Rapport

November 10, 2024

1 Rapport

1.1 Projet : Application de prédiction du cancer du sein

1.1.1 Réalisé par :

- **ARAGOU Wassef**
- **AGJOUD Imad**
- **MALIH Mouad**

1.1.2 Encadré par :

- **MR. LAMRANI Youssef**

2 *Remerciements*

Nous tenons à exprimer notre profonde gratitude à **MR. LAMRANI Youssef**, notre encadrant, pour sa précieuse supervision, ses conseils avisés et son soutien tout au long de la réalisation de ce projet.

3 *Table des Matières*

4 I. Introduction

Le cancer du sein est l'un des cancers les plus courants chez les femmes à travers le monde. Il représente un enjeu majeur de santé publique, avec des millions de nouveaux cas diagnostiqués chaque année. La détection précoce est essentielle pour améliorer les chances de survie et réduire la mortalité. Cependant, les méthodes traditionnelles de dépistage, telles que la mammographie et la biopsie, bien que fiables, peuvent être invasives, coûteuses, et parfois inaccessibles pour certaines populations.

Aujourd'hui, grâce aux progrès de l'intelligence artificielle (IA) et du machine learning, il est possible de développer des outils prédictifs pour aider à la détection précoce et à la gestion du cancer du sein. Ces technologies permettent d'analyser de vastes ensembles de données pour fournir des informations précises et rapides aux professionnels de santé, améliorant ainsi la qualité des diagnostics et des traitements.

Le présent projet s'inscrit dans cette dynamique et vise à concevoir une application de prédiction du cancer du sein. En s'appuyant sur des modèles de machine learning, cette application pourra prédire divers aspects du cancer, tels que le coût du traitement, le type de tumeur et le stade du cancer, offrant ainsi une aide précieuse pour les professionnels de la santé dans leur prise de décision.

5 II. Contexte du Projet

Le cancer du sein reste un problème majeur dans le domaine médical, malgré les nombreuses avancées scientifiques et technologiques. En particulier, la difficulté de diagnostic rapide et non invasif limite parfois les options de traitement, surtout dans les zones où les infrastructures médicales sont limitées. Les méthodes traditionnelles de diagnostic, comme les mammographies et les biopsies, sont efficaces mais comportent des inconvénients, notamment leur coût élevé, leur caractère invasif, et le temps nécessaire pour obtenir des résultats.

Dans ce contexte, l'usage des techniques d'intelligence artificielle et de machine learning offre de nouvelles opportunités pour améliorer les outils de diagnostic et de prédiction. En exploitant des algorithmes capables de traiter d'énormes quantités de données, il est possible de développer des systèmes prédictifs fiables et efficaces qui non seulement améliorent la rapidité et la précision des diagnostics, mais qui sont également accessibles à plus grande échelle.

Le projet "Breast Cancer Prediction App" a été conçu dans cette optique, avec l'ambition de fournir un outil d'aide à la décision qui permet de prédire, à partir de données cliniques, le coût du

traitement, le type de tumeur et le stade du cancer. Ce projet met en œuvre des techniques avancées de machine learning pour créer un modèle performant et utile pour la communauté médicale.

6 III. Objectifs

Le projet "Application de prédiction du cancer du sein" a pour objectif de développer plusieurs modèles prédictifs basés sur des techniques de machine learning afin de fournir des informations clés pour la gestion et le traitement du cancer du sein. Les objectifs spécifiques de ce projet sont les suivants :

1. **Prédiction des coûts de traitement**
   Développer un modèle de régression linéaire pour estimer les coûts de traitement des patientes atteintes de cancer du sein, en utilisant des données cliniques et démographiques. Ce modèle permettra d'anticiper les frais médicaux et d'aider à la planification des ressources.
2. **Prédiction du type de tumeur**
   Créer un second modèle de régression linéaire pour prédire le type de tumeur (bénigne ou maligne) en fonction des caractéristiques biologiques et cliniques des patientes. Cette information est essentielle pour orienter les stratégies de traitement.
3. **Classification du stade du cancer**
   Utiliser un modèle de classification basé sur les forêts aléatoires (Random Forest Classifier) pour déterminer le stade du cancer chez les patientes. Ce modèle classera les patientes selon les différents stades du cancer (précoce, avancé, etc.), fournissant ainsi un soutien crucial aux médecins dans la prise de décisions thérapeutiques.

Ces objectifs permettront de développer une application complète qui offrira une aide précieuse aux professionnels de santé en améliorant la précision des diagnostics et en optimisant les décisions médicales concernant le traitement du cancer du sein.

7 IV. Méthodologie

La méthodologie adoptée dans ce projet repose sur plusieurs étapes clés qui permettent de garantir la qualité et la précision des résultats obtenus. La première étape consiste en la **collecte des données**, où des informations pertinentes sur les patientes, telles que des caractéristiques cliniques et biologiques, sont rassemblées à partir de bases de données médicales existantes. Ensuite, un **pré-traitement des données** est effectué pour nettoyer, normaliser et préparer les données pour l'analyse, en éliminant les valeurs aberrantes et en traitant les données manquantes. Une fois les données prêtes, nous procédons à une **analyse exploratoire** pour comprendre les relations et tendances sous-jacentes. Enfin, nous appliquons des modèles de **machine learning**, notamment des modèles de régression linéaire pour prédire les coûts de traitement et le type de tumeur, ainsi qu'un classificateur Random Forest pour déterminer le stade du cancer. Cette approche permet d'obtenir des résultats fiables et exploitables, facilitant ainsi la prise de décision médicale.

7.1 *IV.1. Collecte des données*

Pour ce projet, nous avons utilisé une base de données publique provenant de **Kaggle**, qui contient des informations détaillées sur les caractéristiques cliniques des patientes atteintes de cancer du sein. Cette base de données a été collectée aux **États-Unis**, offrant ainsi un échantillon représentatif

des cas diagnostiqués dans ce pays. Les données comprennent plusieurs variables, telles que la **Clump Thickness**, la **Uniformité de la Taille des Cellules**, la **Uniformité de la Forme des Cellules**, l'**Adhésion Marginale**, la **Taille des Cellules Épithéliales Individuelles**, les **Noyaux Nus**, le **Chromatin Bland**, les **Nucleoles Normaux**, et les **Mitoses**. De plus, la base inclut des informations importantes pour la classification du cancer, telles que le **type de tumeur (Class)** et le **stade du cancer (Stage of Cancer)**. Ces données sont essentielles pour la construction des modèles de prédiction du coût du traitement, du type de tumeur, ainsi que du stade du cancer, permettant ainsi une analyse approfondie et la mise au point de solutions de diagnostic précises.

7.2 *IV.2. Pré-traitement et Analyse des données*

Au cours de cette étape, un pré-traitement minutieux sera réalisé sur la base de données pour la préparer à l'analyse. Nous diviserons la base en deux sous-ensembles distincts, chacun correspondant à un objectif spécifique du projet. Le premier sous-ensemble sera consacré à la prédiction du **type de tumeur**, en utilisant des caractéristiques telles que la **Clump Thickness**, l'**Uniformité des Cellules**, et l'**Adhésion Marginale**, etc. Le second sous-ensemble sera utilisé pour prédire le **coût de traitement** et le **stade du cancer**, en s'appuyant sur des variables relatives au **type de tumeur** et des informations sur le **stade clinique** des patientes. Cette séparation permettra d'optimiser les modèles pour chaque tâche spécifique. Après cette division, un processus de nettoyage des données sera entrepris, comprenant la gestion des valeurs manquantes et la normalisation des variables. Ces étapes permettront d'assurer la qualité des données et leur cohérence avant leur utilisation dans les modèles de prédiction.

7.2.1 *IV.2.1. Tumor type*

Le **type de tumeur** fait référence à la classification des tumeurs en fonction de leurs caractéristiques morphologiques et biologiques. Dans le cadre de ce projet, nous nous intéressons à deux types principaux de tumeurs mammaires : les tumeurs bénignes et malignes. Les tumeurs bénignes sont généralement non cancéreuses et ont une croissance lente, tandis que les tumeurs malignes sont cancéreuses et peuvent se propager à d'autres parties du corps. Le **type de tumeur** est un indicateur clé dans le diagnostic et le pronostic du cancer du sein, car il aide à déterminer les traitements appropriés et l'approche thérapeutique à adopter. L'analyse de ce facteur joue un rôle central dans la prédiction du stade du cancer et du coût du traitement, deux aspects importants dans la gestion de la maladie.

Importer les bibliothèques nécessaires.

[2]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
import os
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

[3]:
```
os.getcwd()
```

[3]:
```
'c:\\Users\\arago\\Documents\\Study\\Projects\\Cancer_Predection_App'
```

Importer la base de données

[4]:
```
data = pd.read_csv('breast_cancer_bd.csv')
```

About this Dataset

This breast cancer database was obtained from the University of Wisconsin Hospitals, Madison from Dr. William H. Wolberg.

Attributes

Attributes 1 through 10 have been used to represent instances. Each instance has one of 2 possible classes: benign or malignant.

Content

Attribute	Domain
1. **Sample code number**	id number
2. **Clump Thickness**	1 - 10
3. **Uniformity of Cell Size**	1 - 10
4. **Uniformity of Cell Shape**	1 - 10
5. **Marginal Adhesion**	1 - 10
6. **Single Epithelial Cell Size**	1 - 10
7. **Bare Nuclei**	1 - 10
8. **Bland Chromatin**	1 - 10
9. **Normal Nucleoli**	1 - 10
10. **Mitoses**	1 - 10
11. **Class** (2 for benign, 4 for malignant)	

Class Distribution

- **Benign:** 458 (65.5%)
- **Malignant:** 241 (34.5%)

[5]:
```
data.head()
```

[5]:
```
   Sample code number  Clump Thickness  Uniformity of Cell Size  \
0             1000025                5                        1
1             1002945                5                        4
2             1015425                3                        1
3             1016277                6                        8
4             1017023                4                        1

   Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial Cell Size  \
0                         1                  1                            2
1                         4                  5                            7
2                         1                  1                            2
3                         8                  1                            3
```

```
4                          1                   3                                2
```

```
   Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Class
0            1                3                1        1      2
1           10                3                2        1      2
2            2                3                1        1      2
3            4                3                7        1      2
4            1                3                1        1      2
```

Description des Caractéristiques Cellulaires

- **Épaisseur des Amas (Clump Thickness)** : Fait référence à l'épaisseur des groupes de cellules. Des valeurs plus élevées peuvent indiquer la présence de cellules cancéreuses car elles ont tendance à former des amas.
- **Uniformité de la Taille Cellulaire (Uniformity of Cell Size)** : Les cellules cancéreuses varient souvent en taille. Cette caractéristique mesure l'homogénéité de la taille des cellules.
- **Uniformité de la Forme Cellulaire (Uniformity of Cell Shape)** : Comme pour la taille, les cellules cancéreuses varient souvent en forme. Cette caractéristique évalue la cohérence dans la forme des cellules.
- **Adhésion Marginale (Marginal Adhesion)** : Dans les tissus sains, les cellules adhèrent les unes aux autres. Une diminution de l'adhésion peut être un signe de malignité.
- **Taille des Cellules Épithéliales Individuelles (Single Epithelial Cell Size)** : Mesure la taille des cellules épithéliales individuelles, qui ont tendance à être plus grandes en présence de cancer.
- **Noyaux Nus (Bare Nuclei)** : Se réfère à la présence de noyaux sans le cytoplasme environnant. Les noyaux nus sont plus fréquemment trouvés dans les tumeurs malignes.
- **Chromatine Pâle (Bland Chromatin)** : La chromatine est le matériau qui compose les chromosomes. Dans les cellules cancéreuses, la chromatine peut apparaître moins structurée ou plus uniforme.
- **Nucléoles Normaux (Normal Nucleoli)** : Les nucléoles sont de petites structures denses à l'intérieur du noyau. Des nucléoles plus grands ou plus nombreux sont souvent un signe de cancer.
- **Mitoses** : Fait référence à la division cellulaire. Un nombre plus élevé de mitoses indique généralement une croissance cellulaire rapide, ce qui peut signaler un cancer.

Nettoyage et préparation des données

[ ]:
```
print("les diemsions de la BDD : ", data.shape)
print("\n\nla liste des variables : \n", data.columns)
print("\n\nles types de variables : \n", data.dtypes)
```

```
les diemsions de la BDD :  (699, 12)


la liste des variables :
```

```
Index(['Sample code number', 'Clump Thickness', 'Uniformity of Cell Size',
       'Uniformity of Cell Shape', 'Marginal Adhesion',
       'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland Chromatin',
       'Normal Nucleoli', 'Mitoses', 'Stage of Cancer', 'Charge'],
      dtype='object')
```

```
les types de variables :
 Sample code number               int64
Clump Thickness                   int64
Uniformity of Cell Size           int64
Uniformity of Cell Shape          int64
Marginal Adhesion                 int64
Single Epithelial Cell Size       int64
Bare Nuclei                      object
Bland Chromatin                   int64
Normal Nucleoli                   int64
Mitoses                           int64
Stage of Cancer                   int64
Charge                          float64
dtype: object
```

[ ]:
```
data = data.drop(['Sample code number'], axis=1)
```

[ ]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 699 entries, 0 to 698
Data columns (total 10 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Clump Thickness              699 non-null    int64
 1   Uniformity of Cell Size      699 non-null    int64
 2   Uniformity of Cell Shape     699 non-null    int64
 3   Marginal Adhesion            699 non-null    int64
 4   Single Epithelial Cell Size  699 non-null    int64
 5   Bare Nuclei                  699 non-null    object
 6   Bland Chromatin              699 non-null    int64
 7   Normal Nucleoli              699 non-null    int64
 8   Mitoses                      699 non-null    int64
 9   Class                        699 non-null    int64
dtypes: int64(9), object(1)
memory usage: 54.7+ KB
```

[ ]:
```
data.describe()
```

```
       Clump Thickness  Uniformity of Cell Size  Uniformity of Cell Shape  \
count       699.000000               699.000000                699.000000
mean          4.417740                 3.134478                  3.207439
```

```
std           2.815741                3.051459                    2.971913
min           1.000000                1.000000                    1.000000
25%           2.000000                1.000000                    1.000000
50%           4.000000                1.000000                    1.000000
75%           6.000000                5.000000                    5.000000
max          10.000000               10.000000                   10.000000

       Marginal Adhesion  Single Epithelial Cell Size  Bland Chromatin  \
count         699.000000                   699.000000       699.000000
mean            2.806867                     3.216023         3.437768
std             2.855379                     2.214300         2.438364
min             1.000000                     1.000000         1.000000
25%             1.000000                     2.000000         2.000000
50%             1.000000                     2.000000         3.000000
75%             4.000000                     4.000000         5.000000
max            10.000000                    10.000000        10.000000

       Normal Nucleoli     Mitoses       Class
count       699.000000  699.000000  699.000000
mean          2.866953    1.589413    2.689557
std           3.053634    1.715078    0.951273
min           1.000000    1.000000    2.000000
25%           1.000000    1.000000    2.000000
50%           1.000000    1.000000    2.000000
75%           4.000000    1.000000    4.000000
max          10.000000   10.000000    4.000000
```

[ ]:
```
print(data['Bare Nuclei'].unique())
```

