ESIREM
ÉCOLE SUPÉRIEURE D'INGÉNIEURS
NUMÉRIQUE ET MATÉRIAUX

École associée Polytech

Université de Bourgogne

Systèmes Intelligents Avancés TP
The Final Countdown

Auteurs :
Marcelo LOPES
Benjamin MILHET

Professeur :
M. BROUSSE

marcelo_lopes-castanheira@etu.u-bourgogne.fr
benjamin_milhet@etu.u-bourgogne.fr

2023 - 2024

Table des matières

Introduction

L'objectif de ce projet pour la matière systèmes intelligents avancés est d'utiliser l'intelligence artificielle dans un cas pratique en développant nous-même un projet complet. Nous avons décidé d'utiliser nos connaissances acquises pour réaliser une application de prédiction de la date de mort d'une personne. Le site web est composé d'un questionnaire de 25 questions sur plusieurs aspect d'une personne et de sa vie. Ces informations sont ensuite envoyées à notre API que nous avons conçu et qui retourne la date estimée de la mort de la personne. Une fois le calcul par notre algorithme terminée, le questionnaire du site disparaît pour afficher un compte a rebours jusqu'à la date fatidique.

Le code du projet est disponible sur notre dépôt : Github

Le Dataset

2.1 Introduction

Le choix du dataset est l'une des étapes les plus importantes du projet et nous permettra d'entraîner notre modèle d'IA. Pour ce projet, il nous a été assez compliqué d'obtenir des informations sur des personnes décédés en libre accès, ce type de données est assez réglementé et confidentiels en Europe. Le dataset que nous avons choisi et qui nous semblait le plus pertinent provient de la plateforme Kaggle qui fournit des jeux de données. Le dataset sélectionnées est le seul qui contenait autant d'information différentes sur les personnes avec un total de 121 colonnes pour plus de 770 000 lignes de données. Le seul inconvénient de ce dataset est le pays d'origine qui est l'Inde. En effet les indiens ont un mode de vie assez différents des pays européens avec une espérance de vie entre 69 à 72 ans contre 80 à 85 ans pour la France. Cependant c'est le seul jeu de données que nous avons trouvés pour notre projet avec autant d'information pour rendre notre prédiction assez pertinentes.

Le jeu de données est disponible sur notre dépôt github : Dataset

2.2 La sélection des données

Une fois le dataset sélectionné, ils nous faut réaliser une sélection des colonnes que nous utiliserons dans notre algorithme parce que certaines colonnes ne sont pas pertinentes ou que le nombre de réponse est trop faible ou imprécise. Le choix de ces informations est une phase très importante afin d'obtenir un dataset intéressant et clair pour nos futurs traitements.

Le premier filtrage des colonnes est de supprimer toutes les colonnes liées a comment les personnes sont décédés, les symptômes de sa mort ou si la personne était enceinte lorsqu'elle est morte parce que nous allons faire des prédictions sur des personnes vivantes.

Le second filtrage consiste a supprimer les colonnes dont les informations ne sont pas disponible ou compatible avec la France comme par exemple la ville d'origine qui correspond uniquement à des villes indiennes, la consommation de tabac a chiqué qui est interdit en France ou si la personne était le chef de sa famille.

Une fois la sélection des colonnes terminées, ils nous reste a vérifier le contenu des lignes pour ne pas qu'il y est d'anomalie mais aussi réaliser quelque ajustement pour correspondre au mieux à des réponses en lien avec des personnes françaises. Par exemple pour la colonne éducation, nous avons du adapter et regrouper certaine réponse pour correspondre au système de l'éducation française en fonction des années d'études.

Ci-dessous l'ensemble des 36 colonnes que nous avons conservé avec une définition pour chacune :

