CHEMIEL Houda
CARDENAS KHARITONOVA Francisco
TISSEUR Maude

Projet d'Ingénierie des données

Application Data Science

Professeur référent : BOUNEKKAR Ahmed

Année scolaire : 2023-2024

Sommaire

Introduction……...p.3

Interface utilisateur……..p.5

Modules de l’application ……p.9

Difficultés rencontrées……..p.10

Introduction

Afin d'évaluer les traitements pour les patients atteints de myélome, nous avons réalisé une analyse de survie et une analyse coût-efficacité. L'analyse de survie est quant à elle une méthode qui permet d'évaluer la durée de vie des patients après le diagnostic ou le traitement, elle nous permet donc de nous renseigner sur l'efficacité des traitements. L'analyse de coût-efficacité offre la possibilité de comparer les dépenses liées et les résultats de différentes interventions médicales, ce qui permet de déterminer quel traitement propose le meilleur rapport qualité-prix.

Outre l'analyse de survie, l'objectif de ce projet est également de fournir une interface (Web) permettant d'afficher les résultats de l'analyse à l'aide de Streamlit. Streamlit est une bibliothèque Python qui offre aux développeurs la possibilité de concevoir rapidement des applications web pour visualiser des données. Dans le domaine de l'analyse de survie, une discipline de la statistique axée sur l'analyse du temps jusqu'à un événement d'intérêt, Streamlit peut simplifier l'interaction avec les modèles statistiques et les visualisations. Par exemple, l'application offre aux utilisateurs la possibilité de modifier les paramètres des modèles de survie, comme le modèle de Kaplan-Meier, et de visualiser immédiatement l'effet sur les courbes de survie.

L'analyse est effectuée à partir d'un jeu de données contenant des informations cliniques et démographiques sur des patients atteints de myélome multiple. Le myélome est un cancer du sang qui affecte un type de cellules du système immunitaire en particulier : les plasmocytes. Chaque enregistrement du jeu de données représente un patient identifié par un numéro unique, et les variables incluent des données sur le genre, l'IMC et le régime d'affiliation. Les antécédents médicaux et les conditions associées comme l'anémie, l'hypercalcémie, et les différents stades de la maladie rénale chronique (légère, modérée, sévère, sous dialyse) sont également documentés. D'autres aspects cliniques, tels que la présence de lésions osseuses, d'infections récurrentes, et de fragilités, ainsi que les altérations génétiques spécifiques (FISHdel17p1, FISHt_1114, FISHt414, FISHamp1q211) sont inclus. Le jeu de données détaille également les sous-classifications et les stades ISS du myélome multiple, les traitements administrés et l'adhérence au traitement. Des variables sur les opportunités de traitement, les jours de traitement suspendus, et les jours en thérapie sont présentes. Les résultats cliniques, comme la réponse clinique, les évènements (hospitalisation, rémission) et les coûts associés au traitement complètent les informations. Enfin, des données démographiques supplémentaires telles que l'âge, le pays et l'hôpital de traitement sont incluses pour chaque patient.

Afin de réaliser cette tâche, nous avons fait appel à diverses bibliothèques :