```
['1' '10' '2' '4' '3' '9' '7' '?' '5' '8' '6']
```

[ ]:
```
data = data.replace('?', np.nan)

data['Bare Nuclei'] = pd.to_numeric(data['Bare Nuclei'])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 699 entries, 0 to 698
Data columns (total 10 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Clump Thickness              699 non-null    int64
 1   Uniformity of Cell Size      699 non-null    int64
 2   Uniformity of Cell Shape     699 non-null    int64
 3   Marginal Adhesion            699 non-null    int64
 4   Single Epithelial Cell Size  699 non-null    int64
 5   Bare Nuclei                  683 non-null    float64
 6   Bland Chromatin              699 non-null    int64
 7   Normal Nucleoli              699 non-null    int64
```

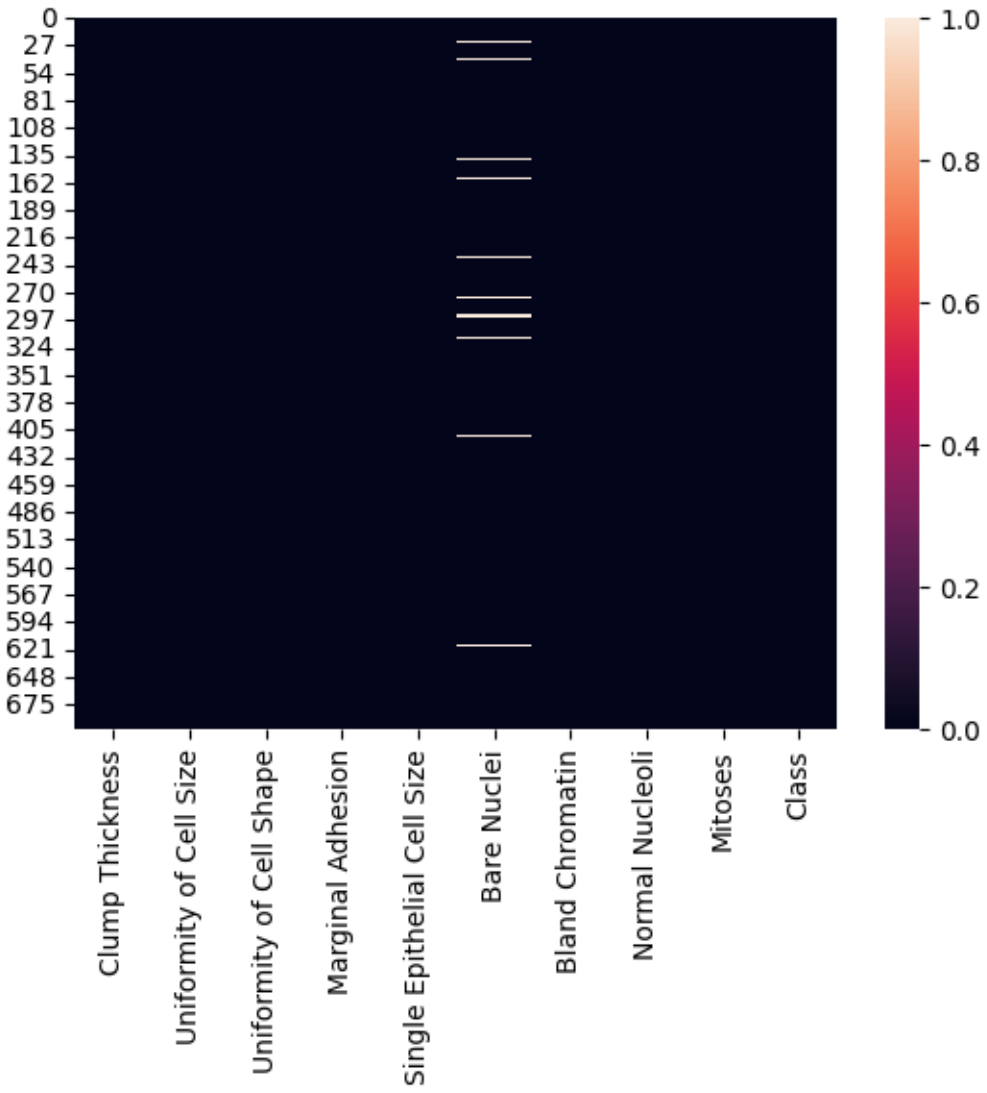
```
 8   Mitoses                      699 non-null    int64
 9   Class                        699 non-null    int64
dtypes: float64(1), int64(9)
memory usage: 54.7 KB
```

[ ]:
```
print(data.isnull().sum())
```

```
Clump Thickness               0
Uniformity of Cell Size       0
Uniformity of Cell Shape      0
Marginal Adhesion             0
Single Epithelial Cell Size   0
Bare Nuclei                  16
Bland Chromatin               0
Normal Nucleoli               0
Mitoses                       0
Class                         0
dtype: int64
```

[ ]:
```
sns.heatmap(data.isnull())
```

```
<Axes: >
```



[ ]:
```
data['Bare Nuclei'] = data['Bare Nuclei'].fillna(data['Bare Nuclei'].median())
```

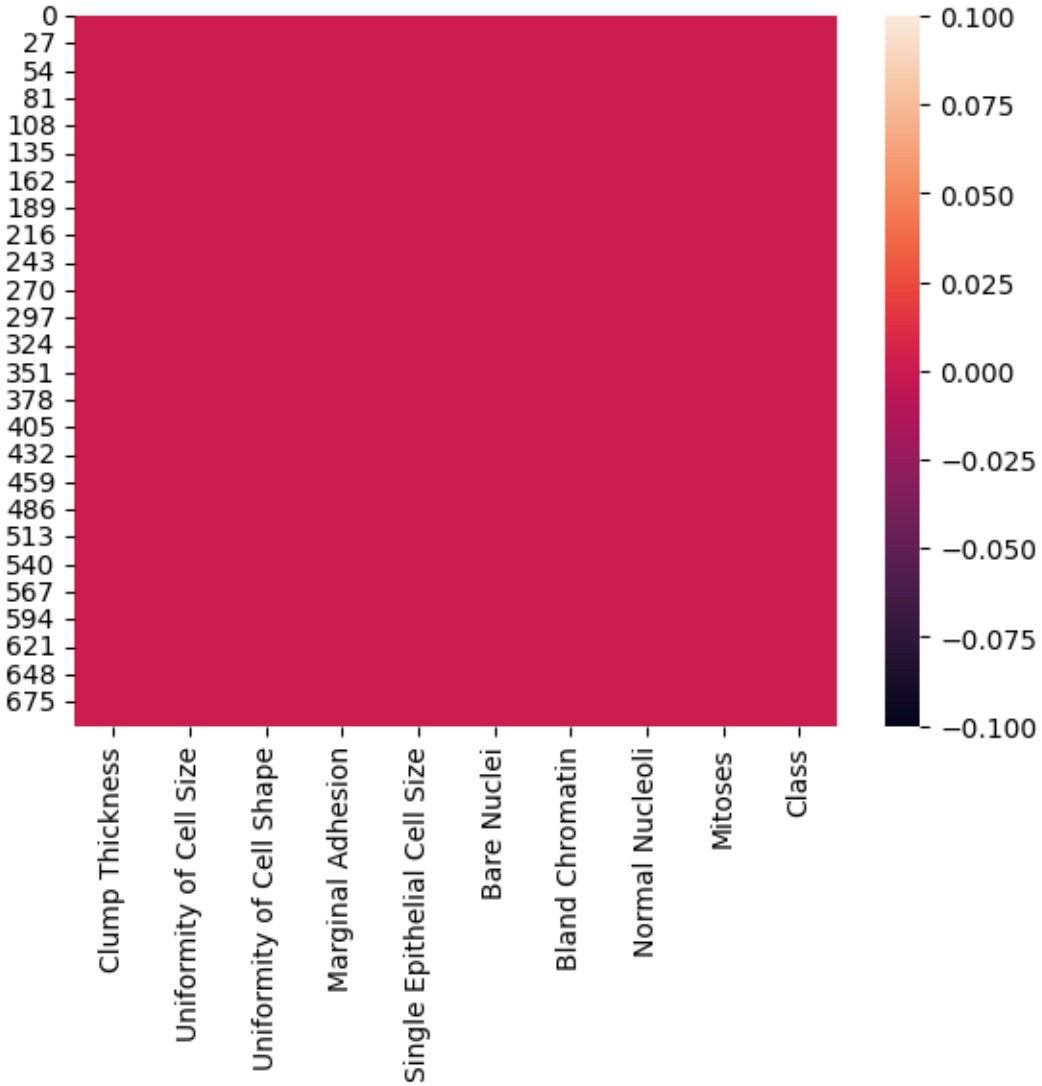
[ ]:
```
print(data.isnull().sum())
```

```
Clump Thickness               0
Uniformity of Cell Size       0
Uniformity of Cell Shape      0
Marginal Adhesion             0
Single Epithelial Cell Size   0
Bare Nuclei                   0
Bland Chromatin               0
Normal Nucleoli               0
Mitoses                       0
Class                         0
```

```
dtype: int64
```

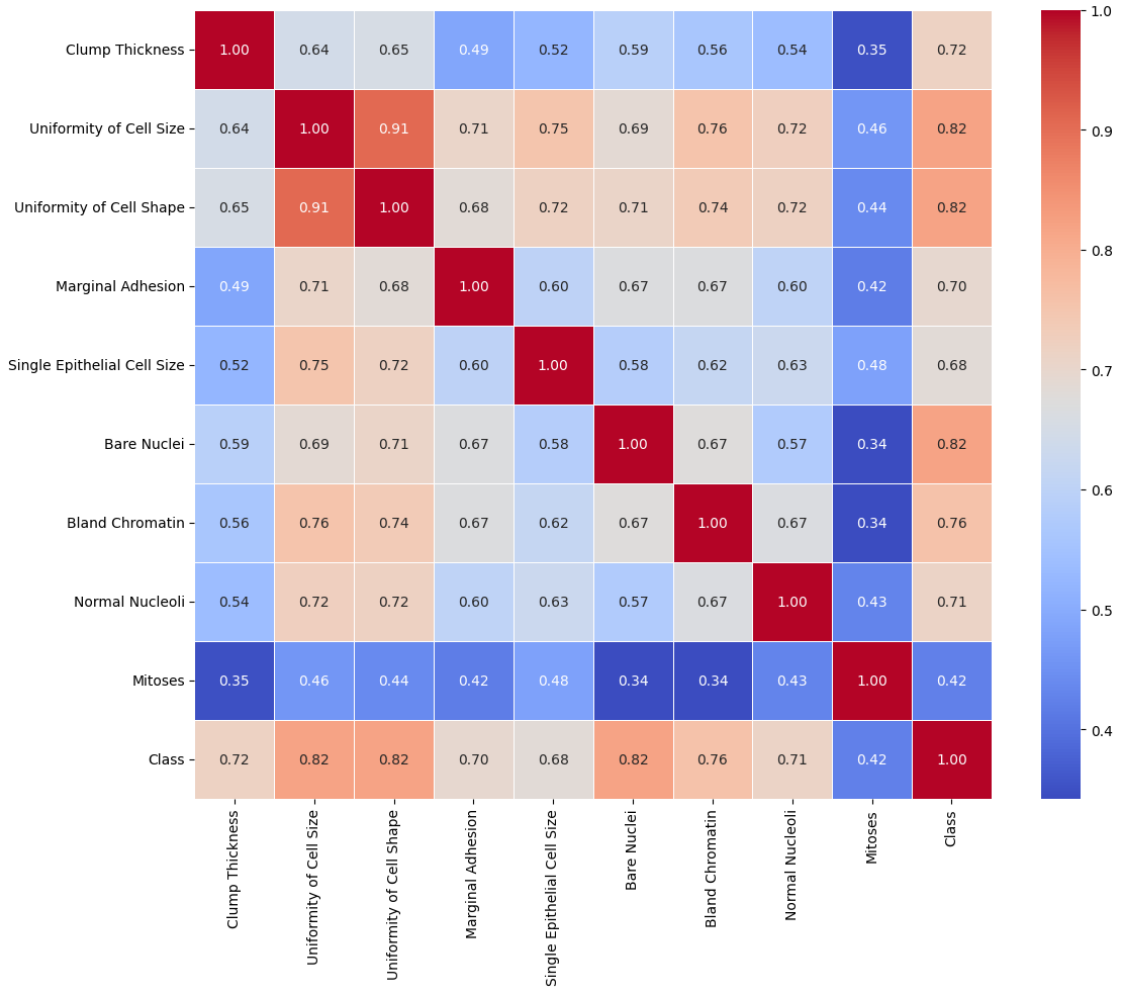
[ ]:
```
sns.heatmap(data.isnull())
```

```
<Axes: >
```



[ ]:
```
# Correlation matrix
correlation_matrix = data.corr()

# Plot heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',␣
 ↪linewidths=0.5)
plt.tight_layout()
```

```
plt.show()
```



8 Interprétation de la Matrice de Corrélation

La matrice de corrélation fournit des aperçus sur les relations entre différents attributs liés aux données du cancer du sein. Chaque valeur dans la matrice indique la force et la direction de la relation linéaire entre les paires d'attributs, allant de -1 à 1. Une valeur proche de 1 implique une forte corrélation positive, tandis qu'une valeur proche de -1 implique une forte corrélation négative. Une valeur autour de 0 indique une absence de corrélation.

8.1 Observations Principales

- **Épaisseur des Amas** :
  - **Corrélation la plus élevée** : Classe (0,716)
  - Cela suggère que lorsque l'épaisseur des amas augmente, la probabilité d'une classe maligne augmente également.

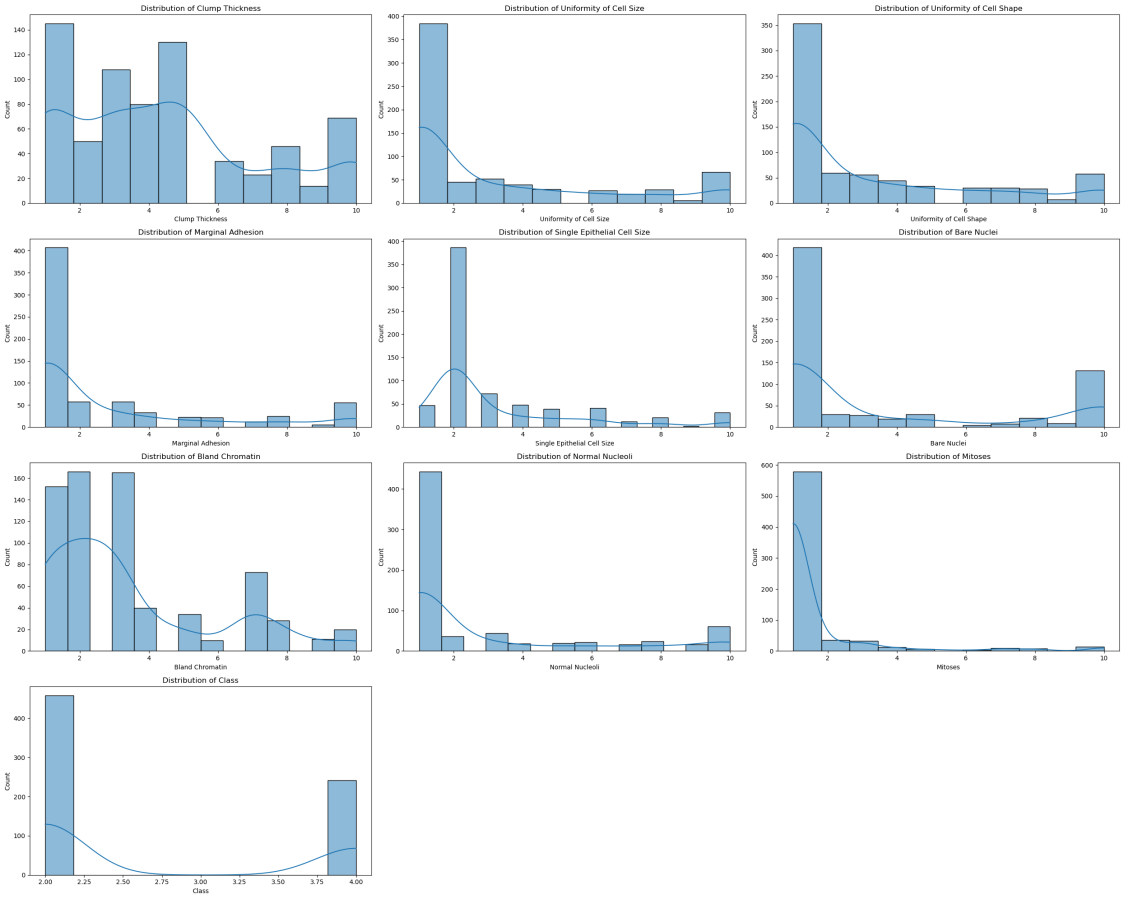
- **Uniformité de la Taille Cellulaire** :
  - **Corrélation la plus élevée** : Uniformité de la Forme Cellulaire (0,907)
  - Cela indique une relation très forte entre l'uniformité de la taille des cellules et l'uniformité de leur forme, suggérant que les changements de l'une peuvent refléter des changements de l'autre.
- **Uniformité de la Forme Cellulaire** :
  - **Corrélation la plus élevée** : Uniformité de la Taille Cellulaire (0,907)
  - Similaire à ci-dessus, la relation avec la taille cellulaire renforce l'importance de l'uniformité dans les deux dimensions pour évaluer le risque de malignité.
- **Adhésion Marginale** :
  - **Corrélation la plus élevée** : Taille des Cellules Épithéliales Individuelles (0,600)
  - Cela indique qu'une adhésion plus faible peut être corrélée avec des tailles de cellules épithéliales plus grandes, ce qui peut être un marqueur de malignité.
- **Noyaux Nus** :
  - **Corrélation la plus élevée** : Classe (0,819)
  - La forte corrélation implique que la présence de noyaux nus est étroitement liée aux classifications malignes, indiquant leur importance potentielle dans le diagnostic.
- **Chromatine Pâle** :
  - **Corrélation la plus élevée** : Classe (0,757)
  - Similaire aux noyaux nus, la chromatine pâle est également un indicateur significatif de la probabilité de malignité.
- **Nucléoles Normaux** :
  - **Corrélation la plus élevée** : Classe (0,712)
  - La présence de nucléoles normaux est une autre caractéristique pertinente qui indique le potentiel de tumeurs malignes.
- **Mitoses** :
  - **Corrélation avec la Classe** : (0,423)
  - Bien qu'il y ait une corrélation positive, elle est relativement plus faible par rapport aux autres attributs, indiquant que l'activité mitotique n'est pas un indicateur aussi fort de malignité que les autres caractéristiques.

8.2 Conclusion

Dans l'ensemble, les attributs comme les **Noyaux Nus**, la **Classe**, et la **Chromatine Pâle** montrent les corrélations les plus fortes, suggérant que ces caractéristiques sont cruciales pour distinguer les cas bénins des cas malins. Les fortes corrélations entre les mesures d'uniformité soulignent davantage l'importance d'évaluer ces caractéristiques ensemble lors de l'analyse des données sur le cancer du sein.

[ ]:
```
plt.figure(figsize=(25, 20))
num_columns = len(data.columns)
rows = (num_columns // 3) + (num_columns % 3 > 0)  # Calculate rows needed for␣
 ↪the columns
for i, col in enumerate(data.columns, 1):
    plt.subplot(rows, 3, i)  # Adjust the number of rows dynamically
    sns.histplot(data[col], kde=True)
    plt.title(f'Distribution of {col}')
plt.tight_layout()
```