1. **age** : Âge de la personne.
2. **sex** : Sexe biologique de la personne.
3. **marital_status** : Statut matrimonial actuel de la personne (par exemple, célibataire, marié, divorcé).
4. **occupation_status** : Situation professionnelle ou occupation actuelle de la personne.
5. **highest_qualification** : Niveau d'éducation ou de qualification le plus élevé atteint par la personne.
6. **religion** : Affiliation religieuse de la personne.
7. **order_of_birth** : La position de la personne dans l'ordre de naissance au sein de sa famille (par exemple, aîné, second enfant).
8. **diagnosed_for** : Toutes les conditions médicales ou maladies dont la personne a été diagnostiquée.
9. **iscoveredbyhealthscheme** : Indique si la personne est couverte par une assurance maladie ou un régime de soins de santé.
10. **disability_status** : Détails de tout handicap que la personne pourrait avoir.
11. **regular_treatment** : Indique si la personne suit un traitement médical régulier.
12. **rural** : Indique si la personne vit dans une zone rurale.
13. **owner_status** : Statut de propriété du logement de la personne (par exemple, propriétaire, locataire).
14. **land_possessed** : Informations sur la propriété ou la possession de terre.
15. **drinking_water_source** : Source principale d'eau potable pour le ménage de la personne.
16. **is_water_filter** : Indique si le ménage utilise un filtre à eau.
17. **is_toilet_shared** : Si les installations sanitaires sont partagées avec d'autres ménages.
18. **household_have_electricity** : Indique si le ménage a accès à l'électricité.
19. **lighting_source** : Source principale d'éclairage utilisée dans le ménage.
20. **cooking_fuel** : Type de combustible utilisé pour la cuisson dans le ménage.
21. **no_of_dwelling_rooms** : Nombre de pièces dans le logement de la personne.
22. **kitchen_availability** : Si le ménage dispose d'une cuisine.
23. **is_radio** : Indique si le ménage possède une radio.
24. **is_television** : Indique si le ménage possède une télévision.
25. **is_computer** : Indique si le ménage possède un ordinateur.
26. **is_telephone** : Indique si le ménage possède un téléphone.
27. **is_washing_machine** : Indique si le ménage possède une machine à laver.
28. **is_refrigerator** : Indique si le ménage possède un réfrigérateur.
29. **is_sewing_machine** : Indique si le ménage possède une machine à coudre.
30. **is_bicycle** : Indique si le ménage possède un vélo.
31. **is_water_pump** : Indique si le ménage possède une pompe à eau.
32. **is_scooter** : Indique si le ménage possède un scooter.
33. **is_car** : Indique si le ménage possède une voiture.
34. **is_tractor** : Indique si le ménage possède un tracteur.
35. **smoke** : Informations sur les habitudes de tabagisme de la personne.
36. **alcohol** : Informations sur les habitudes de consommation d'alcool de la personne.

2.3 Conclusion

Cette partie était la plus importante car le choix des informations va est ce qui va rendre nos modèles d'intelligence artificiel pertinent ou non. De plus, il est important de s'assurer qu'aucune colonnes importantes n'a été oublié.

Le site web

3.1 Introduction

Notre application se divise en deux partie distincts, le frontend qui comprend toute l'interface utilisateur du site avec une question par colonnes du dataset. Les réponses de ce formulaire seront envoyées à la deuxième partie de notre application, le backend qui comprend une API pour créer une interaction entre les deux parties et notre modèle d'intelligence artificiel. Nous avons décidé d'utiliser le framework web Angular pour développement de notre site web parce que celui-ci est populaire et nous souhaitons le prendre en main une première fois en vue de nos stages respectifs.

3.2 Angular

Les principaux avantages d'utiliser le framework web Angular sont sa structure claire et modulaire, son utilisation du modèle MVC et sa maintenabilité grâce au support de Google qui réalise régulièrement des mises à jour. L'avantage de sa structure est la réalisation de composant réutilisable et très facilement personnalisable. De plus, Angular adopte un modèle MVC (Modèle-Vue-Contrôleur) flexible, permettant une séparation claire et efficace entre la logique métier, l'interface utilisateur et le contrôle des données. Enfin, grâce au support de Google, ce framework est souvent mis à jour permettant une garantie sur le lon termes.

3.3 Le questionnaire

Notre application web est composée principalement d'un questionnaire qui reprend le contenu des différentes colonnes de notre dataset. Nos questions sont stockées dans un objet contenant pour chaque ligne un id, la question, le type de champs de la questions et les réponses possibles comme le montre l'exemple suivant :

```
this.questions = [
    { id: 1, text: 'Quel est votre age ?', fieldType: 'number' },
    { id: 2, text: 'Quel est votre sexe ?', fieldType: 'select', options: [{ label: 'Homme', value:
        '1' }, { label: 'Femme', value: '2' }] },
    { id: 3, text: 'Quel est votre statut marital ?', fieldType: 'select', options: [{ label:
        'Celibataire', value: '1' }, { label: 'Marie(e)', value: '2' }, { label: 'Remarie(e)',
        value: '4' }, { label: 'Veuf(ve)', value: '5' }, { label: 'Divorce(e)', value: '6' }, {
        label: 'Non precise', value: '8' }] },
  ]
```

Le site web est composé de 3 pages de 6 questions avec principalement des des champs de type 'select'. Ce type de champs permet de contraindre l'utilisateur a des choix précis correspondant aux réponse possible dans notre dataset.