- **Streamlit (st)** est une bibliothèque open-source qui offre la possibilité de concevoir des applications web interactives pour l'apprentissage automatique et la science des données. Elle rend le développement plus facile en convertissant les scripts Python en applications web interactives.
- **NumPy (np)** est une bibliothèque utilisée pour effectuer des calculs numériques en Python. Elle offre un outil pour les tableaux à plusieurs dimensions et des fonctions mathématiques pour les manipuler.
- **Pandas (pd)** est une librairie Python qui permet de manipuler et d'analyser des données. Elle met à disposition des structures de données simples à utiliser, telles que les DataFrames, ainsi que des outils permettant de réaliser des opérations sur ces informations.
- **Lifelines** est une extension Python qui permet d'analyser la survie et de modéliser les durées. Elle offre des instruments permettant d'évaluer les courbes de survie, de réaliser la régression de Cox et d'autres méthodes d'analyse de survie.
- **Scikit-learn** est une bibliothèque open-source de Python pour l'apprentissage automatique. Son offre comprend une variété d'algorithmes d'apprentissage supervisé et non supervisé, ainsi que des outils permettant d'évaluer, de choisir et de prétraiter les données.
- **Plotly** offre une interface interactive pour visualiser des données. Son utilisation permet de concevoir des graphiques et des diagrammes interactifs dans les applications en ligne.
- **Matplotlib** est une bibliothèque Python qui permet de visualiser des données. Son utilisation offre la possibilité de concevoir une multitude de graphiques statiques, tels que des histogrammes, des diagrammes en barres et des nuages de points.
- **Seaborn** est une bibliothèque Matplotlib qui permet de visualiser des données. Son utilisation facilite la conception de graphiques esthétiques en offrant des fonctionnalités de pointe pour des tâches courantes.

L'application Streamlit peut être déployée sur GitHub en utilisant Streamlit Cloud pour l'hébergement de l'application et GitHub pour le contrôle de version et le déploiement web.

Dans un premier temps, il est nécessaire de créer un nouveau dépôt GitHub et d'y insérer le code source. Il est nécessaire d'inclure un document `*requirements.txt*` qui répertorie toutes les dépendances requises pour la mise en œuvre de notre application. Ensuite, il est nécessaire de se rendre sur le site de Streamlit Community Cloud et de se connecter à GitHub. Il est possible de choisir un dépôt ainsi que la branche qui contient notre application. Le Cloud Communauté Streamlit générera une URL unique pour notre application, que nous pourrons partager avec d'autres utilisateurs. Afin de rendre le déploiement automatique, il est

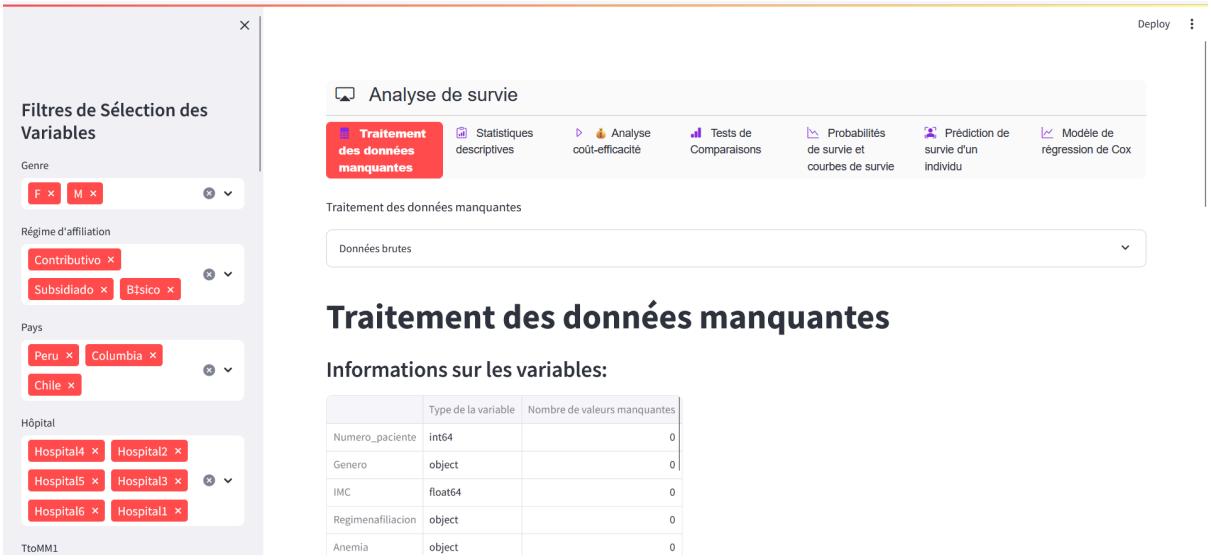
possible de configurer des GitHub Actions qui vont créer et déployer notre application chaque fois que l'on insère un nouveau code dans la branche spécifiée. Ceci assure que notre application est constamment mise à jour

Pour plus de détails sur les commandes spécifiques et les étapes de configuration, vous pouvez consulter la documentation officielle de Streamlit.

