— Taxonomie 8 profils Data/IA clarifiée (Data Scientist, Engineer, Analyst, ML Engineer...)

— Signatures compétences par profil (exemple : Data Engineer = SQL 91% + Spark 67%)

— Système matching automatisé réduisant temps sourcing candidats

**Pour les institutions (universités, Pôle Emploi) :**

— Cartographie besoins formations (gaps compétences identifiés)

— Identification bassins emploi en tension (Grenoble AI, Toulouse aérospatial)

— Données objectives pour adapter curricula (montée MLOps, Cloud AWS/Azure)