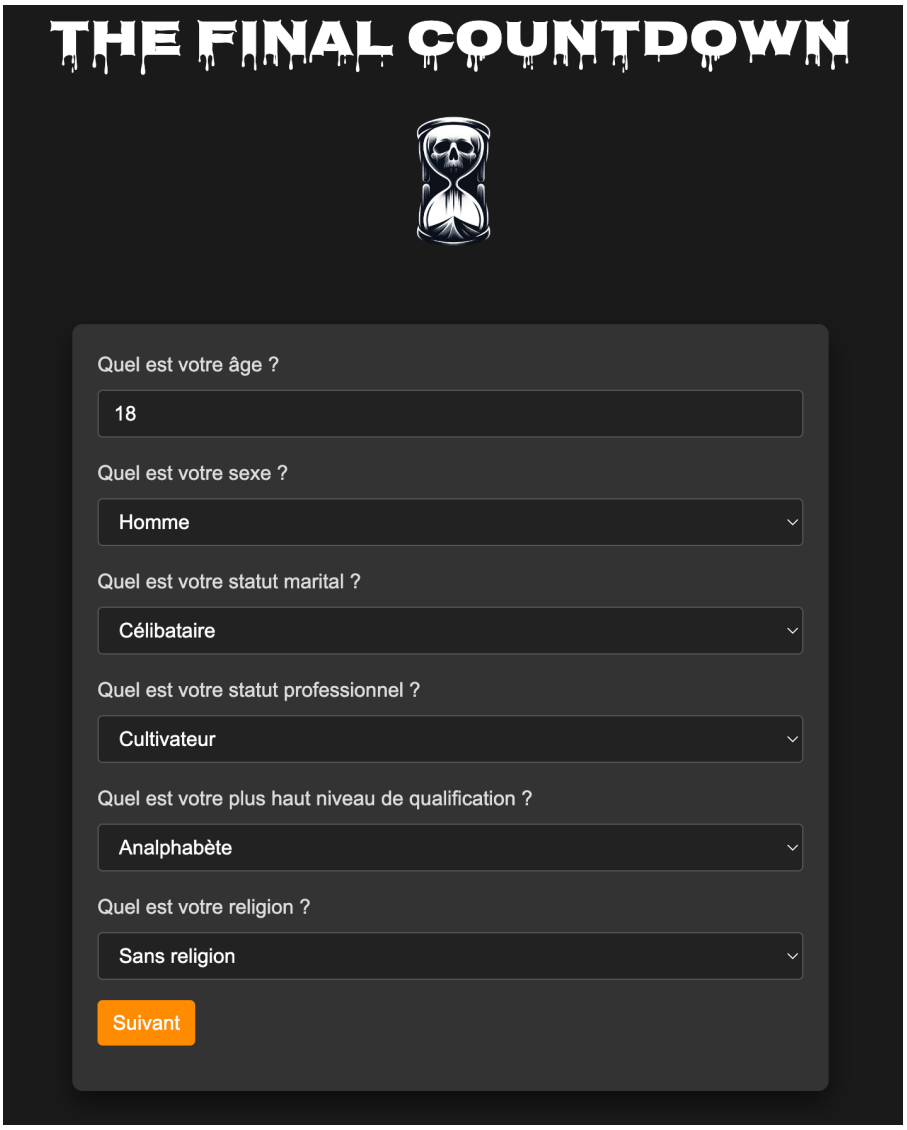


FIGURE 3.1 – Page d'accueil du site contenant la partie 1 du questionnaire

La dernière page du questionnaire est composé d'une liste d'objet que pourrait posséder une personne avec un système de case à cocher :