Code d'installation des librairies :

```
pip install streamlit
pip install numpy
pip install streamlit-option-menu
pip install lifelines
pip install pandas
pip install scikit-learn
pip install plotly
pip install matplotlib
pip install seaborn
```

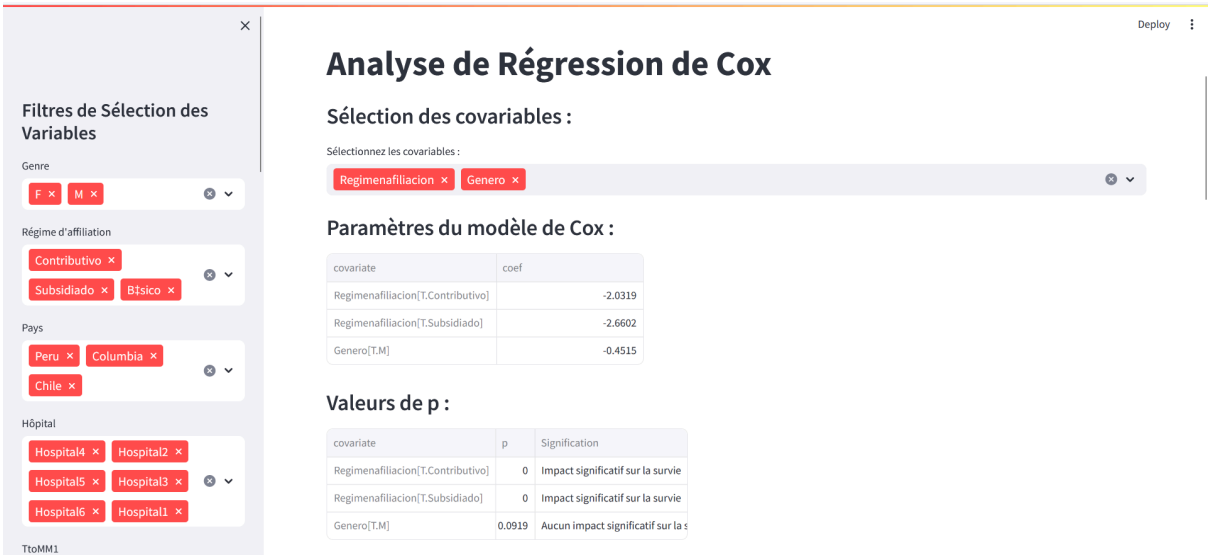
Interface utilisateur

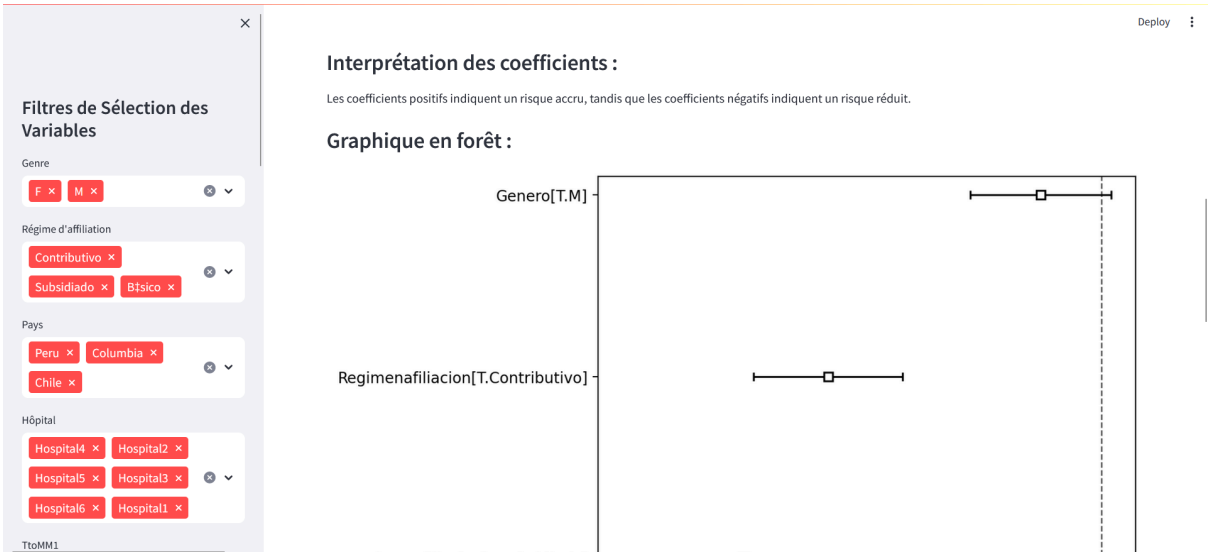


La barre latérale, à gauche de l'écran, permet de trier les informations en fonction de critères particuliers tels que le sexe, le pays, l'établissement hospitalier, le type de traitement, etc. Les utilisateurs ont la possibilité de choisir les filtres adaptés à leurs besoins d'analyse et d'observer les données se mettre à jour immédiatement en fonction de leurs choix. Ils