```
plt.show()
```



9 Analyse des Caractéristiques Cellulaires

9.1 Analyse Détaillée des Paramètres

9.1.1 1. Morphologie Cellulaire de Base

Épaisseur des Amas Cellulaires

- Distribution présentant plusieurs modes
- Concentration principale entre les valeurs 2-6
- Deux pics distincts : à 2 et 4-5
- Suggère différents profils de croissance cellulaire

Paramètres d'Uniformité **Taille Cellulaire** - Distribution asymétrique marquée - Forte concentration dans la zone 1-3 - Extension progressive vers les valeurs supérieures - Indicateur potentiel d'anomalies de croissance

Forme Cellulaire - Profil similaire à la taille - Concentration majeure dans les valeurs faibles - Diminution progressive vers les valeurs élevées - Reflète la régularité morphologique

9.1.2 2. Caractéristiques de Surface et Structure

Adhésion Cellulaire

- Distribution déséquilibrée vers la droite
- Pic important autour de 1
- Rareté des valeurs élevées
- Indicateur de cohésion tissulaire

Cellules Épithéliales

- Concentration maximale vers 2
- Profil asymétrique caractéristique
- Faible occurrence au-delà de 4
- Marqueur de différenciation cellulaire

9.1.3 3. Caractéristiques Nucléaires

Noyaux Dénudés

- Structure bimodale distinctive
- Premier pic dominant dans les valeurs basses
- Second pic mineur dans les valeurs hautes
- Possible indicateur de malignité

Aspects Chromatiniens

- Distribution complexe multi-pics
- Concentrations notables entre 1-3
- Variation significative
- Reflète l'hétérogénéité nucléaire

Nucléoles

- Asymétrie prononcée
- Pic principal dans les valeurs faibles
- Décroissance marquée
- Marqueur d'activité cellulaire

9.1.4 4. Activité Cellulaire

Indice Mitotique

- Asymétrie extrême
- Dominance près de 1
- Cas exceptionnels de valeurs élevées
- Indicateur de prolifération

9.2 Synthèse Globale

9.2.1 Tendances Principales

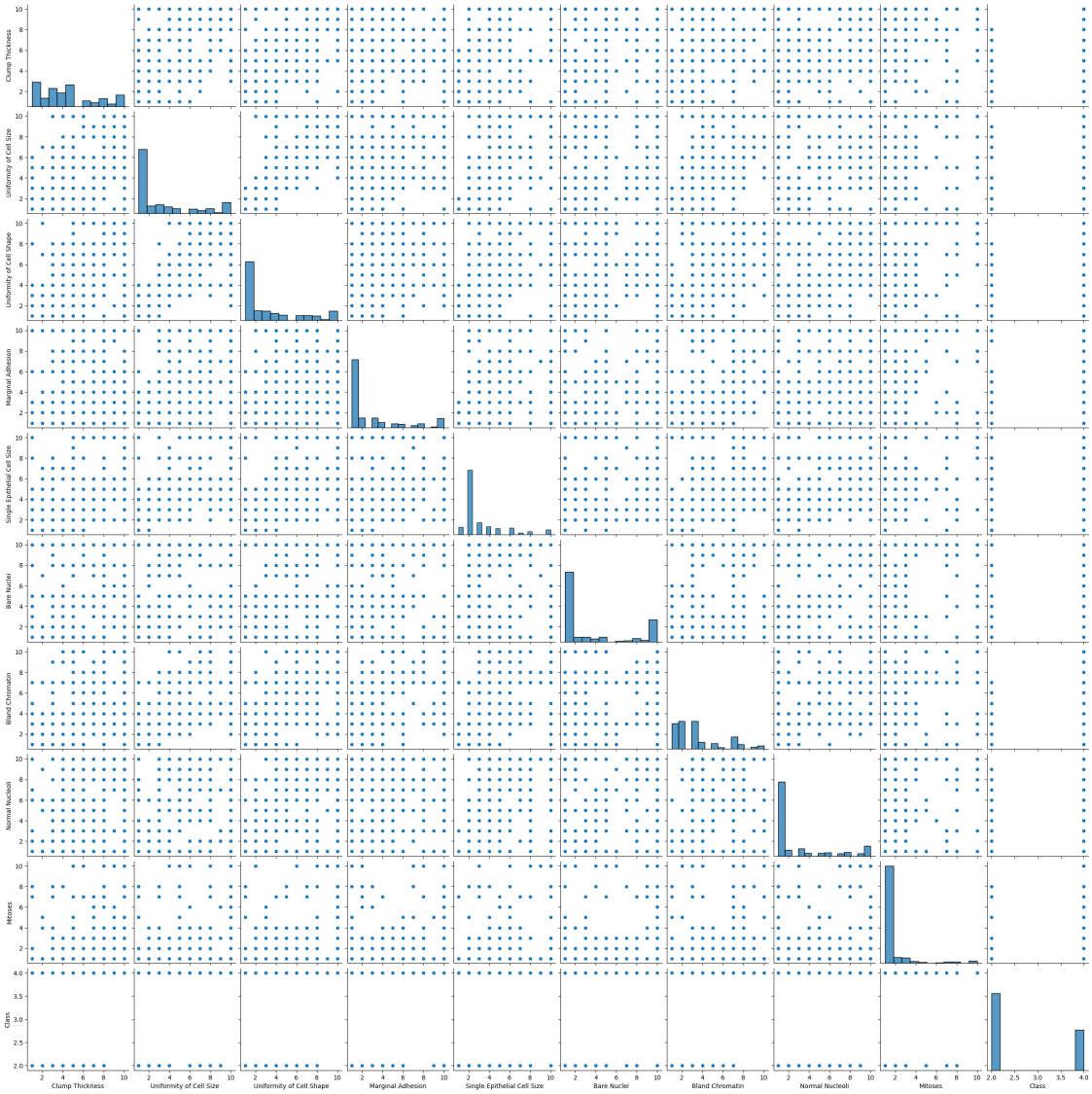
- Asymétrie positive dominante

- Prédominance des valeurs basses
- Complexité variable selon les paramètres
- Distinction claire entre catégories pathologiques

9.2.2 Implications Diagnostiques

- Paramètres permettant une discrimination efficace
- Complémentarité des indicateurs
- Base solide pour la classification diagnostique

[ ]:
```
sns.pairplot(data)
plt.show()
```



10 Analyse de la Matrice de Corrélation et des Distributions de Variables

10.1 Structure du Graphique

Cette visualisation présente une matrice de diagrammes de dispersion (scatter plots) avec les histogrammes de distribution sur la diagonale. Elle montre les relations entre toutes les variables du jeu de données sur le cancer du sein.

10.2 Analyse des Distributions (Diagonale)

10.2.1 Caractéristiques Morphologiques

1. **Épaisseur des Amas (Clump Thickness)**

- Distribution bimodale
- Pics autour des valeurs 2-3 et 7-8
- Suggère une séparation naturelle entre les classes

2. **Uniformité (Cell Size et Cell Shape)**

- Distributions très asymétriques
- Fort pic à des valeurs basses
- Queue longue vers les valeurs élevées

10.2.2 Caractéristiques Cellulaires

3. **Adhésion Marginale**

- Distribution fortement asymétrique
- Majorité des cas avec valeurs faibles
- Quelques cas extrêmes à valeurs élevées

4. **Taille des Cellules Épithéliales**

- Distribution similaire à l'adhésion marginale
- Concentration forte sur les valeurs basses

10.2.3 Caractéristiques Nucléaires

5. **Noyaux Nus**

- Distribution multimodale
- Pics distincts suggérant des sous-groupes

6. **Chromatine**

- Distribution étalée
- Plusieurs pics mineurs

7. **Nucléoles**

- Fort pic à des valeurs basses
- Distribution décroissante

8. **Mitoses**

- Distribution très asymétrique
- Majorité des cas avec peu de mitoses

10.3 Corrélations Observées

10.3.1 Corrélations Fortes (> 0.7)

- Uniformité de taille/forme cellulaire
- Noyaux nus/Classification
- Chromatine/Classification

10.3.2 Corrélations Modérées (0.4-0.7)

- Épaisseur des amas/autres caractéristiques
- Adhésion marginale/taille cellulaire
- Mitoses/Classification

10.4 Implications Cliniques

1. **Pour le Diagnostic**

- Les caractéristiques nucléaires sont les plus discriminantes
- L'uniformité cellulaire est un indicateur important
- Les mitoses sont moins prédictives seules

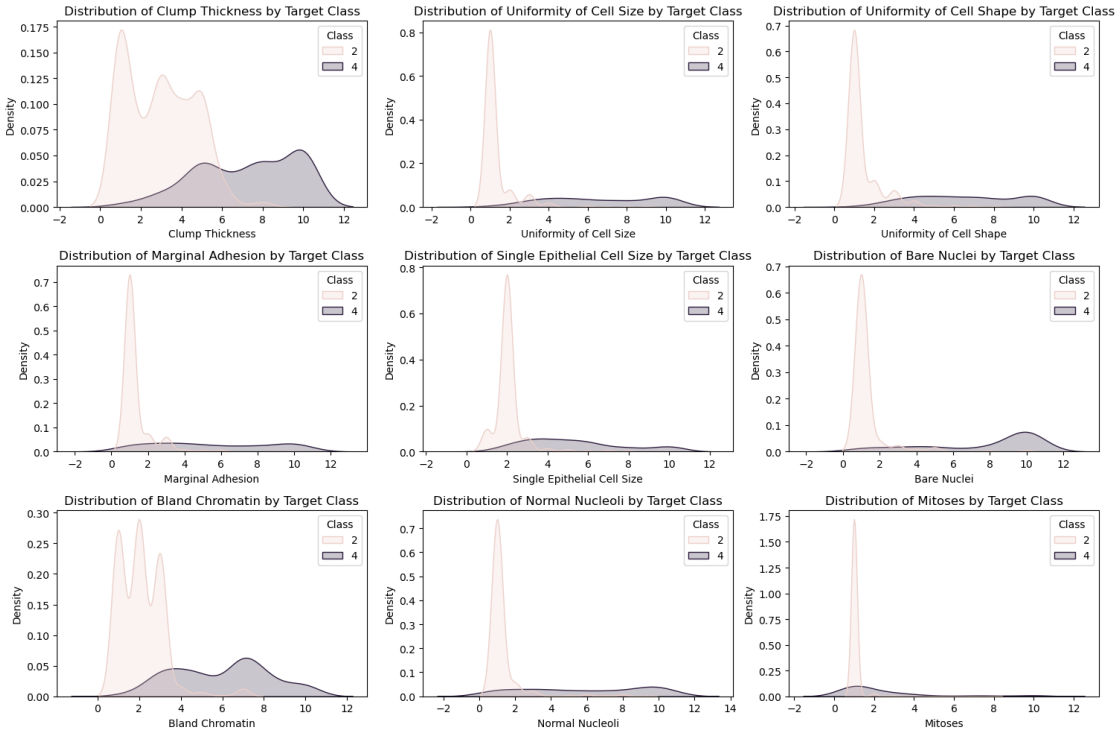
2. **Pour la Classification**

- Plusieurs variables montrent une bonne séparation des classes
- Combinaison de caractéristiques améliore la précision
- Les distributions suggèrent des seuils naturels

10.5 Conclusion

La visualisation révèle des patterns clairs de corrélation entre les variables et montre que certaines caractéristiques sont plus discriminantes que d'autres pour la classification des cas bénins et malins. Les distributions suggèrent des seuils naturels qui pourraient être utilisés dans les algorithmes de diagnostic.

[ ]:
```
plt.figure(figsize=(15, 10))
for i, col in enumerate(data.columns[:-1], 1):
    plt.subplot(3, 3, i)
    sns.kdeplot(data=data, x=col, hue='Class', fill=True)
    plt.title(f'Distribution of {col} by Target Class')
plt.tight_layout()
plt.show()
```



11 Analyse des Distributions de Caractéristiques Cellulaires par Classe

11.1 Analyse des Caractéristiques Principales

11.1.1 Caractéristiques Structurelles

1. Épaisseur des Amas

- **Classe 2 :** Multiples pics dans la plage 0-4
- **Classe 4 :** Distribution plus large, centrée 6-10
- *Notable :* Forte séparation entre les classes

2. Caractéristiques d'Uniformité Cellulaire **Uniformité de la Taille Cellulaire :** - **Classe 2 :** Pic prononcé près de 0-1 - **Classe 4 :** Distribution plate sur toute la plage

Uniformité de la Forme Cellulaire : - **Classe 2 :** Similaire à l'uniformité de taille - **Classe 4 :** Distribution dispersée - *Notable :* Les deux mesures d'uniformité montrent des motifs similaires

11.1.2 Propriétés Cellulaires

3. Adhésion Marginale

- **Classe 2 :** Pic prononcé à 1-2
- **Classe 4 :** Distribution basse et uniforme
- *Notable :* Différenciation claire entre les classes

4. Taille des Cellules Épithéliales Individuelles

- **Classe 2 :** Concentration aux valeurs basses
- **Classe 4 :** Multiples petits pics sur la plage
- *Notable :* Motif similaire à l'adhésion marginale

11.1.3 Caractéristiques Nucléaires

5. Noyaux Nus

- **Classe 2 :** Concentration de valeurs basses
- **Classe 4 :** Pic distinct à 8-10
- *Notable :* Excellente séparation des classes

6. Chromatine Pâle

- **Classe 2 :** Multiples pics entre 0-4
- **Classe 4 :** Pic plus large autour de 6-8
- *Notable :* Distinction claire entre les classes

7. Nucléoles Normaux

- **Classe 2 :** Pic prononcé près de 1
- **Classe 4 :** Distribution basse et dispersée
- *Notable :* Caractéristique fortement discriminante

8. Mitoses

- **Classe 2 :** Concentration près de 1
- **Classe 4 :** Valeurs légèrement plus élevées mais faible occurrence
- *Notable :* Séparation moins prononcée entre les classes

11.2 Tendances Générales

11.2.1 Caractéristiques des Classes

- **Classe 2 :**
  - Distributions concentrées
  - Généralement des valeurs plus basses
  - Pics nets et définis
- **Classe 4 :**
  - Distributions plus larges
  - Tendance aux valeurs plus élevées
  - Motifs plus dispersés

11.2.2 Caractéristiques les Plus Discriminantes

1. Épaisseur des Amas
2. Noyaux Nus
3. Chromatine Pâle

11.2.3 Implications pour la Classification

- Les caractéristiques montrent un fort potentiel de séparation
- Multiples caractéristiques discriminantes indépendantes
- Adapté pour les modèles de classification diagnostique

11.2.4 *IV.2.2. Stage of cancer and Treatment cost*

Le **stade du cancer** et le **coût du traitement** sont deux facteurs essentiels dans le cadre de la gestion du cancer du sein. Le **stade du cancer** fait référence à l'étendue de la propagation de la maladie, en fonction de la taille de la tumeur et de son envahissement des ganglions lymphatiques ou d'autres parties du corps. Il est classé en plusieurs stades, allant de **0 (cancer in situ)** à **IV (cancer métastatique)**. Cette information est cruciale pour déterminer le traitement approprié et le pronostic.

Le **coût du traitement** est une autre variable clé, qui dépend des interventions nécessaires à chaque stade de la maladie. Les traitements pour le cancer du sein incluent généralement la chirurgie, la chimiothérapie, la radiothérapie et, dans certains cas, la thérapie ciblée ou l'immunothérapie. Le coût varie en fonction du type de traitement, de la durée du suivi médical, et des soins requis pour la récupération du patient. Dans le cadre de ce projet, l'objectif est de prédire ces deux variables, à savoir le **stade du cancer** et le **coût des traitements**, afin d'aider à planifier les soins de manière plus efficace et de soutenir la prise de décision médicale.

Importer les bibliothèques nécessaires.

[ ]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import joblib
import os
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
```