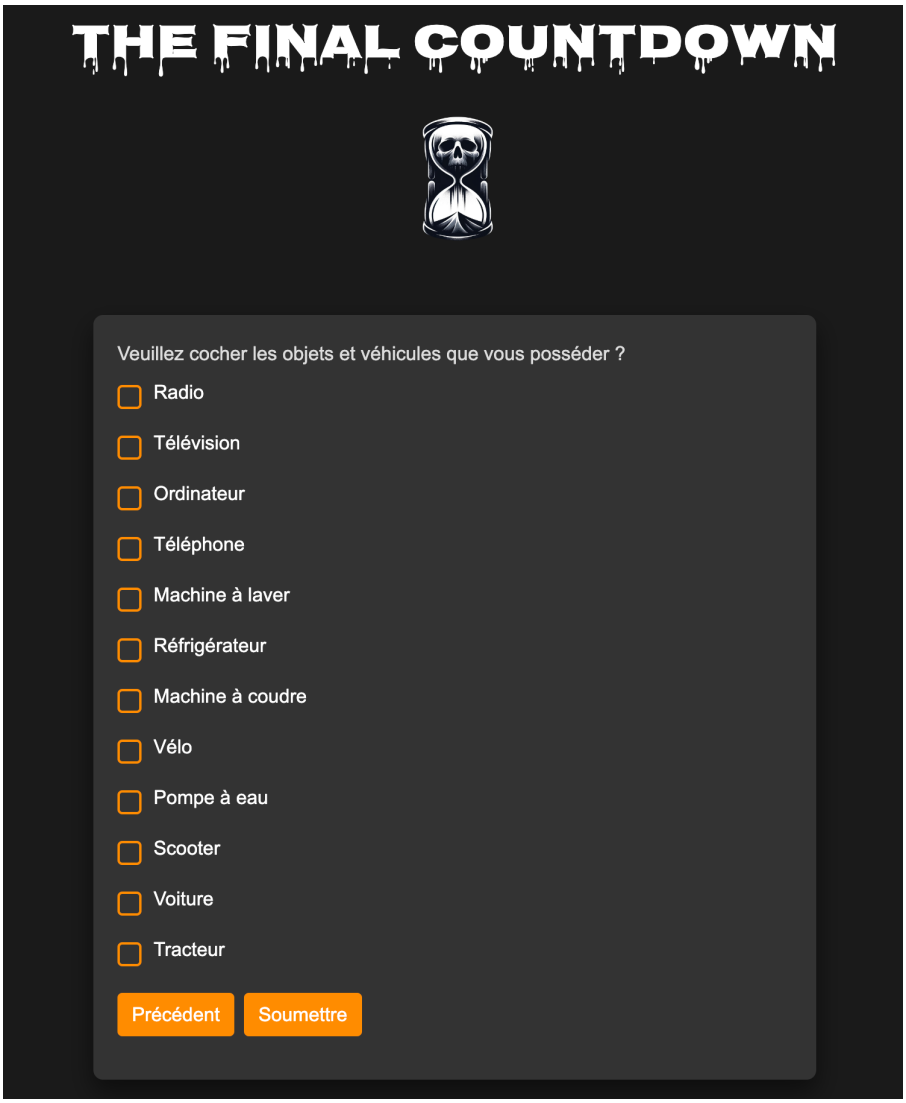


FIGURE 3.2 – Dernière page du questionnaire

Une fois le questionnaire terminée, les données sont envoyées sous la forme d'un dictionnaire à l'API Python avec comme paramètre l'id de la colonne et la réponse du questionnaire pour cette question sous cette forme :

```
{'age': 18, 'sex': '1', 'marital_status': '1', 'occupation_status': '1', 'highest_qualification':
    '0', 'religion': '8', 'order_of_birth': '0', 'diagnosed_for': '0', 'iscoveredbyhealthscheme':
    '1', 'disability_status': '0', 'regular_treatment': '3', 'rural': '1', 'owner_status': '1',
    'land_possessed': '1', 'drinking_water_source': '1', 'is_water_filter': '1',
    'is_toilet_shared': '2', 'household_have_electricity': '1', 'lighting_source': '1',
    'cooking_fuel': '7', 'no_of_dwelling_rooms': '2', 'kitchen_availability': '1', 'smoke': '4',
    'alcohol': '4', 'is_radio': '2', 'is_television': '1', 'is_computer': '1', 'is_telephone':
    '2', 'is_washing_machine': '2', 'is_refrigerator': '2', 'is_sewing_machine': '2',
    'is_bicycle': '2', 'is_water_pump': '2', 'is_scooter': '2', 'is_car': '2', 'is_tractor': '2'}
```

3.4 Le décompte

Une fois les données envoyées à notre API Python, celle-ci appel notre modèle d'IA qui effectue son traitement expliqué dans la partie suivante et nous renvoie une date correspondant à la prédiction de la mort de la personne. La date qui nous est retournée est stocké dans le localstorage du site web, ce qui permet de stocké la date même lors d'une actualisation. Cela nous permet d'empêcher la personne de repasser le questionnaire. De plus, lorsque la personne retourne sur notre site, le décompte s'affiche automatiquement à la place du questionnaire. Il est facile de contourner cela en supprimant le cache et le données du site, mais nous voulions ajouter un coté un peu plus immersif.

FIGURE 3.3 – Décompte jusqu'à la date de mort

3.5 Conclusion

La réalisation du frontend nous a permis de découvrir le framework web Angular qui nous sera utile dans nos stage respectif. De plus, cela permet d'intégrer notre modèle d'IA dans un projet concret et complet.