donnent à l'utilisateur un contrôle complet sur les données qu'il souhaite analyser, lui permettant ainsi de réaliser des analyses plus précises et ciblées.

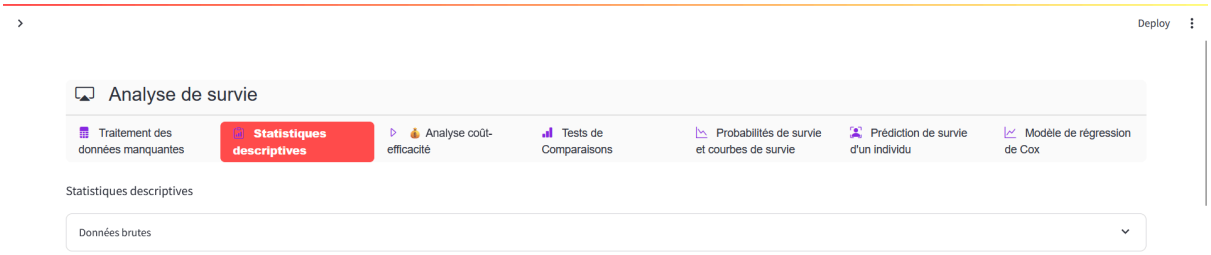
La partie principale de l'interface est située au centre de l'écran, où sont affichés les résultats de l'analyse et les visualisations. Cette partie est en constante

évolution et s'ajuste en fonction de la sélection effectuée dans la barre latérale. À titre d'exemple, si un utilisateur décide de réaliser une analyse de régression de Cox, cette partie présentera les résultats de cette analyse, incluant les coefficients du modèle , les valeurs p, les graphiques de la fonction de survie, ainsi que d'autres informations pertinentes…