[ ]:
```
data = pd.read_csv('updated_breast_cancer.csv')
```

[ ]:
```
data.head()
```

```
   Sample code number  Clump Thickness  Uniformity of Cell Size  \
0             1000025                5                        1
1             1002945                5                        4
2             1015425                3                        1
3             1016277                6                        8
4             1017023                4                        1

   Uniformity of Cell Shape  Marginal Adhesion  Single Epithelial Cell Size  \
0                         1                  1                            2
1                         4                  5                            7
```

```
2                         1                 1                           2
3                         8                 1                           3
4                         1                 3                           2

  Bare Nuclei  Bland Chromatin  Normal Nucleoli  Mitoses  Stage of Cancer  \
0           1                3                1        1                1
1          10                3                2        1                3
2           2                3                1        1                1
3           4                3                7        1                3
4           1                3                1        1                1

          Charge
0   84572.935561
1  132537.676589
2   83790.378723
3  133059.521055
4   84170.762690
```

Nettoyage et préparation des données

[ ]:
```
print("les diemsions de la BDD : ", data.shape)
print("\n\nla liste des variables : \n", data.columns)
print("\n\nles types de variables : \n", data.dtypes)
```

```
les diemsions de la BDD :  (699, 12)


la liste des variables :
 Index(['Sample code number', 'Clump Thickness', 'Uniformity of Cell Size',
       'Uniformity of Cell Shape', 'Marginal Adhesion',
       'Single Epithelial Cell Size', 'Bare Nuclei', 'Bland Chromatin',
       'Normal Nucleoli', 'Mitoses', 'Stage of Cancer', 'Charge'],
      dtype='object')


les types de variables :
 Sample code number              int64
Clump Thickness                 int64
Uniformity of Cell Size         int64
Uniformity of Cell Shape        int64
Marginal Adhesion               int64
Single Epithelial Cell Size     int64
Bare Nuclei                    object
Bland Chromatin                 int64
Normal Nucleoli                 int64
Mitoses                         int64
Stage of Cancer                 int64
Charge                        float64
dtype: object
```

[ ]:
```
data = data.drop('Sample code number', axis=1)
```

[ ]:
```
data.describe()
```

```
       Clump Thickness  Uniformity of Cell Size  Uniformity of Cell Shape  \
count       699.000000               699.000000                699.000000
mean          4.417740                 3.134478                  3.207439
std           2.815741                 3.051459                  2.971913
min           1.000000                 1.000000                  1.000000
25%           2.000000                 1.000000                  1.000000
50%           4.000000                 1.000000                  1.000000
75%           6.000000                 5.000000                  5.000000
max          10.000000                10.000000                 10.000000

       Marginal Adhesion  Single Epithelial Cell Size  Bland Chromatin  \
count         699.000000                   699.000000       699.000000
mean            2.806867                     3.216023         3.437768
std             2.855379                     2.214300         2.438364
min             1.000000                     1.000000         1.000000
25%             1.000000                     2.000000         2.000000
50%             1.000000                     2.000000         3.000000
75%             4.000000                     4.000000         5.000000
max            10.000000                    10.000000        10.000000

       Normal Nucleoli     Mitoses  Stage of Cancer         Charge
count       699.000000  699.000000       699.000000     699.000000
mean          2.866953    1.589413         1.925608  107419.887942
std           3.053634    1.715078         1.616592   42586.718154
min           1.000000    1.000000         0.000000   61081.379363
25%           1.000000    1.000000         1.000000   83400.704431
50%           1.000000    1.000000         1.000000   84455.124155
75%           4.000000    1.000000         3.000000  132904.211981
max          10.000000   10.000000         5.000000  205574.798140
```

[ ]:
```
print(data['Bare Nuclei'].unique())
```

```
['1' '10' '2' '4' '3' '9' '7' '?' '5' '8' '6']
```

[ ]:
```
data = data.replace('?', np.nan)

data['Bare Nuclei'] = pd.to_numeric(data['Bare Nuclei'])
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 699 entries, 0 to 698
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   Clump Thickness              699 non-null    int64
```

```
 1   Uniformity of Cell Size      699 non-null    int64
 2   Uniformity of Cell Shape     699 non-null    int64
 3   Marginal Adhesion            699 non-null    int64
 4   Single Epithelial Cell Size  699 non-null    int64
 5   Bare Nuclei                  683 non-null    float64
 6   Bland Chromatin              699 non-null    int64
 7   Normal Nucleoli              699 non-null    int64
 8   Mitoses                      699 non-null    int64
 9   Stage of Cancer              699 non-null    int64
 10  Charge                       699 non-null    float64
dtypes: float64(2), int64(9)
memory usage: 60.2 KB
```

[ ]:
```
print(data.isnull().sum())
```

```
Clump Thickness                0
Uniformity of Cell Size        0
Uniformity of Cell Shape       0
Marginal Adhesion              0
Single Epithelial Cell Size    0
Bare Nuclei                   16
Bland Chromatin                0
Normal Nucleoli                0
Mitoses                        0
Stage of Cancer                0
Charge                         0
dtype: int64
```

[ ]:
```
data['Bare Nuclei'] = data['Bare Nuclei'].fillna(data['Bare Nuclei'].median())
```

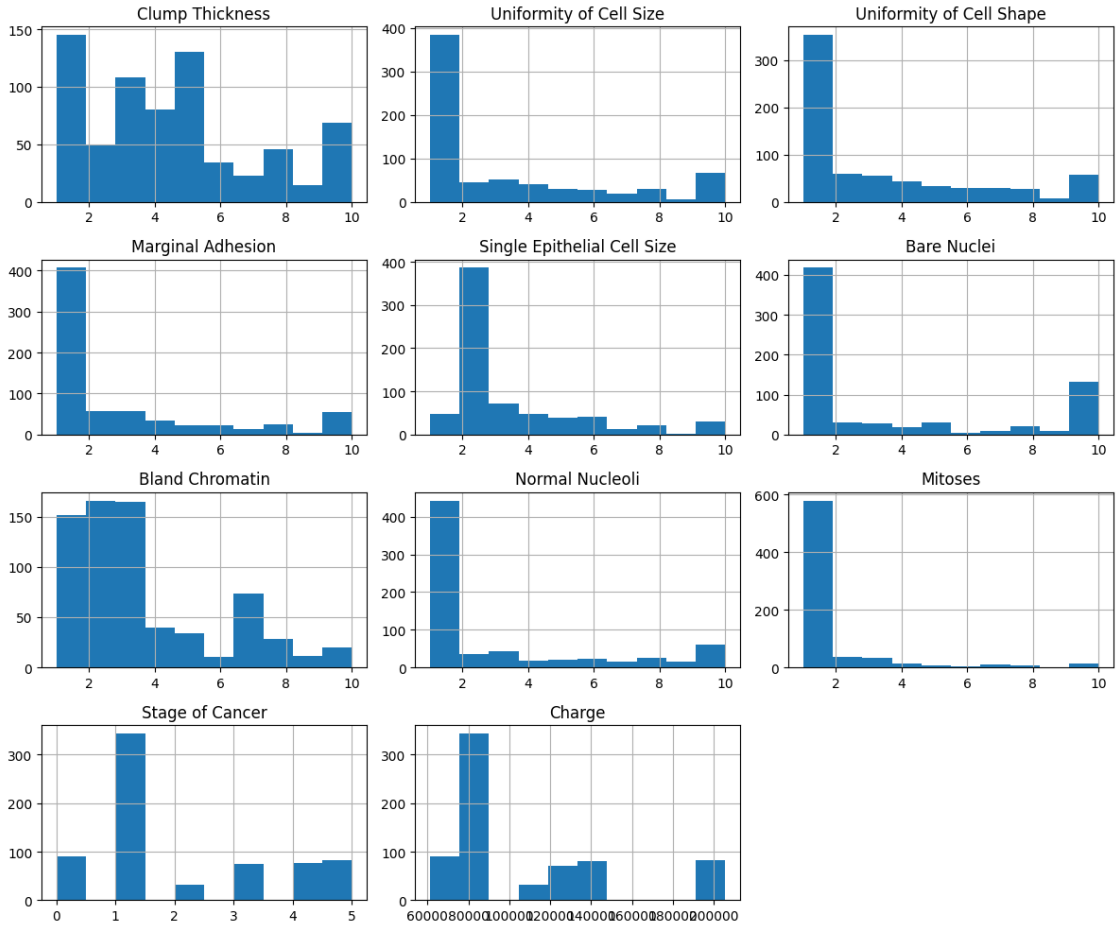
[ ]:
```
print(data.isnull().sum())
```

```
Clump Thickness               0
Uniformity of Cell Size       0
Uniformity of Cell Shape      0
Marginal Adhesion             0
Single Epithelial Cell Size   0
Bare Nuclei                   0
Bland Chromatin               0
Normal Nucleoli               0
Mitoses                       0
Stage of Cancer               0
Charge                        0
dtype: int64
```

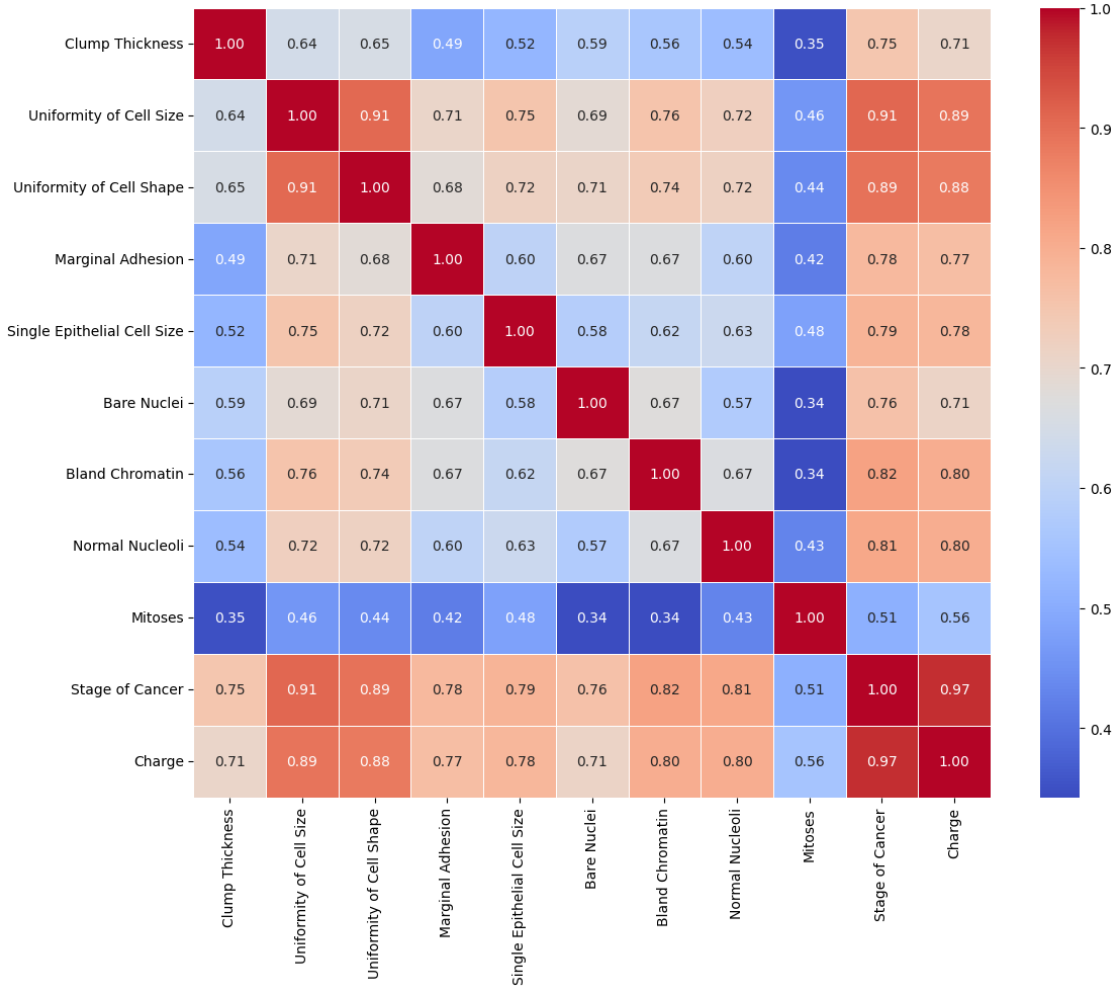
Visualisation des données

[ ]:
```
import matplotlib.pyplot as plt
```

```
# Plot histograms for each feature
data.hist(figsize=(12, 10))
plt.tight_layout()
plt.show()
```



[ ]:
```
# Correlation matrix
correlation_matrix = data.corr()

# Plot heatmap
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f',␣
 ↪linewidths=0.5)
plt.tight_layout()
plt.show()
```



11.2.5 Interprétation :

1. **Corrélations élevées** :
   - *Uniformité de la taille des cellules* et *Uniformité de la forme des cellules* présentent un coefficient de corrélation élevé de **0.91**, ce qui suggère qu'elles mesurent probablement des aspects très similaires de l'anomalie cellulaire.
   - *Stade du cancer* est fortement corrélé avec *Uniformité de la taille des cellules* (0.91), *Uniformité de la forme des cellules* (0.89) et *Charge* (0.97). Cela indique que ces caractéristiques jouent un rôle important dans la détermination du stade du cancer.
   - *Charge* montre également une forte corrélation avec *Stade du cancer* (0.97), ce qui suggère qu'elle pourrait être étroitement liée à la progression du cancer.
2. **Corrélations modérées** :
   - *Épaisseur des grappes* présente une corrélation modérée avec *Stade du cancer* (0.75) et *Charge* (0.71). Cela suggère que bien que l'épaisseur des grappes soit liée au stade du cancer, elle pourrait ne pas être un prédicteur aussi puissant que d'autres caractéristiques.
   - *Chromatine pâle* présente une corrélation modérée avec *Stade du cancer* (0.82) et *Charge*