Le Modèle de prédiction

4.1 Introduction

Pour ce projet, une étape cruciale est l'analyse approfondie des données. Comprendre la répartition et les corrélations entre les différentes variables est essentiel pour établir les fondements de notre modèle. Cette démarche méthodique nous guide dans le choix des modèles d'intelligence artificielle. Ainsi, cette section détaille notre approche visant à décortiquer les données, puis à sélectionner le modèle optimal pour notre application prédictive.

4.2 Analyse des Données et Préparation

La première étape a consisté à examiner les corrélations au sein du jeu de données. La matrice de corrélation a révélé des relations telles que l'association entre la possession d'une télévision et la disponibilité de l'électricité dans la maison, ce qui semble plutôt logique. Cependant, des corrélations plus étranges comme la forte corrélation entre la possession d'un tracteur et les habitudes de tabagisme et de consommation d'alcool ont été observées. Il est donc important de prendre en compte ces analyses lors du choix du modèle et de l'interprétation des résultats.

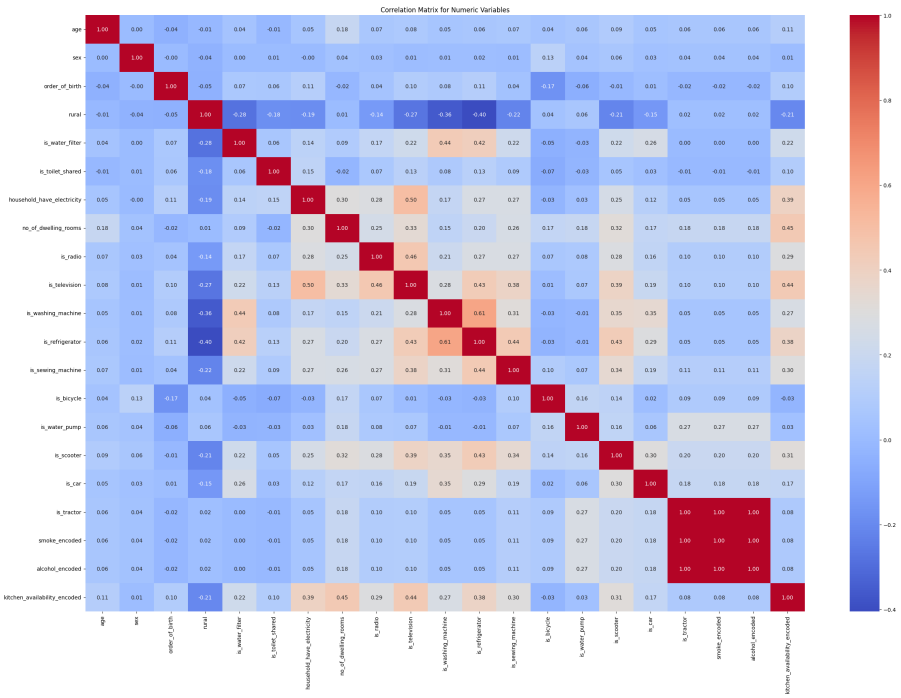


FIGURE 4.1 – Matrice de corrélation pour les variables numériques

Par la suite, nous avons visualisé la distribution des valeurs de chaque colonne. L'analyse de la distribution de l'âge a révélé une centralisation autour de 50 ans, ce qui est assez bas par rapport à l'âge moyen en France, qui est d'environ 80 ans.

