Projet MidlJob : Recherche du Meilleur Appariement entre les Candidats et les Recruteurs

NASRO Rona

6 juin 2024

1 Introduction

Le projet MidlJob a pour objectif de trouver le meilleur appariement entre les candidats et les entreprises en utilisant une approche basée sur la similarité de leurs profils respectifs. Pour cela, nous avons développé un système qui encode les caractéristiques des candidats et des offres d'emploi, puis utilise la similarité cosinus pour déterminer les meilleures correspondances.

2 Jeu de Données

Pour ce projet, nous avons utilisé deux fichiers CSV extraits de la collection de Firebase *jobs* :

— **companies_job.csv** : Contient les informations sur les offres d'emploi des entreprises.

— **candidates_job.csv** : Contient les informations sur les candidats à la recherche d'emploi.

Ces jeux de données incluent des champs tels que les compétences techniques et non techniques, les valeurs, le type de contrat, l'expérience, le niveau d'études, et bien d'autres.

3 Méthodologie

La méthodologie suivie dans ce projet comprend plusieurs étapes clés :

1. **Encodage des attributs** : Les caractéristiques des candidats et des offres sont encodées en utilisant des mappings prédéfinis pour les variables catégorielles telles que le type de contrat, l'expérience, le niveau d'études, etc.
2. **Remplissage des valeurs manquantes** : Pour les entrées manquantes, nous avons attribué des valeurs par défaut afin de garantir l'intégrité des données.
3. **Encodage des compétences et des valeurs** : Les compétences non techniques et les valeurs sont encodées sous forme de matrices binaires.
4. **Vectorisation TF-IDF** : Les compétences techniques sont vectorisées en utilisant la méthode TF-IDF (*Term Frequency-Inverse Document Frequency*).
5. **Combinaison des caractéristiques** : Toutes les caractéristiques encodées sont combinées en un seul DataFrame pour les candidats et les offres.
6. **Calcul de la similarité cosinus** : La similarité cosinus entre les caractéristiques des candidats et des offres est calculée pour déterminer les meilleures correspondances.

4 Encodage des Attributs

Les attributs des candidats et des offres sont encodés à l'aide de mappings spécifiques :

— **Statut de télétravail** : Présentiel, Télétravail complet, Hybride, Indifférent
— **Disponibilité** : Immédiate, Moins de 3 mois, Plus de 3 mois
— **Type de contrat** : CDI, CDD, Freelance, Alternance, Stage
— **Temps de travail** : Temps plein, Temps partiel, Mi-temps
— **Expérience** : Sans expérience, 1-2 ans, 3-5 ans, 5-10 ans, Plus de 10 ans
— **Niveau d'études** : Autodidacte, CAP, BEP, Bac, DUT, BTS, Licence, Master, Doctorat
— **Salaire** : Jusqu'à 20 000 €, 20 000 à 30 000 €, 30 000 à 40 000 €, 40 000 à 55 000 €, 55 000 à 70 000 €, Plus de 70 000 €

5 Encodage des Compétences et des Valeurs

Les compétences et les valeurs sont transformées en matrices binaires pour une meilleure manipulation des données.

5.1 Compétences Non Techniques

Les compétences non techniques incluent des attributs tels que :
— Organisation
— Résilience
— Créativité
— Esprit d'équipe
— Gestion du temps
— Motivation

5.2 Valeurs

Les valeurs incluent des attributs tels que :
— Égalité des chances
— Diversité et inclusion
— Droits de l'homme
— Protection de l'environnement
— Justice sociale

6 Vectorisation des Compétences Techniques

Les compétences techniques sont vectorisées en utilisant la méthode TF-IDF, qui permet de convertir des textes en vecteurs de caractéristiques numériques

en tenant compte de l'importance relative des termes.

7 Calcul de la Similarité Cosinus

Le calcul de la similarité cosinus entre les vecteurs de caractéristiques des candidats et des offres permet d'évaluer la pertinence des correspondances. La similarité cosinus est définie comme le produit scalaire des vecteurs, normalisé par les normes des vecteurs.

$$\text{Similarité Cosinus} = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|}$$

8 Fonction de Recherche des Meilleures Correspondances

La fonction `get_all_best_match_candidates` permet de trouver les meilleurs candidats pour une entreprise spécifique en fonction de la similarité cosinus calculée.

```
def get_all_best_match_candidates(company_uid):
    offers_indices = offers_df[offers_df['uid'] == company_uid].index
    if len(offers_indices) == 0:
        return "No offers found for the company"
    best_candidate_scores = {}
    for offer_index in offers_indices:
        offer_features = offers_features.loc[[offer_index]]
        similarity_scores = cosine_similarity(offer_features, candidates_feature
        best_candidate_indices = similarity_scores.argsort()[0][::-1][:10]
        for candidate_index in best_candidate_indices:
            best_candidate_id = candidates_df.iloc[candidate_index]['uid']
            best_candidate_score = similarity_scores[0][candidate_index]
            best_candidate_scores[best_candidate_id] = {'Similarity Score': best
    return best_candidate_scores
```

9 Tester l'Algorithme

Pour tester l'algorithme, il suffit d'entrer l'identifiant de l'utilisateur, qu'il soit candidat ou recruteur. Le résultat est une liste de tous les candidats ou offres, ordonnée du score le plus élevé (le plus compatible) au plus bas.

10 Conclusion

Le système développé pour MidlJob permet d'identifier efficacement les meilleurs candidats pour les offres d'emploi en se basant sur une analyse exhaustive des caractéristiques des deux parties. En utilisant des techniques d'encodage avancées et la similarité cosinus, nous assurons un appariement précis et pertinent pour les entreprises et les candidats.

11 Références

— Predict Tinder Matches with Machine Learning - GeeksforGeeks
— Data Encoding - StudySmarter
— Machine Learning - Udemy Course