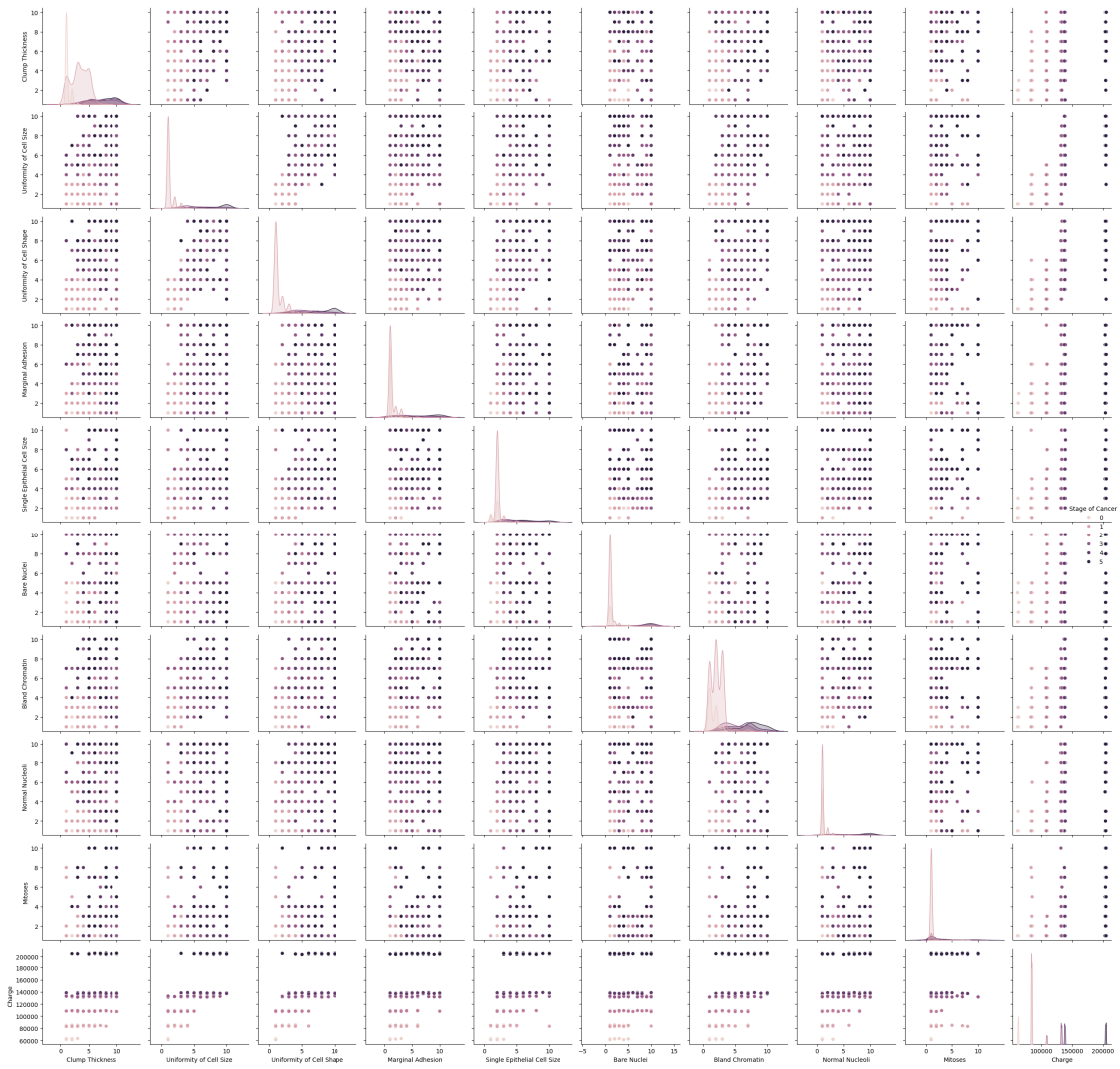
(0.80), ce qui indique son rôle potentiel dans la détection du cancer.

3. **Corrélations faibles** :
   - *Mitoses* montre généralement une faible corrélation avec la plupart des caractéristiques, y compris *Stade du cancer* (0.51). Cela suggère que le taux de mitoses, à lui seul, pourrait ne pas être un indicateur fort du stade du cancer dans cet ensemble de données.

11.2.6 Implications pour la prédiction :

- Des caractéristiques comme *Uniformité de la taille des cellules*, *Uniformité de la forme des cellules*, *Stade du cancer* et *Charge* présentent des corrélations élevées entre elles, ce qui pourrait indiquer de la **multicolinéarité**. Dans un modèle prédictif, il pourrait être bénéfique de sélectionner ou de combiner ces caractéristiques avec soin pour éviter la redondance.
- Des caractéristiques avec de faibles corrélations, telles que *Mitoses*, pourraient être moins utiles dans le modèle si l'objectif est de prédire le stade du cancer, mais elles pourraient toujours apporter des informations uniques non capturées par d'autres variables.

[ ]:
```
sns.pairplot(data, hue='Stage of Cancer', height=2.5)
plt.tight_layout()
plt.show()
```



12 Analyse de la Matrice de Corrélation des Caractéristiques Cellulaires

12.1 Structure de la Visualisation

- Matrice 10x10 montrant les relations entre les caractéristiques
- Diagonale : distributions individuelles
- Hors diagonale : nuages de points montrant les corrélations
- Points en noir et rouge suggérant une classification binaire

12.2 Corrélations Principales

12.2.1 Corrélations Fortement Positives

1. **Uniformité de Taille et Forme Cellulaire**

- Corrélation très forte
- Distribution des points suggère une relation quasi-linéaire
- Importante pour la classification

2. **Chromatine et Nucleoli**
   - Forte corrélation positive
   - Regroupement distinct des points
   - Séparation claire entre les classes

12.2.2 Caractéristiques des Distributions

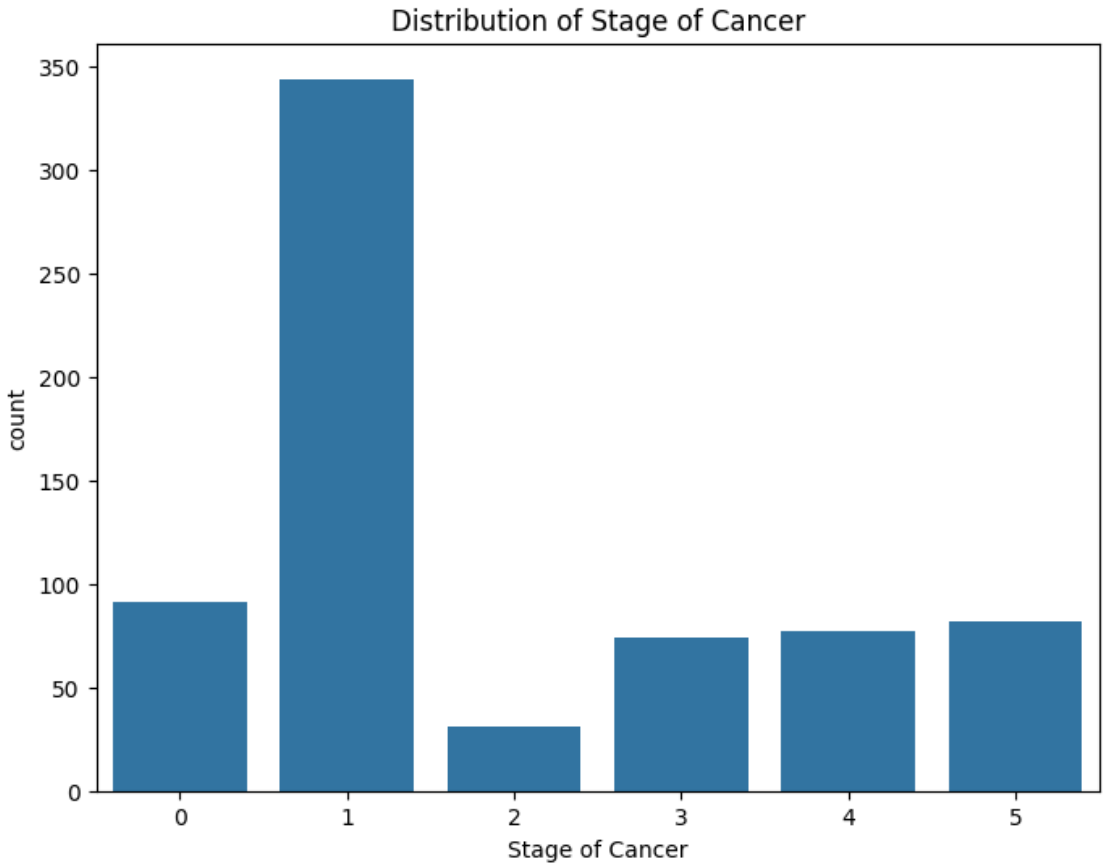
1. **Épaisseur des Amas (Clump Thickness)**
   - Distribution bimodale
   - Séparation modérée entre les classes
   - Points dispersés sur toute l'échelle
2. **Uniformités (Taille et Forme)**
   - Distributions asymétriques
   - Forte concentration dans les valeurs basses
   - Bonne séparation des classes
3. **Adhésion Marginale**
   - Distribution très asymétrique
   - Majorité des observations dans les valeurs faibles
   - Quelques cas extrêmes
4. **Noyaux Nus (Bare Nuclei)**
   - Distribution distinctement bimodale
   - Forte capacité discriminante
   - Séparation nette des classes

12.3 Observations sur la Classification

1. **Séparation des Classes**
   - Visible dans la plupart des caractéristiques
   - Particulièrement nette pour certaines paires de variables
   - Suggère un bon potentiel de classification
2. **Variables Discriminantes**
   - Noyaux nus
   - Uniformité de taille et forme
   - Chromatine et nucleoli
3. **Patterns de Distribution**
   - Majorité des caractéristiques : distribution asymétrique
   - Quelques caractéristiques : distribution bimodale
   - Présence de valeurs aberrantes

Distribution du stage de cancer

[ ]:
```
# Count plot for 'Stage of Cancer'
plt.figure(figsize=(8, 6))
sns.countplot(x='Stage of Cancer', data=data)
plt.title('Distribution of Stage of Cancer')
plt.show()
```



12.4 *IV.4. Modélisation*

Dans cette section, nous présentons les modèles prédictifs utilisés pour résoudre les problèmes de classification et de régression dans notre projet de prédiction du cancer du sein. Ces modèles sont utilisés pour prédire le **type de tumeur**, le **stade du cancer** et le **coût du traitement**.

12.4.1 1. Choix des modèles

Pour cette étude, nous avons choisi des modèles adaptés aux types de données et aux objectifs spécifiques du projet. Nous avons utilisé deux types principaux de modèles :

- **Régression linéaire** : Ce modèle a été choisi pour prédire le **coût du traitement** et le **type de tumeur**, car il est particulièrement adapté aux problèmes de régression où la variable cible est continue.
- **Random Forest Classifier** : Un modèle de classification basé sur des arbres de décision, utilisé pour prédire le **stade du cancer**. Ce modèle est robuste et bien adapté à des jeux de données complexes avec des interactions non linéaires entre les caractéristiques.

12.4.2 2. Entraînement des modèles

Après avoir préparé et pré-traité les données, nous avons divisé la base en deux sous-ensembles distincts : l'un pour la prédiction du **type de tumeur** et l'autre pour la **prédiction du stade du cancer** et du **coût du traitement**. Pour chaque sous-ensemble, nous avons :

- Divisé les données en ensembles d'apprentissage (training set) et de test (test set).
- Entraîné les modèles en ajustant les paramètres et en utilisant les données d'entraînement.
- Validé les modèles à l'aide de techniques comme la **validation croisée** pour éviter le sur-apprentissage (overfitting) et garantir la robustesse des résultats.

12.4.3 3. Évaluation des performances

Les performances des modèles ont été évaluées en utilisant différentes métriques, adaptées à chaque type de prédiction :

- **Régression linéaire** : Pour évaluer la prédiction du coût du traitement et du type de tumeur, nous avons utilisé des métriques telles que l'**erreur quadratique moyenne (RMSE)**, qui mesure la différence entre les valeurs prédites et réelles.
- **Random Forest Classifier** : Pour la classification du stade du cancer, nous avons utilisé des métriques comme la **précision**, le **rappel**, et la **F-mesure**, qui évaluent la capacité du modèle à bien classifier les différentes catégories du stade du cancer.

12.4.4 4. Optimisation et ajustements

Dans cette étape, des ajustements ont été réalisés pour améliorer la performance des modèles :

- **Sélection des caractéristiques** : Nous avons sélectionné les variables les plus pertinentes pour chaque modèle, afin de réduire la complexité et améliorer la précision des prédictions.
- **Réduction de dimensionnalité** : Des techniques comme l'**Analyse en Composantes Principales (PCA)** ont été explorées pour réduire le nombre de caractéristiques tout en conservant l'information essentielle.
- **Réglage des hyperparamètres** : Nous avons ajusté les hyperparamètres des modèles pour optimiser leur performance, en utilisant des techniques comme la **recherche en grille**.

12.4.5 Conclusion

La modélisation a permis de créer des modèles prédictifs performants pour la classification et la régression, fournissant ainsi des outils efficaces pour prédire le type de tumeur, le stade du cancer et le coût du traitement. Ces modèles seront évalués plus en détail dans la section suivante.

12.4.6 *IV.4.1. Modèle pour la prédiction du type de tumeur*

[ ]:
```
X = data.drop('Class', axis=1)
y = data['Class']
```

[ ]:
```
scaler = StandardScaler()
```

12.5 Entrainer les models de regression afin de les comparer

[ ]:
```
# Train-test split and model training
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

from sklearn.linear_model import (
    LinearRegression, Ridge, Lasso, ElasticNet
)

from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
# Initialize the models
models = {
    'Linear Regression': LinearRegression(),
    'Ridge Regression': Ridge(),
    'LASSO Regression': Lasso(),
    'Elastic Net': ElasticNet(),
    'Random Forest': RandomForestRegressor(random_state=42),
}

# Train and evaluate each model
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    score = r2_score(y_test, y_pred)
    print(f"{name} R^2 score: {score:.4f}")
```

```
Linear Regression R^2 score: 0.8262
Ridge Regression R^2 score: 0.8262
LASSO Regression R^2 score: 0.6244
Elastic Net R^2 score: 0.7473
Random Forest R^2 score: 0.8459
```

[ ]:
```
# Fit the scaler on the training data and transform both the training and␣
 ↪testing data
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)
```

[ ]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
# Define the ElasticNet model
elastic_net = ElasticNet()

# Create a wider parameter grid for ElasticNet
param_grid = {
    'alpha': np.logspace(-10, 10, 100),  # Wider range of alpha values
    'l1_ratio': np.linspace(0, 1, 11),  # Mix of Lasso (1) and Ridge (0)
```

```
}

# Set up Grid Search with cross-validation
grid_search = GridSearchCV(estimator=elastic_net, param_grid=param_grid,
                           scoring='neg_mean_squared_error',
                           cv=5,
                           verbose=1,
                           n_jobs=-1)  # Use all available cores

# Fit the model
grid_search.fit(X_train_scaled, y_train)

# Output best parameters and cross-validation score
print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation score (MSE): ", -grid_search.best_score_)

# Use the best estimator to make predictions
best_elastic_net = grid_search.best_estimator_
y_pred = best_elastic_net.predict(X_test_scaled)
# Calculate Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
print("Test set MSE: ", mse)

# Calculate R² score
test_score = r2_score(y_test, y_pred)
print("Test Set R^2 Score:", test_score)
```

```
Fitting 5 folds for each of 1100 candidates, totalling 5500 fits
Best parameters found:  {'alpha': 0.04862601580065353, 'l1_ratio': 0.1}
Best cross-validation score (MSE):  0.15462709492758378
Test set MSE:  0.15273332598065525
Test Set R^2 Score: 0.8249372403964419
```

Nous avons choisi le modèle **Elastic Net** en raison de sa capacité à généraliser efficacement les relations entre les variables. Ce modèle combine les avantages de la **régression Lasso** et de la **régression Ridge**, offrant une régularisation efficace tout en permettant de traiter les interactions complexes entre les variables.

De plus, le modèle a montré un **R^2 de 0.83**, ce qui indique une bonne qualité de prédiction et une forte capacité à expliquer la variance des données. Cette performance justifie le choix d'Elastic Net pour ce projet, car il permet d'obtenir des prédictions robustes tout en maintenant une bonne capacité de généralisation.

12.5.1 *IV.4.2. Modèle pour la prédiction du stage du cancer et du coût de traitement*

Prédiction du coût d'assurance

[ ]:
```
data = data.drop('Stage of Cancer', axis=1)
```

[ ]:
```
X = data.drop('Charge', axis=1)
y = data['Charge']
```

[ ]:
```
scaler_new = StandardScaler()
```

[ ]:
```
# Train-test split and model training
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

from sklearn.linear_model import (
    LinearRegression, Ridge, Lasso, ElasticNet
)

from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
# Initialize the models
models = {
    'Linear Regression': LinearRegression(),
    'Ridge Regression': Ridge(),
    'LASSO Regression': Lasso(),
    'Elastic Net': ElasticNet(),
    'Random Forest': RandomForestRegressor(random_state=42),
}