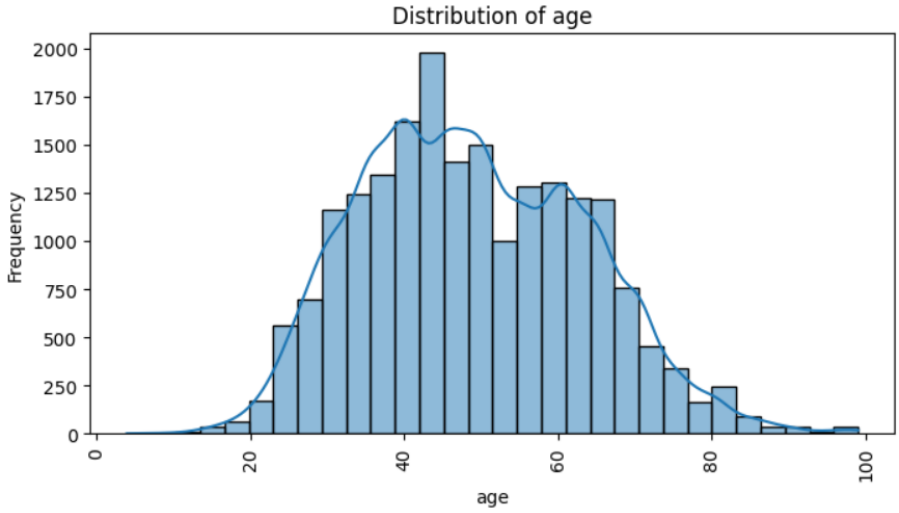
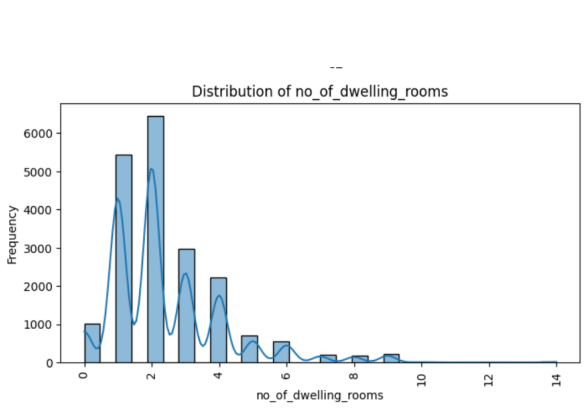
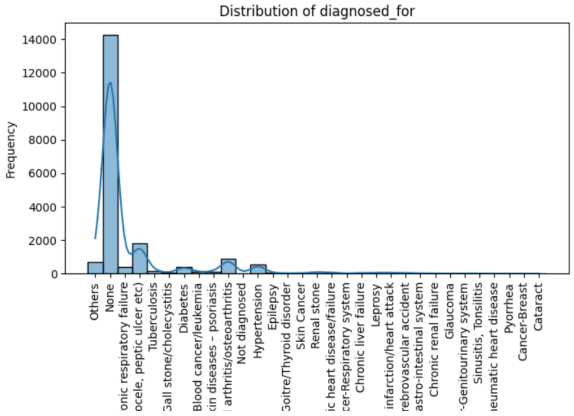


FIGURE 4.2 – Distribution de l'âge dans le jeu de données

Des distributions gaussiennes ont été identifiées, notamment pour le nombre de pièces d'habitation, mais aussi des distributions asymétriques, comme dans le cas des différents types de maladies possibles. Ces informations sont très importantes, car elles impliquent que Naive Bayes ne fonctionnerait pas très bien, tandis que les arbres de décision, le boosting et les forêts aléatoires seraient mieux adaptés à ce travail.



(a) Distribution du nombre de pièces d'habitation



(b) Distribution des différents types de diagnostics des personnes

FIGURE 4.3 – Exemples de distribution

Enfin, la visualisation du test du khi-carré et de la valeur P peut nous fournir des indications sur l'indépendance entre les variables. Le test du khi-carré évalue l'association entre les variables, et la valeur P indique la probabilité que cette association soit due au hasard. Dans le graphe, les colonnes à gauche nous montrent une importante association avec l'âge, et la décroissance du test du khi-carré montre la diminution de l'importance de l'association. La valeur P inférieure à 0,05 pour toutes les colonnes ne nous permet pas de rejeter l'hypothèse selon laquelle ces variables ne sont pas indépendantes de l'âge de décès.

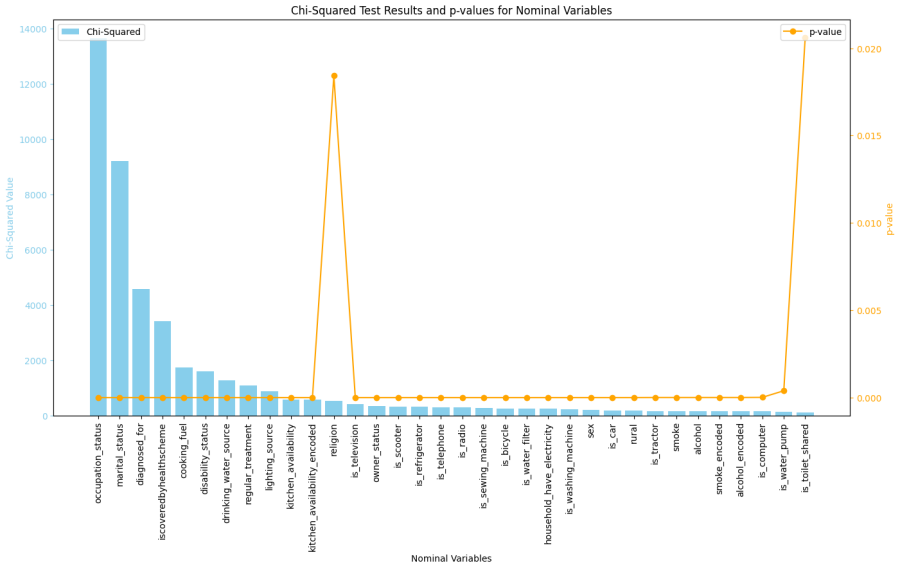


FIGURE 4.4 – Résultats du test du Khi-carré et valeurs P pour les variables nominales

Ces analyses approfondies éclairent notre compréhension des données et orientent notre choix de modèle pour la prédiction de l'âge de décès.