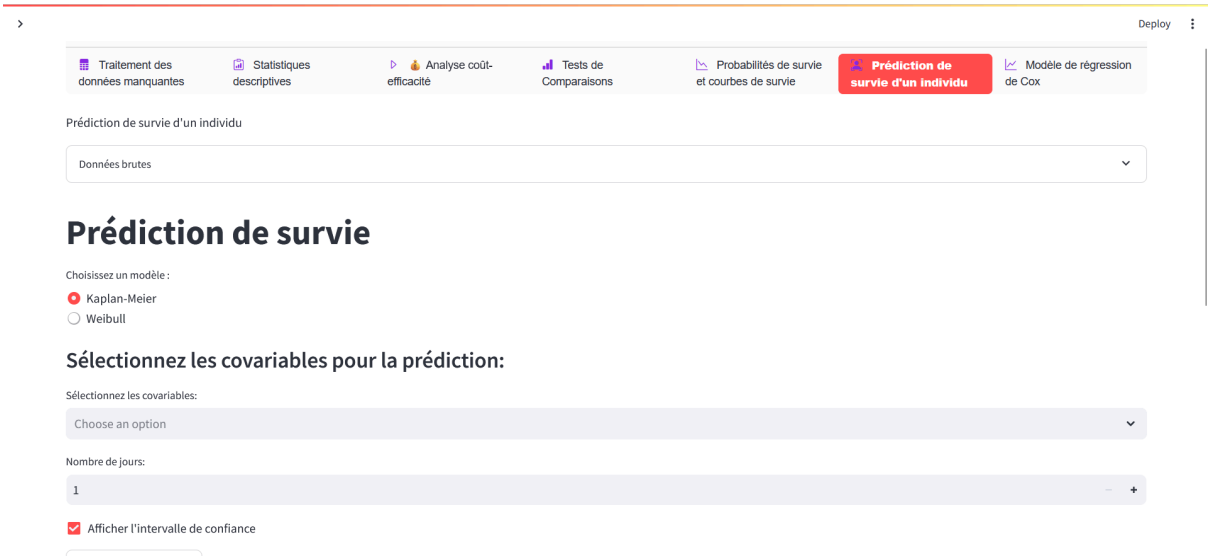




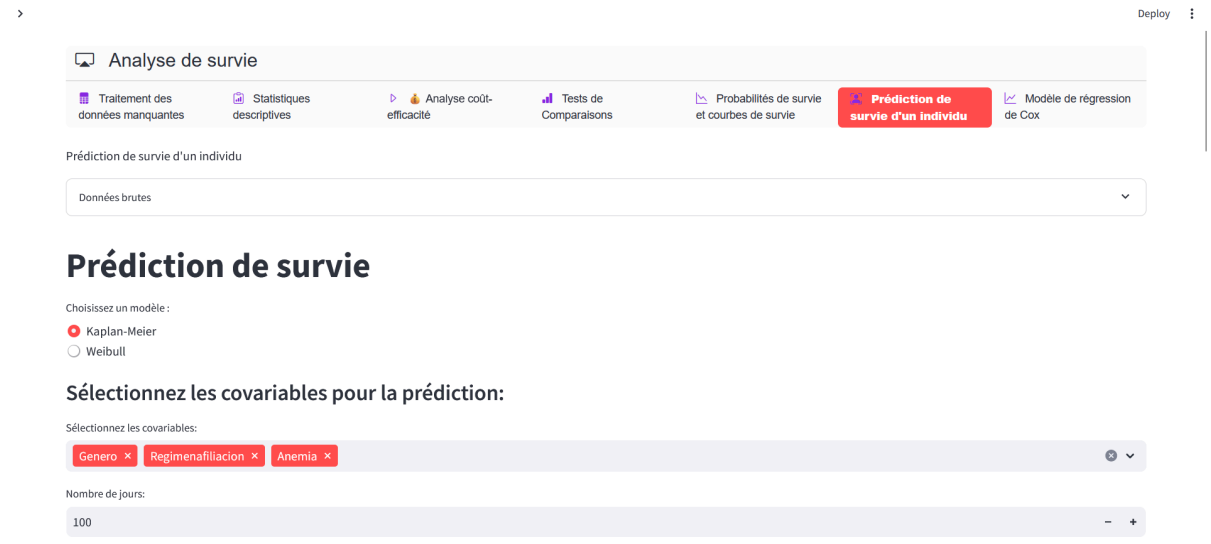
Enfin, l'interface comprend des boutons d'action et des options interactives qui offrent aux utilisateurs la possibilité de réaliser des actions précises, comme lancer une prédiction de survie, afficher des statistiques descriptives ou comparer différents traitements.