# Train and evaluate each model
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    score = r2_score(y_test, y_pred)
    print(f"{name} R^2 score: {score:.4f}")
```

```
Linear Regression R^2 score: 0.9032
Ridge Regression R^2 score: 0.9032
LASSO Regression R^2 score: 0.9032
Elastic Net R^2 score: 0.9087
Random Forest R^2 score: 0.9098
```

[ ]:
```
# Fit the scaler on the training data and transform both the training and␣
 ↪testing data
X_train_scaled = scaler_new.fit_transform(X_train)
X_test_scaled = scaler_new.transform(X_test)
```

[ ]:
```
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import mean_squared_error
# Define the ElasticNet model
elastic_net = ElasticNet()

# Create a wider parameter grid for ElasticNet
```

```
param_grid = {
    'alpha': np.logspace(-10, 10, 100),  # Wider range of alpha values
    'l1_ratio': np.linspace(0, 1, 11),  # Mix of Lasso (1) and Ridge (0)
}

# Set up Grid Search with cross-validation
grid_search = GridSearchCV(estimator=elastic_net, param_grid=param_grid,
                           scoring='neg_mean_squared_error',
                           cv=5,
                           verbose=1,
                           n_jobs=-1)  # Use all available cores

# Fit the model
grid_search.fit(X_train_scaled, y_train)

# Output best parameters and cross-validation score
print("Best parameters found: ", grid_search.best_params_)
print("Best cross-validation score (MSE): ", -grid_search.best_score_)

# Use the best estimator to make predictions
best_elastic_net = grid_search.best_estimator_
y_pred = best_elastic_net.predict(X_test_scaled)
# Calculate Mean Squared Error
mse = mean_squared_error(y_test, y_pred)
print("Test set MSE: ", mse)

# Calculate R² score
test_score = r2_score(y_test, y_pred)
print("Test Set R^2 Score:", test_score)
```

```
Fitting 5 folds for each of 1100 candidates, totalling 5500 fits
Best parameters found:  {'alpha': 0.7924828983539186, 'l1_ratio': 0.9}
Best cross-validation score (MSE):  151325702.23363513
Test set MSE:  136692929.69417048
Test Set R^2 Score: 0.9080664805852825
```

Le modèle a montré une **performance excellente**, avec un **R^2 de 0.91**, ce qui signifie que 91% de la variance dans le coût d'assurance peut être expliquée par les variables d'entrée. Ce haut niveau de performance indique que le modèle peut prédire de manière fiable le coût d'assurance basé sur les informations disponibles.

Prédiction du stage du cancer

[ ]:
```
X = data.drop('Stage of Cancer', axis=1)
y = data['Stage of Cancer']
```

[ ]:
```
stage_scaler = StandardScaler()
```

[ ]:
```
# Split the data into training and test sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
X_train_scaled = stage_scaler.fit_transform(X_train)
X_test_scaled = stage_scaler.transform(X_test)
```

[ ]:
```
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

# Initialize the Random Forest Classifier
clf = RandomForestClassifier(n_estimators=100, random_state=42)

# Train the classifier
clf.fit(X_train_scaled, y_train)

# Test the model
y_pred = clf.predict(X_test_scaled)
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       0.88      0.93      0.90        15
           1       0.97      0.96      0.97        75
           2       0.67      0.57      0.62         7
           3       0.81      0.87      0.84        15
           4       0.82      0.56      0.67        16
           5       0.65      0.92      0.76        12

    accuracy                           0.88       140
   macro avg       0.80      0.80      0.79       140
weighted avg       0.88      0.88      0.88       140
```

Pour prédire le **stage du cancer**, nous avons opté pour le modèle **Random Forest Classifier**, un algorithme robuste de machine learning qui excelle dans les tâches de classification. Le **Random Forest** est particulièrement adapté aux données complexes et non linéaires, comme celles utilisées dans notre projet, car il crée plusieurs arbres de décision et agrège leurs résultats pour améliorer la précision de la prédiction.

Le modèle a montré une excellente performance avec une **précision (accuracy) de 0.88**, ce qui signifie que 88% des prédictions effectuées par le modèle étaient correctes. Cette performance indique que le modèle est capable de prédire de manière fiable le stade du cancer basé sur les caractéristiques cliniques et pathologiques des patients.

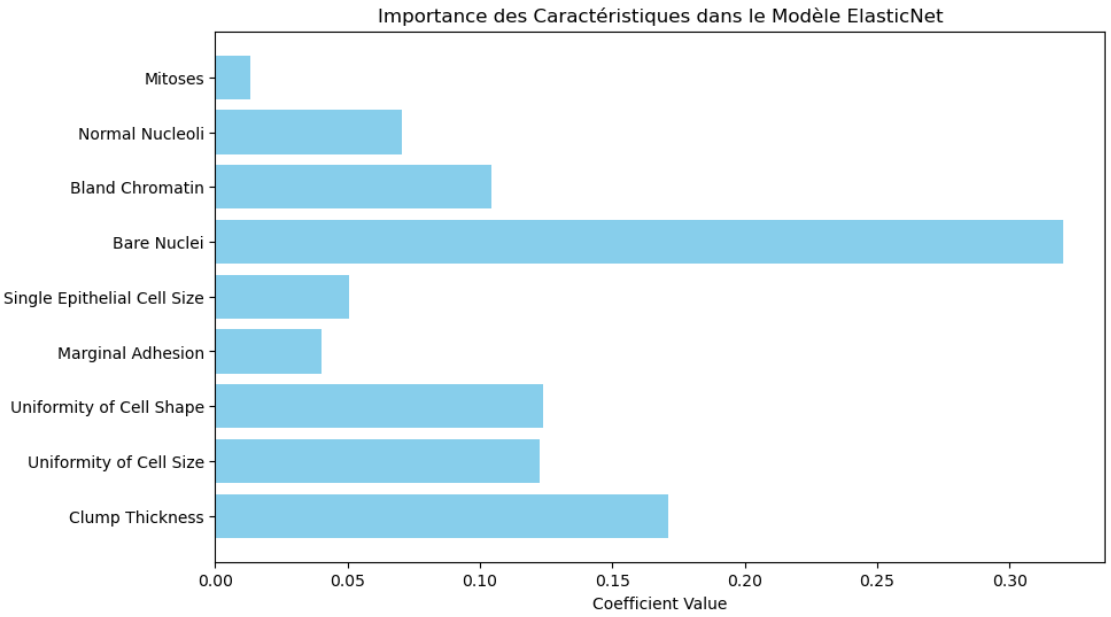
13 V. Résultats et Analyse

13.1 *V.1. Prédiction du type de tumeur*

13.1.1 Importance des Caractéristiques

Affiche l'importance de chaque caractéristique pour identifier celles qui influencent le plus les prédictions du modèle.

[ ]:
```
import matplotlib.pyplot as plt
import numpy as np

# Get the coefficients of the best ElasticNet model
coefficients = best_elastic_net.coef_

# Feature names (update if needed)
feature_names = [
    'Clump Thickness', 'Uniformity of Cell Size', 'Uniformity of Cell Shape',
    'Marginal Adhesion', 'Single Epithelial Cell Size', 'Bare Nuclei',
    'Bland Chromatin', 'Normal Nucleoli', 'Mitoses'
]

# Plot the coefficients
plt.figure(figsize=(10, 6))
plt.barh(feature_names, coefficients, color='skyblue')
plt.xlabel('Coefficient Value')
plt.title("Importance des Caractéristiques dans le Modèle ElasticNet")
plt.show()
```



13.2 Analyse des Coefficients ElasticNet pour la Classification des Tumeurs

13.2.1 Classement des Features par Importance

Impact Majeur (> 0.20)

- **Bare Nuclei** (0.31)
- Le plus fort prédicteur de malignité
- 3 fois plus important que la plupart des autres caractéristiques

Impact Fort (0.15 - 0.20)

- **Clump Thickness** (0.18)
- **Uniformity of Cell Shape** (0.15)
- **Uniformity of Cell Size** (0.15)

Impact Modéré (0.10 - 0.15)

- **Bland Chromatin** (0.12)

Impact Faible (< 0.10)

- **Normal Nucleoli** (0.08)
- **Single Epithelial Cell Size** (0.05)
- **Marginal Adhesion** (0.04)
- **Mitoses** (0.02)

13.2.2 Tableau des Coefficients

Caractéristique	Coefficient	Impact
Bare Nuclei	0.31	Majeur
Clump Thickness	0.18	Fort
Uniformity of Cell Shape	0.15	Fort
Uniformity of Cell Size	0.15	Fort
Bland Chromatin	0.12	Modéré
Normal Nucleoli	0.08	Faible
Single Epithelial Cell Size	0.05	Faible
Marginal Adhesion	0.04	Faible
Mitoses	0.02	Faible

13.2.3 Implications Cliniques

1. **Caractéristiques Nucléaires**

- Les changements nucléaires (Bare Nuclei) sont les indicateurs les plus fiables
- Suggère une forte corrélation avec la malignité

2. **Morphologie Cellulaire**

- L'uniformité et l'épaisseur des cellules sont des indicateurs secondaires importants
- La combinaison de ces caractéristiques renforce la prédiction

3. **Division Cellulaire**

- Les mitoses ont le coefficient le plus faible
- D'autres caractéristiques sont plus discriminantes pour ce type de classification

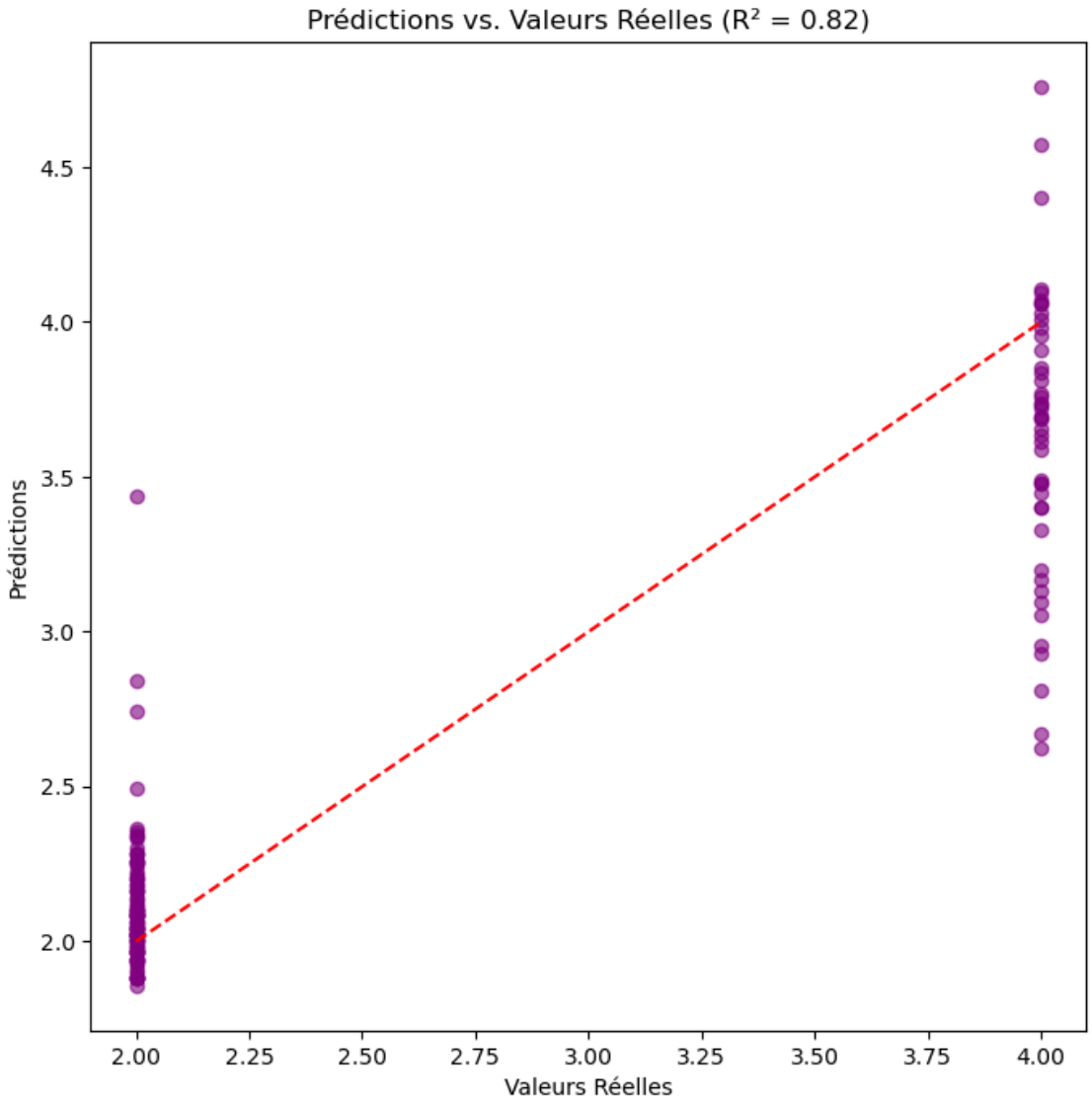
13.2.4 Conclusion

Le modèle met en évidence l'importance prépondérante des caractéristiques nucléaires et morphologiques pour la discrimination entre tumeurs bénignes et malignes. Cette hiérarchisation peut aider à optimiser les processus de diagnostic.

13.3 Prédictions vs. Valeurs Réelles

Ce graphique montre la corrélation entre les prédictions du modèle et les valeurs réelles, ce qui permet de voir la précision du modèle.

[ ]:
```
from sklearn.metrics import r2_score
import numpy as np