4.3 Les modèles

Le processus de sélection du modèle a débuté par l'implémentation d'un modèle de régression linéaire standard en tant que point de départ. Cependant, les résultats obtenus en regardant l'erreur quadratique moyenne (MSE) et en visualisant des exemples, ont montré que cela ne répondait pas pleinement aux attentes.

```
Mean Squared Error: 128.4969463766922
       Actual  Predicted
4924       44  43.374022
11571      42  40.089192
14601      51  44.784122
2218       40  32.824037
19861      66  70.765274
8434       44  44.604372
13417      38  42.005439
1628       47  57.866175
17217      35  45.578396
5944       29  41.860577
2355       46  50.074561
18269      65  52.965615
5671       35  45.156619
4936       30  41.048588
8746       45  49.916782
10390      40  41.099779
15090      56  51.718650
5921       55  45.576370
4649       70  54.167976
19352      37  63.740997
```

FIGURE 4.5 – Résultats du premier modèle

Afin de mieux comprendre les lacunes du modèle, une analyse détaillée des valeurs de MSE à chaque tranche d'âge a été entreprise. Cette investigation a révélé que le modèle présentait des faiblesses autour des valeurs aberrantes, notamment celles en dessous de 20 ans et au-dessus de 75 ans, suggérant une gestion insuffisante de ces tranches d'âge, peut-être en raison d'un manque de données.

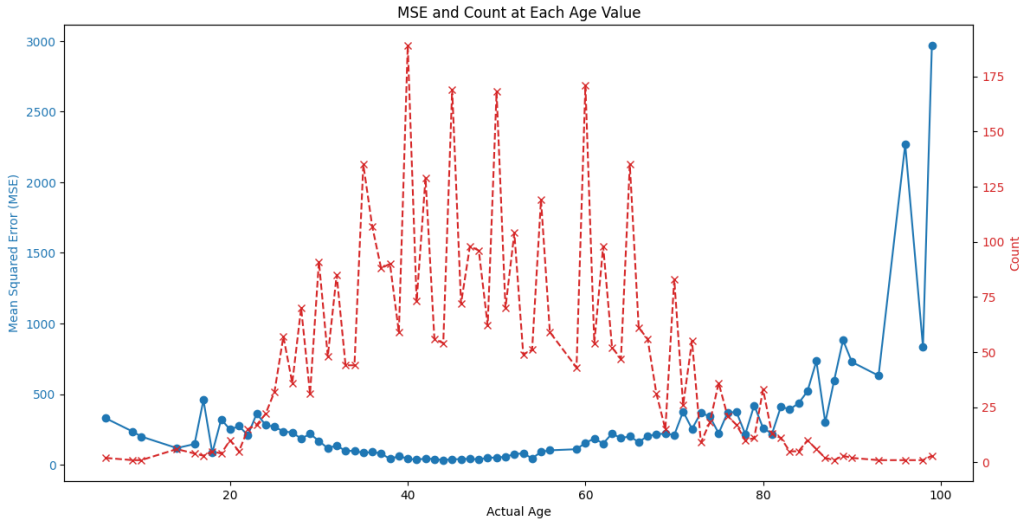


FIGURE 4.6 – MSE et nombre d'occurrences à chaque valeur d'âge

Par la suite, une alternative a été explorée avec un modèle Support Vector Machine (SVM) utilisant un noyau linéaire. L'avantage de ce choix résidait dans la moindre sensibilité de SVM aux valeurs aberrantes. Cependant, malgré cette caractéristique bénéfique, les performances du modèle n'ont pas été meilleures. D'autres modèles basés sur des arbres de décision ont également été testés, mais les résultats en termes de MSE se sont avérés encore plus élevés.

La démarche a alors conduit à l'utilisation d'un modèle XGBoost Regressor, un choix motivé par sa robustesse et sa capacité à traiter des ensembles de données complexes. Initialement, les performances n'étaient pas optimales, mais grâce à une optimisation des hyperparamètres à l'aide de GridSearchCV de scikit-learn, une recherche systématique des combinaisons optimales pour max_depth, n_estimators, et learning_rate a été entreprise. Cette approche a abouti à une amélioration significative des performances, rendant le modèle XGBoost le plus performant parmi les modèles testés avec une MSE de 123.