Pour l'onglet "Prédiction de survie", l'utilisateur sélectionne les covariables qu'il veut étudier ainsi que le nombre de jours souhaité.



A titre d'exemple :

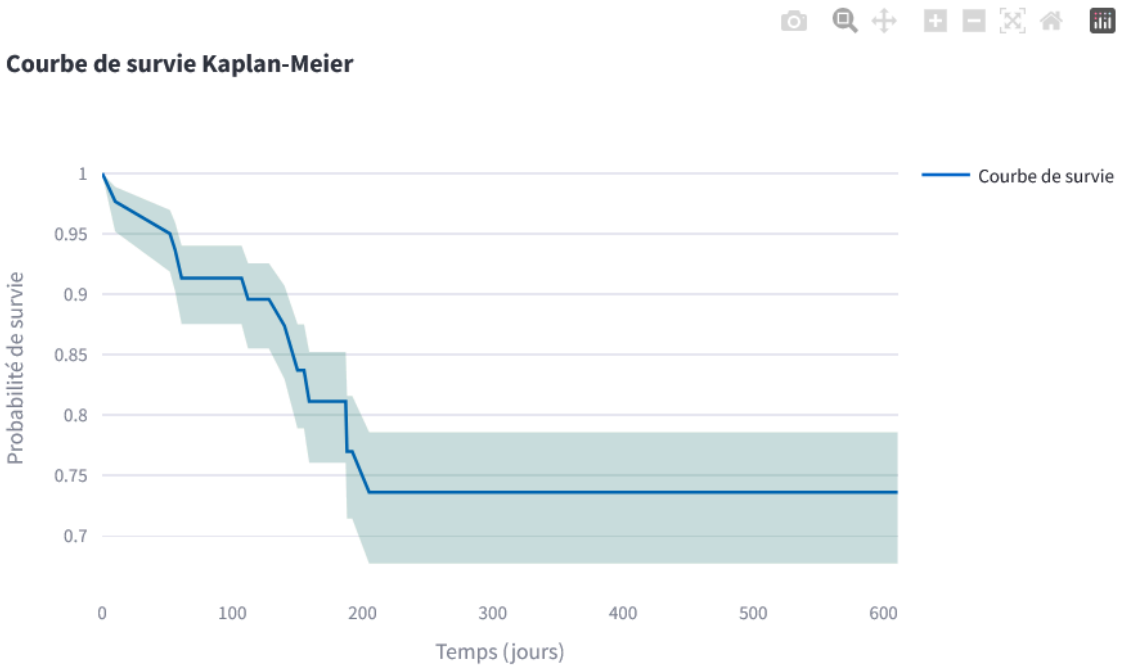


Ensuite, il faut cliquer sur le bouton “effectuer la prédiction" afin de générer la courbe et le tableau de survie associés.

Estimation de la survie pour `100` jours avec le modèle Kaplan-Meier:

`0.9133333333333337`

Courbe de survie Kaplan-Meier



Deploy

Tableau de survie Kaplan-Meier

	event_at	removed	observed	censored	entrance	at_risk
0	0	0	0	0	300	300
10	7	7	0	0	300	
52	8	8	0	0	293	
56	4	4	0	0	285	
61	7	7	0	0	281	
103	7	0	7	0	274	
107	7	0	7	0	267	
112	5	5	0	0	260	
128	11	0	11	0	255	
140	6	6	0	0	244	