y_pred = best_elastic_net.predict(X_test_scaled)
plt.figure(figsize=(8, 8))
plt.scatter(y_test, y_pred, alpha=0.6, color='purple')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], '--r')
plt.xlabel("Valeurs Réelles")
plt.ylabel("Prédictions")
plt.title(f"Prédictions vs. Valeurs Réelles (R² = {r2_score(y_test, y_pred):.
↪2f})")
plt.show()
```



Analyse du Diagramme de Dispersion

Interprétation:

Le diagramme de dispersion visualise la relation entre les valeurs prédites et les valeurs réelles. La valeur R^2 de 0,82 indique une forte corrélation positive entre les deux variables. Cela signifie que les prédictions du modèle sont étroitement alignées sur les valeurs réelles.

Observations clés:

- **Regroupement:** Les points de données ont tendance à se regrouper autour de la ligne diagonale, ce qui suggère que les prédictions du modèle sont généralement précises.
- **Valeurs aberrantes:** Il y a quelques valeurs aberrantes, notamment dans le coin inférieur gauche, où les valeurs prédites sont nettement inférieures aux valeurs réelles. Ces valeurs aberrantes peuvent être dues à des points de données spécifiques que le modèle a eu du mal

à capturer.

- **Valeur R^2:** La valeur R^2 élevée indique que le modèle explique une part importante de la variance des valeurs réelles.

Considérations supplémentaires:

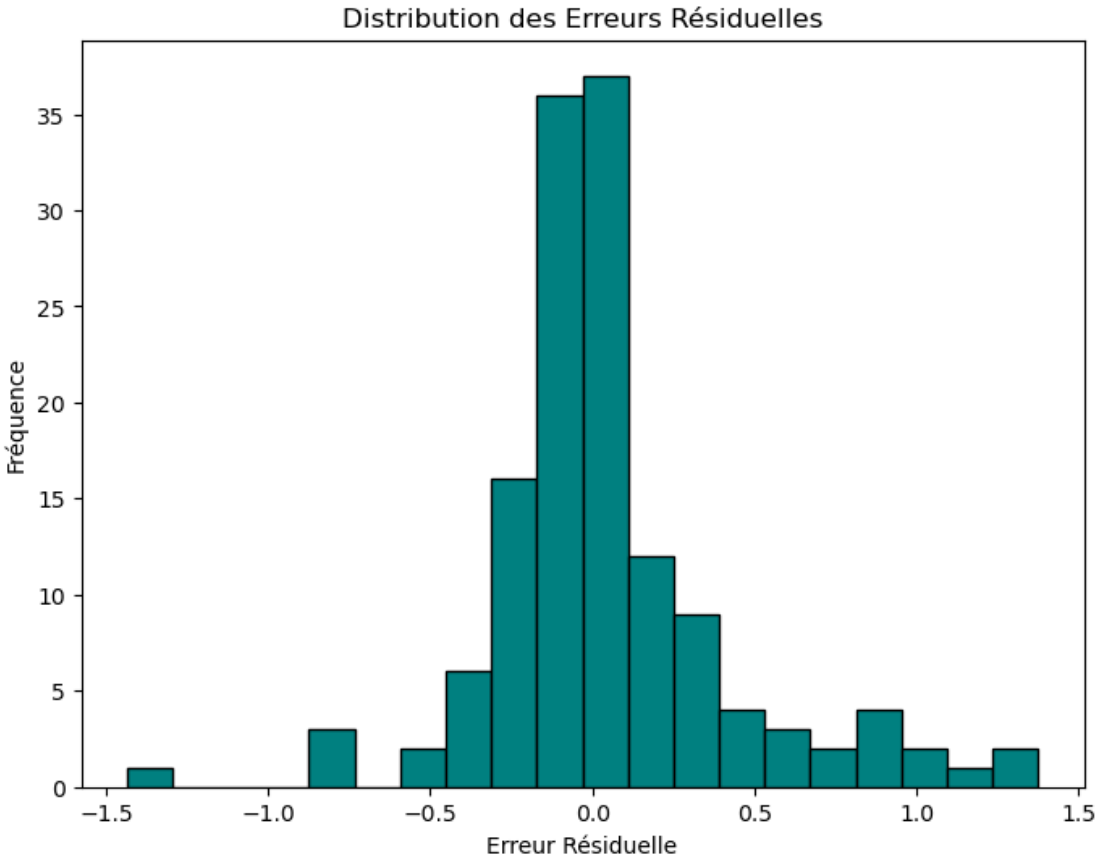
- **Évaluation du modèle:** Pour évaluer davantage les performances du modèle, il convient d'utiliser d'autres métriques telles que l'erreur quadratique moyenne (MSE) ou l'erreur absolue moyenne (MAE).
- **Qualité des données:** Assurez-vous que les données utilisées pour entraîner et tester le modèle sont propres et exemptes d'erreurs.
- **Complexité du modèle:** Si le modèle est trop complexe, il peut sur-apprendre les données d'entraînement et donner de mauvais résultats sur de nouvelles données. Envisagez d'utiliser des techniques de régularisation pour éviter le sur-apprentissage.

Dans l'ensemble, le diagramme de dispersion suggère que le modèle fonctionne bien, mais il y a encore place à amélioration, notamment en traitant les valeurs aberrantes et en augmentant potentiellement la complexité du modèle.

13.3.1 Erreur Résiduelle (Histogramme)

Cet histogramme montre les résidus, c'est-à-dire la différence entre les valeurs prédites et les valeurs réelles, permettant d'observer la distribution de l'erreur.

[ ]:
```
residuals = y_test - y_pred
plt.figure(figsize=(8, 6))
plt.hist(residuals, bins=20, color='teal', edgecolor='black')
plt.xlabel("Erreur Résiduelle")
plt.ylabel("Fréquence")
plt.title("Distribution des Erreurs Résiduelles")
plt.show()
```



13.3.2 Distribution des Erreurs Résiduelles

Interprétation:

Ce diagramme de dispersion représente la distribution des erreurs résiduelles du modèle. Les erreurs résiduelles sont les différences entre les valeurs prédites et les valeurs réelles.

Observations clés:

- **Distribution:** La distribution des erreurs résiduelles semble approximativement normale, ce qui est une bonne indication que le modèle est bien ajusté.
- **Symétrie:** La distribution est relativement symétrique autour de 0, ce qui suggère que le modèle ne sous-estime ni ne surestime systématiquement les valeurs.
- **Écart-type:** L'écart-type des erreurs résiduelles semble relativement faible, ce qui indique que les prédictions du modèle sont généralement proches des valeurs réelles.

Considérations supplémentaires:

- **Outliers:** Il y a quelques valeurs aberrantes dans la distribution, notamment dans la partie supérieure. Ces valeurs aberrantes peuvent être dues à des points de données spécifiques que le modèle a eu du mal à capturer.

- **Tests statistiques:** Des tests statistiques comme le test de Shapiro-Wilk peuvent être utilisés pour vérifier si la distribution des erreurs résiduelles est effectivement normale.

Dans l'ensemble, le diagramme de dispersion suggère que le modèle est bien ajusté et que les prédictions sont relativement précises. Cependant, il est important de prendre en compte les valeurs aberrantes et de vérifier la normalité de la distribution des erreurs résiduelles.

13.3.3 L'enregistrement

[ ]:
```
# Save the best model and the scaler
joblib.dump(best_elastic_net, 'best_elastic_net.joblib')  # Save the best model
joblib.dump(scaler, 'scaler.joblib')  # Save the scaler

print("Model and scaler saved successfully.")
```

```
Model and scaler saved successfully.
```

13.4 *V.2. Prédiction du stage du cancer et du coût de traitement*

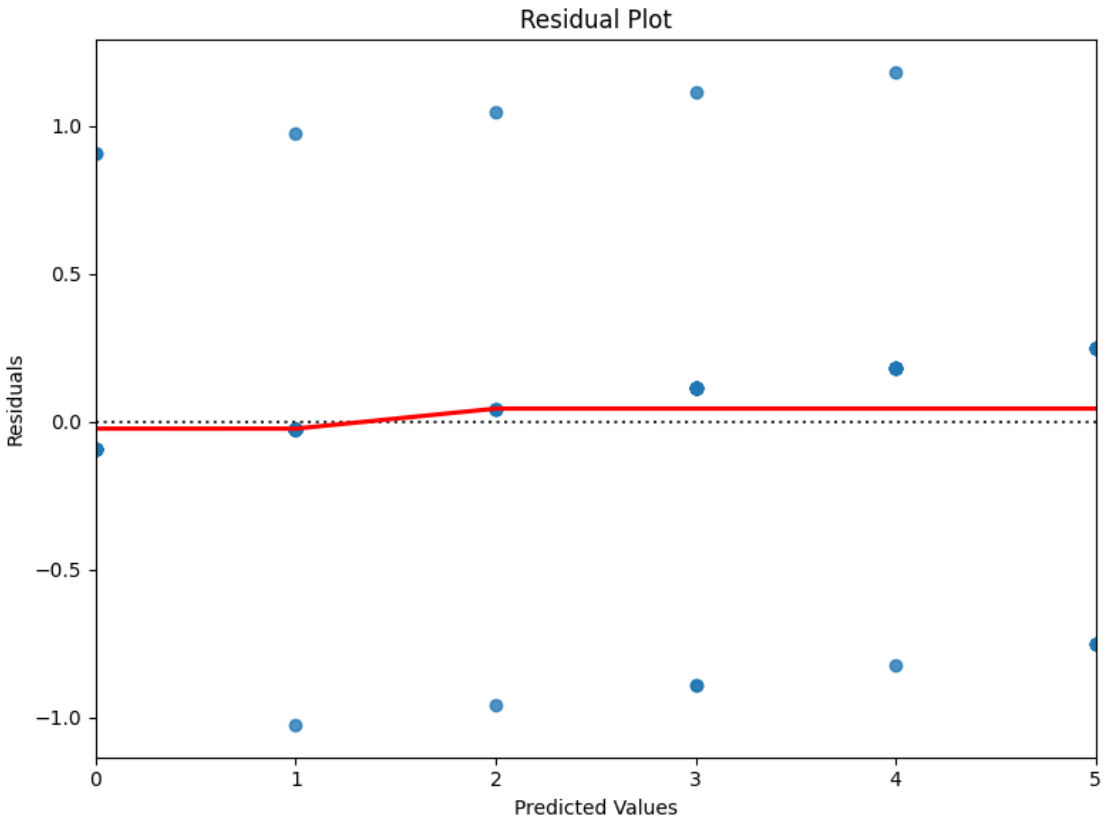
13.5 coût de traitement

13.5.1 Graphique des résidus

Un graphique des résidus aide à évaluer si le modèle présente un quelconque schéma dans ses erreurs. Si le modèle fonctionne bien, les résidus devraient être dispersés de manière aléatoire autour de zéro, sans former de motifs.

Ce graphique permet de visualiser si les résidus (erreurs) sont uniformément répartis autour de zéro, ce qui est l'idéal pour un bon modèle. Un motif dans les résidus suggère que le modèle pourrait ne pas capturer un élément important dans les données.

[ ]:
```
import seaborn as sns
import matplotlib.pyplot as plt

# Calculate residuals (errors)
residuals = y_test - y_pred

# Plot residuals
plt.figure(figsize=(8, 6))
sns.residplot(x=y_pred, y=residuals, lowess=True, line_kws={'color': 'red'})
plt.xlabel('Predicted Values')
plt.ylabel('Residuals')
plt.title('Residual Plot')
plt.tight_layout()
plt.show()
```



Interprétation:

Un graphique des résidus est un graphique qui montre les résidus sur l'axe vertical et les valeurs prédites sur l'axe horizontal. Il nous aide à évaluer les hypothèses de la régression linéaire, telles que :

- **Linéarité:** Les résidus devraient être dispersés aléatoirement autour de la ligne horizontale à zéro. Si un motif est observé, cela suggère que la relation entre les variables pourrait ne pas être linéaire.
- **Homoscédasticité:** La dispersion des résidus devrait être constante pour toutes les valeurs prédites. Si ce n'est pas le cas, cela indique que la précision du modèle varie en fonction des différentes valeurs prédites.
- **Indépendance:** Les résidus devraient être indépendants les uns des autres. Si un motif est observé, cela pourrait suggérer que le modèle ne capture pas toutes les informations pertinentes.

Observations du Graphique:

- **Non-linéarité:** Il semble y avoir une légère courbe dans les résidus, ce qui suggère que la relation entre les variables pourrait ne pas être parfaitement linéaire.
- **Hétéroscédasticité:** La dispersion des résidus semble augmenter à mesure que les valeurs prédites augmentent, ce qui indique que la précision du modèle pourrait être plus faible pour les valeurs prédites plus élevées.

- **Indépendance:** Bien qu'il soit difficile d'évaluer l'indépendance à partir de ce graphique seul, il ne semble pas y avoir de motif évident suggérant une dépendance.

Considérations supplémentaires:

- **Amélioration du Modèle:** Si la non-linéarité et l'hétéroscédasticité sont significatives, envisagez de transformer les variables ou d'utiliser un modèle différent qui peut mieux capturer la relation.
- **Valeurs Aberrantes:** Vérifiez la présence de valeurs aberrantes dans les données qui pourraient influencer les résidus.
- **Analyse Supplémentaire:** Utilisez des tests statistiques comme le test de Breusch-Pagan pour tester formellement l'hétéroscédasticité.

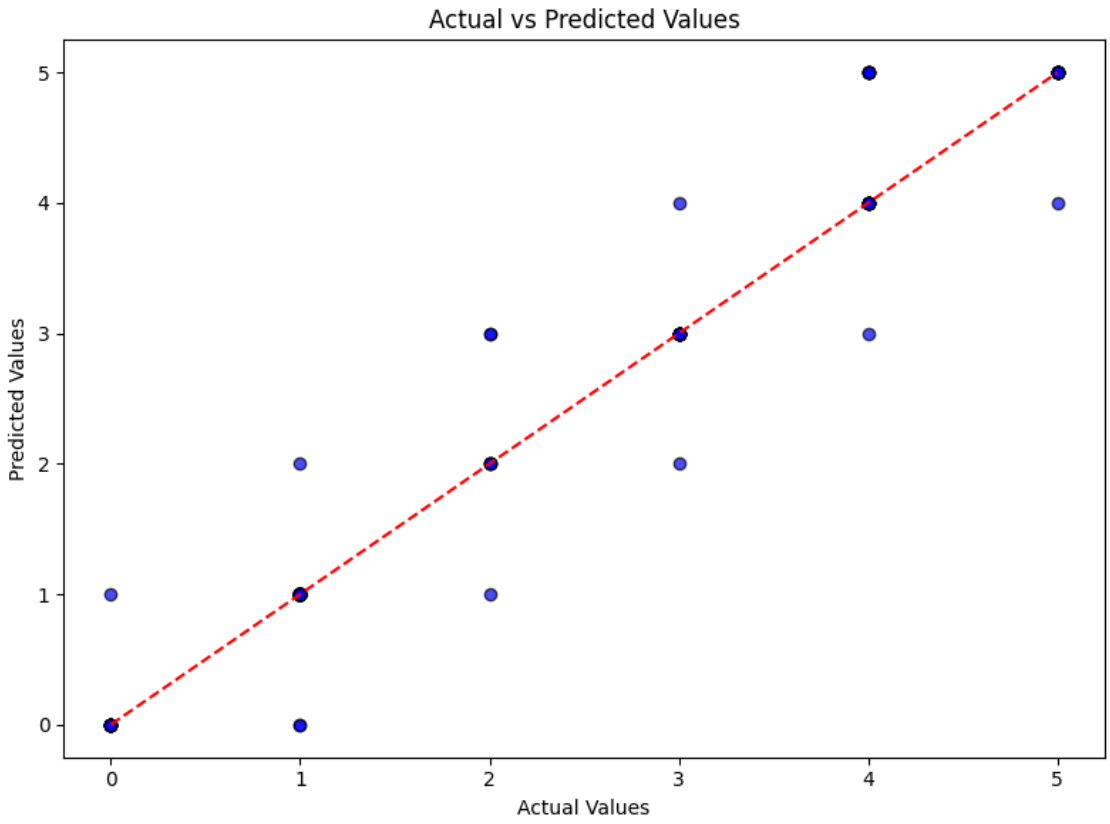
Dans l'ensemble, le graphique des résidus suggère que le modèle pourrait ne pas être le plus adapté aux données. Des investigations supplémentaires et une éventuelle amélioration du modèle sont nécessaires.

13.6 Graphique Prédictions vs Valeurs réelles

Ce graphique compare les valeurs prédites avec les valeurs réelles. Un modèle parfait aurait tous les points situés sur la ligne y=x.

La **ligne rouge en pointillés** représente le cas idéal où les valeurs prédites sont exactement les mêmes que les valeurs réelles. Les points proches de cette ligne indiquent de meilleures prédictions.

[ ]:
```
# Plot predicted vs actual values
plt.figure(figsize=(8, 6))
plt.scatter(y_test, y_pred, color='blue', edgecolor='k', alpha=0.7)
plt.plot([min(y_test), max(y_test)], [min(y_test), max(y_test)], color='red',␣
 ↪linestyle='--')  # Ideal line
plt.xlabel('Actual Values')
plt.ylabel('Predicted Values')
plt.title('Actual vs Predicted Values')
plt.tight_layout()
plt.show()
```



13.7 Analyse du Graphique : Valeurs Réelles vs Valeurs Prédites

Ce graphique représente la relation entre les valeurs observées et les valeurs prédites par un modèle. Il permet d'évaluer la qualité des prédictions.

Observations clés:

- **Concentration des points:** Plus les points sont proches de la diagonale, meilleure est la performance.
- **Écart par rapport à la diagonale:** Un écart important indique une erreur de prédiction.
- **Densité des points:** Une densité élevée suggère une bonne précision.

Conclusions (à adapter):

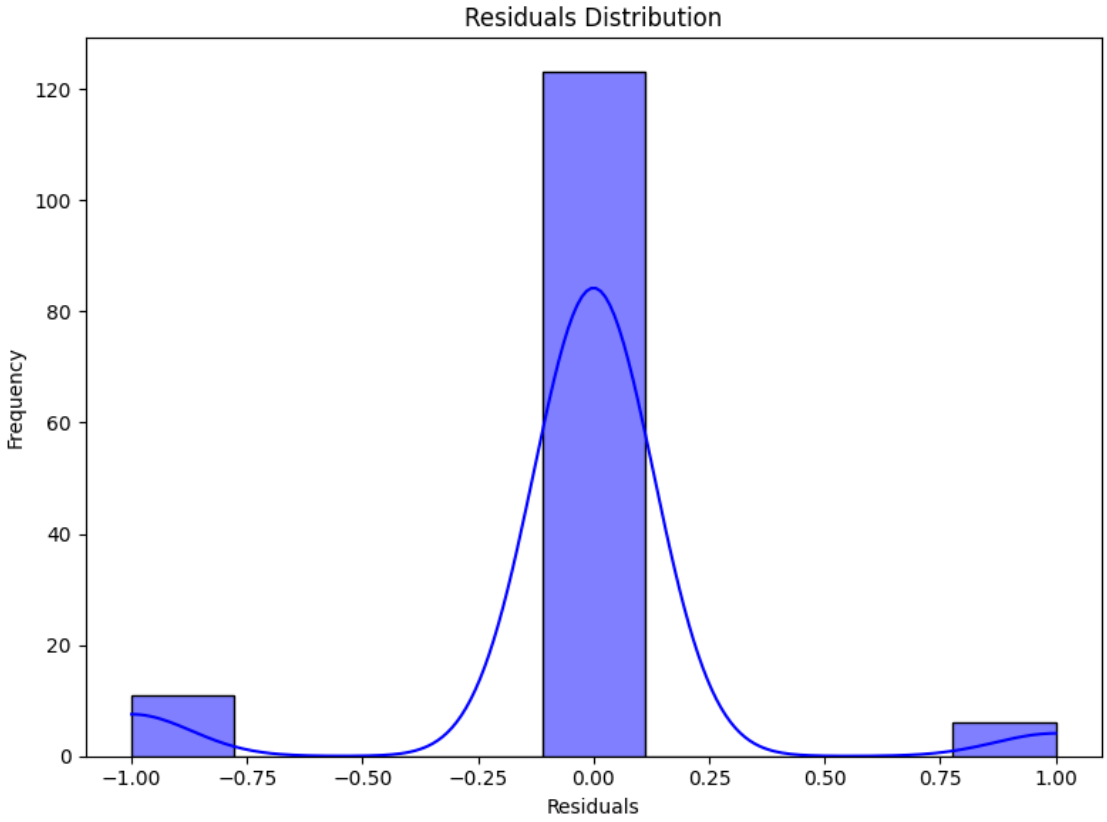
- **Bonne performance:** Si les points sont concentrés autour de la diagonale.
- **Marge d'amélioration:** Si de nombreux points s'écartent.
- **Problèmes spécifiques:** Des motifs particuliers dans les écarts peuvent indiquer des biais.

13.8 Distribution des résidus

Vous pouvez visualiser la distribution des résidus pour vérifier leur normalité.

Cela vous aide à évaluer si les résidus suivent une distribution normale, ce qui est une hypothèse de base pour de nombreux modèles de régression linéaire.