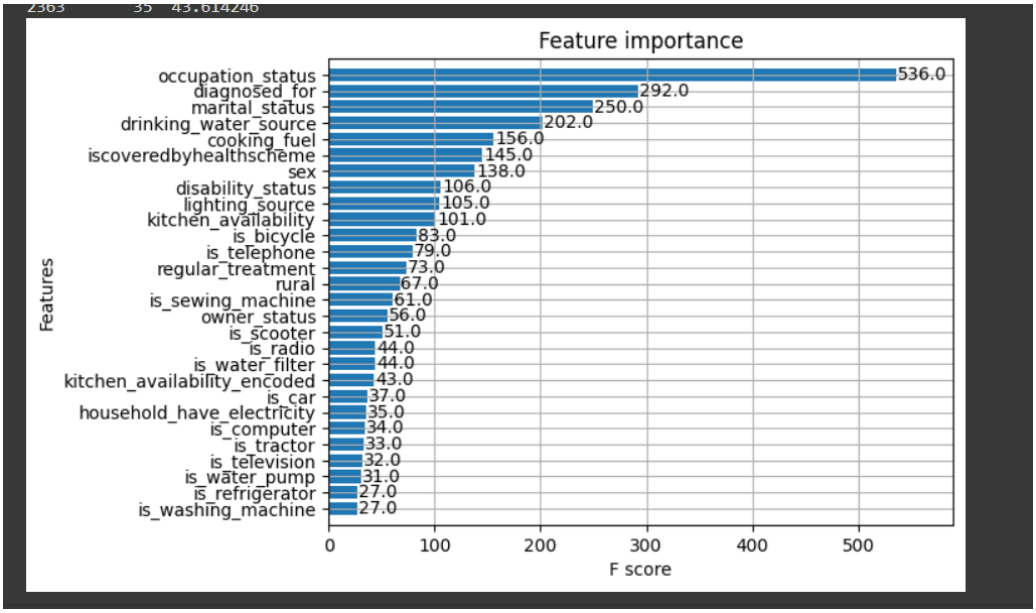


FIGURE 4.7 – Niveau d'importance des colonnes dans le modèle XGBoost

Cette visualisation indique quelles caractéristiques ont le plus d'impact sur les prédictions du modèle. Nous pouvons observer que la colonne ayant le plus d'influence sur le modèle est celle indiquant le statut d'activité de la personne. De plus, il est notable que les colonnes relatives à la consommation de tabac/alcool et à la possession de tracteur ne sont pas incluses dans cette visualisation, ce qui est positif. En effet, cela concorde avec nos observations dans la matrice de corrélation, où nous avions identifié ces variables comme potentiellement problématiques.

4.4 API Flask

Dans le cadre de notre projet, nous avons mis en place une mini-API en utilisant Flask, un framework Python léger et efficace pour le développement web. Notre API possède une seule route avec une requête POST qui permet d'envoyer un dictionnaire avec les réponses de notre formulaire. Ces données sont envoyées à notre modèle d'intelligence artificielle qui nous retourne une date correspondant à la mort de la personne sous ce format : "yyyy-mm-ddThh :mm :ss".

```
@app.route("/getDate", methods=['POST'])
def getDate():
data = request.get_json()
date = modele.getDate(data)
return jsonify({"date": date})
```

4.5 Conclusion

En résumé, le modèle XGBoost Regressor, bien que prometteur après une optimisation, est loin d'être parfait. L'une des causes principales est sans doute les données, notamment leur provenance et certaines incohérences observées lors de la matrice de corrélation ou encore lors de la visualisation de certaines colonnes.

Conclusion

Ce projet nous a permis de mettre en relation les compétences et la méthode de travail acquises lors des cours et exercices de systèmes intelligents avancés, mais aussi, plus généralement, de l'ensemble des cours suivis à l'ESIREM, allant de la programmation web au cours de gestion de projet. Ce projet combine la création de notre modèle d'intelligence artificielle à l'utilisation d'une API Python Flask et à la réalisation de notre propre frontend à l'aide du framework web Angular.

Ce projet met également en lumière que le traitement des données est aussi crucial que le choix du modèle. Des données de qualité, complètes et cohérentes, sont essentielles pour des prédictions robustes. Ainsi, pour maximiser l'efficacité des modèles d'intelligence artificielle, une attention particulière à la qualité des données s'avère impérative.