	Numero_paciente	Genero	IMC	Regimenafiliacion	Anemia	Hipercalcemia	ERC_Leve	ERC_moderada	ERC_severa	ERC_dialisis	Lesiones_oseas	Infecciones_recurrentes	Fragilidad	FISHdel17p1
0	1,001	F	21.3	Contributivo	NO	NO	NO	NO	NO	SI	NO	NO	SI	NO
1	1,002	F	24.4	Subsidiado	NO	NO	NO	NO	SI	NO	NO	NO	NO	NO
2	1,003	M	21.8	Contributivo	NO	NO	NO	SI	NO	NO	SI	NO	NO	SI
3	1,004	F	26.2	Contributivo	NO	NO	NO	SI	NO	NO	NO	NO	NO	NO

Modules de l'application

`proba_survie()` : Elle est chargée de calculer les fonctions de survie et de risque à l'aide des méthodes de Kaplan-Meier et Nelson-Aalen, puis de les afficher graphiquement à l'aide de la bibliothèque Plotly. De plus, elle fournit une comparaison entre les fonctions de survie de Kaplan-Meier et Nelson-Aalen.

`traitement_donnees_manquantes()` : Cette fonction permet de traiter les données manquantes dans le jeu de données. Elle offre plusieurs méthodes de traitement, telles que la suppression des lignes ou des colonnes, l'imputation par la moyenne, la médiane, la valeur la plus fréquente ou les k-plus proches voisins (KNN). Elle affiche également le jeu de données après le traitement et offre la possibilité de télécharger le nouveau jeu de données au format CSV.

`test_comparaison()` : Cette fonction permet de comparer les courbes de survie entre différentes caractéristiques du jeu de données. Elle utilise les méthodes de Kaplan-Meier, Nelson-Aalen et Weibull pour estimer les fonctions de survie et les afficher graphiquement. L'utilisateur peut sélectionner la variable à utiliser pour la comparaison à partir d'une liste déroulante.

`cox_regression()` : Elle permet à l'utilisateur de sélectionner les covariables à inclure dans le modèle, ajuste le modèle de régression de Cox aux données et affiche les résultats, y compris les paramètres du modèle, les p-valeurs, les rapports de risque, etc.

`cost_analysis()` : Cette fonction réalise une analyse coût-efficacité. Elle permet à l'utilisateur de sélectionner un traitement, un pays et un hôpital, puis affiche les statistiques descriptives du coût du traitement ainsi qu'une comparaison avec d'autres traitements.

`preprocess_data()` : Cette fonction effectue le prétraitement des données en convertissant les variables catégorielles en variables indicatrices (One-Hot Encoding).

`survival_prediction()` : Cette fonction permet à l'utilisateur de prédire la survie d'un individu en fonction de covariables sélectionnées. Elle propose deux modèles de prédiction : Kaplan-Meier et Weibull. Elle affiche les estimations de survie ainsi que des graphiques correspondants.

`descriptive_statistics()` : Cette fonction réalise une analyse statistique descriptive. Elle permet à l'utilisateur de sélectionner une variable et affiche les statistiques descriptives ainsi qu'un graphique approprié (histogramme ou diagramme à barres).

Difficultés rencontrées

Durant le projet PID, nous avons rencontré quelques difficultés techniques. L'un d'eux a été le développement et la correction des bugs. Lorsque nous avons intégré les différentes parties du code, chacune développée par un membre de l'équipe, des problèmes d'incompatibilité et de multiples erreurs sont apparus. Cette phase d'unification a révélé des conflits, tels que des erreurs de développement et des problèmes d'affichage, nécessitant des ajustements.

La gestion des filtres a également été source de problèmes. Malgré un certain nombre de modifications, cette fonctionnalité ne fonctionne toujours pas correctement. Enfin, afin de mener à bien ce projet, nous souhaitons garantir le déploiement de notre application afin qu'elle puisse être accessible à quiconque via une URL provenant d'un référentiel GitHub. Toutefois, le processus de chargement du fichiers.csv rencontre des difficultés sans que l'on puisse expliquer pourquoi, car il est bien disponible dans le répertoire, comme on peut le constater lorsqu'on le charge dans un navigateur : Application.