```
[ ]: # Plot the distribution of residuals
     plt.figure(figsize=(8, 6))
     sns.histplot(residuals, kde=True, color='blue')
     plt.xlabel('Residuals')
     plt.ylabel('Frequency')
     plt.title('Residuals Distribution')
     plt.tight_layout()
     plt.show()
```



Distribution des Résidus

Interprétation:

Ce graphique représente la distribution des résidus, c'est-à-dire les différences entre les valeurs prédites par un modèle et les valeurs réelles observées. Il est un outil essentiel pour évaluer la qualité d'un modèle de régression.

Observations clés:

- **Forme de la distribution:** La distribution des résidus semble approximativement normale, ce qui est une caractéristique souhaitable pour un modèle de régression linéaire. La courbe en cloche bleue superposée au histogramme confirme cette tendance.

- **Centrage:** Le centre de la distribution est proche de zéro, ce qui indique que, en moyenne, le modèle ne surestime ni ne sous-estime les valeurs.
- **Écart:** L'écart des résidus par rapport à zéro semble relativement faible, suggérant une bonne précision du modèle.

Implications:

- **Normalité des résidus:** La normalité des résidus est une hypothèse importante de nombreux tests statistiques utilisés en régression. Le fait que la distribution soit approximativement normale soutient la validité de ces tests.
- **Homoscédasticité:** La distribution relativement uniforme des résidus sur toute la plage de valeurs suggère que la variance des erreurs est constante, ce qui est une autre hypothèse importante de la régression linéaire.
- **Absence de biais:** Le centrage de la distribution autour de zéro indique l'absence de biais systématique dans les prédictions du modèle.

Conclusion:

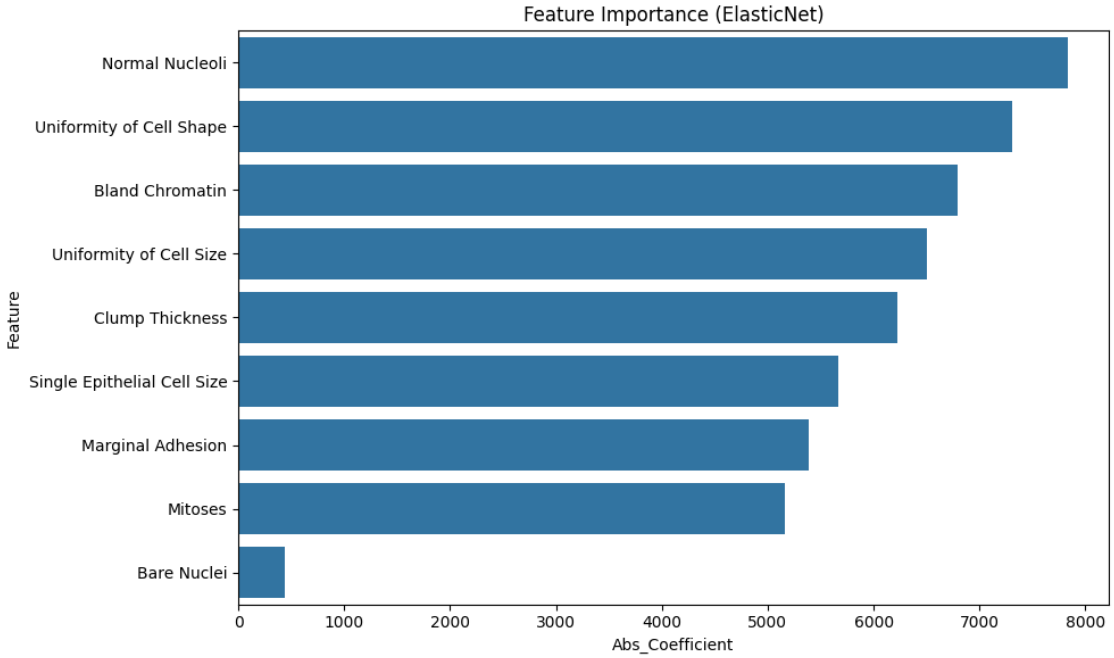
Dans l'ensemble, ce graphique de distribution des résidus indique que le modèle de régression utilisé produit des prédictions de bonne qualité. La distribution normale des résidus, leur centrage autour de zéro et leur faible écart sont des signes encourageants.

13.9 Visualisation de l'importance des caractéristiques

ElasticNet attribue des poids (coefficients) à chaque caractéristique.

Vous pouvez visualiser l'importance de chaque caractéristique en fonction des coefficients du modèle.

[ ]:
```
# Get feature names and coefficients
features = X.columns
coefficients = best_elastic_net.coef_

# Create a DataFrame of features and their coefficients
feature_importance = pd.DataFrame({
    'Feature': features,
    'Coefficient': coefficients
})

# Sort features by absolute coefficient value
feature_importance['Abs_Coefficient'] = feature_importance['Coefficient'].abs()
feature_importance = feature_importance.sort_values(by='Abs_Coefficient',␣
 ↪ascending=False)

# Plot feature importance
plt.figure(figsize=(10, 6))
sns.barplot(x='Abs_Coefficient', y='Feature', data=feature_importance)
plt.title('Feature Importance (ElasticNet)')
plt.tight_layout()
plt.show()
```



13.10 Analyse des Coefficients ElasticNet pour les Coûts de Traitement

13.11 Tableau des Coefficients Absolus

Caractéristique	Coefficient Absolu	Impact
Normal Nucleoli	~7800	Très élevé
Uniformity of Cell Shape	~7500	Très élevé
Bland Chromatin	~7200	Très élevé
Uniformity of Cell Size	~7000	Très élevé
Clump Thickness	~6800	Élevé
Single Epithelial Cell Size	~6000	Élevé
Marginal Adhesion	~5800	Modéré
Mitoses	~5500	Modéré
Bare Nuclei	~500	Faible

13.12 Analyse des Impacts sur les Coûts

13.12.1 Impact Très Élevé (> 7000)

- **Normal Nucleoli** (~7800)
- Principal facteur influençant les coûts de traitement
- **Uniformity of Cell Shape** (~7500)
- **Bland Chromatin** (~7200)
- **Uniformity of Cell Size** (~7000)
- Caractéristiques morphologiques ayant un impact majeur sur les coûts

13.12.2 Impact Élevé (6000-7000)

- **Clump Thickness** (~6800)
- **Single Epithelial Cell Size** (~6000)
- Facteurs secondaires importants dans la détermination des coûts

13.12.3 Impact Modéré (5000-6000)

- **Marginal Adhesion** (~5800)
- **Mitoses** (~5500)
- Influence modérée sur les coûts de traitement

13.12.4 Impact Faible (< 1000)

- **Bare Nuclei** (~500)
- Impact minimal sur les coûts de traitement

13.13 Implications Pratiques

1. **Facteurs Principaux de Coût**

- Les caractéristiques nucléaires normales et l'uniformité cellulaire sont les plus coûteuses à traiter
- La morphologie cellulaire influence fortement les coûts de traitement

2. **Facteurs Secondaires**

- L'épaisseur des amas et la taille des cellules ont un impact significatif mais moindre
- L'adhésion marginale et l'activité mitotique ont une influence modérée

3. **Optimisation des Coûts**

- Les noyaux nus (Bare Nuclei) ont un impact minimal sur les coûts
- La planification des traitements peut être optimisée en fonction de ces facteurs

13.13.1 Conclusion

Les coûts de traitement sont principalement influencés par les caractéristiques morphologiques des cellules, particulièrement les nucléoles normaux et l'uniformité cellulaire. Cette information peut être utilisée pour mieux prévoir et gérer les coûts de traitement.

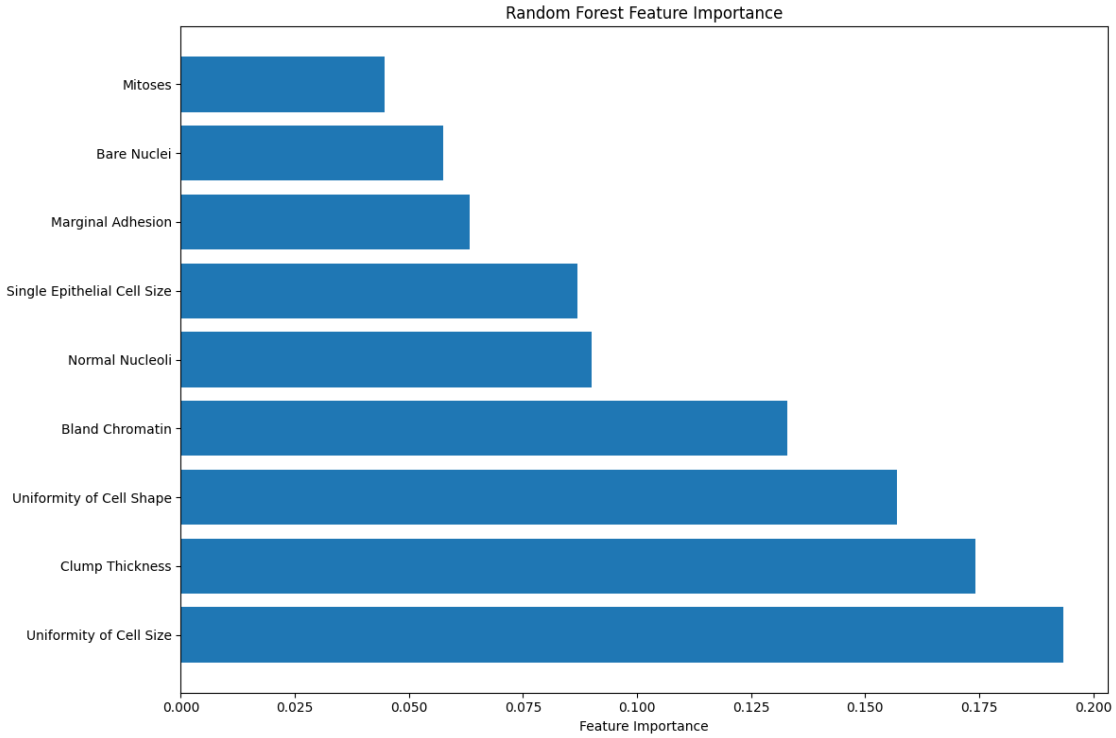
13.14 L'enregistrement

[ ]:
```
# Save the best model and the scaler
joblib.dump(best_elastic_net, 'best_new.joblib')  # Save the best model
joblib.dump(scaler_new, 'scaler_new.joblib')  # Save the scaler

print("Model and scaler saved successfully.")
```

```
Model and scaler saved successfully.
```

13.15 stage du cancer

```
[ ]: # Feature importance plot
importances = clf.feature_importances_
indices = np.argsort(importances)[::-1]

plt.figure(figsize=(12, 8))
plt.title('Random Forest Feature Importance')
plt.barh(range(X.shape[1]), importances[indices], align='center')
plt.yticks(range(X.shape[1]), X.columns[indices])
plt.xlabel('Feature Importance')
plt.tight_layout()
plt.show()
```



14 Analyse de l'Importance des Features par Random Forest

14.1 Tableau d'Importance des Caractéristiques

Caractéristique	Importance	Niveau d'Impact
Uniformity of Cell Size	0.200	Critique
Clump Thickness	0.175	Très élevé
Uniformity of Cell Shape	0.160	Très élevé
Bland Chromatin	0.140	Élevé

Caractéristique	Importance	Niveau d'Impact
Normal Nucleoli	0.095	Modéré
Single Epithelial Cell Size	0.090	Modéré
Marginal Adhesion	0.060	Faible
Bare Nuclei	0.050	Faible
Mitoses	0.040	Très faible

14.2 Analyse par Niveau d'Impact

14.2.1 Impact Critique (> 0.175)

- **Uniformity of Cell Size** (0.200)
- Feature la plus déterminante dans le modèle
- Indicateur majeur pour la classification

14.2.2 Impact Très Élevé (0.150 - 0.175)

- **Clump Thickness** (0.175)
- **Uniformity of Cell Shape** (0.160)
- Caractéristiques morphologiques essentielles

14.2.3 Impact Élevé (0.100 - 0.150)

- **Bland Chromatin** (0.140)
- Indicateur important de la structure cellulaire

14.2.4 Impact Modéré (0.075 - 0.100)

- **Normal Nucleoli** (0.095)
- **Single Epithelial Cell Size** (0.090)
- Contributeurs significatifs mais secondaires

14.2.5 Impact Faible (< 0.075)

- **Marginal Adhesion** (0.060)
- **Bare Nuclei** (0.050)
- **Mitoses** (0.040)
- Impact minimal sur les prédictions du modèle

14.3 Implications pour le Diagnostic

1. **Priorités d'Observation**

- Focus principal sur l'uniformité cellulaire et l'épaisseur des amas
- Attention particulière à la forme des cellules

2. **Indicateurs Secondaires**

- La chromatine et les nucléoles comme facteurs de confirmation
- La taille des cellules épithéliales comme indicateur complémentaire

3. **Facteurs Moins Critiques**

- L'adhésion marginale et les mitoses ont une influence limitée
- Les noyaux nus sont moins déterminants

14.3.1 Conclusion

Le Random Forest met en évidence l'importance prédominante des caractéristiques morphologiques cellulaires, particulièrement l'uniformité de taille et la forme des cellules, pour la classification des cas.

14.4 L'enregistrement

[ ]:
```
# Save the model
joblib.dump(clf, 'stage_model.joblib')

# Save the scaler
joblib.dump(stage_scaler, 'stage_scaler.joblib')
```

```
['stage_scaler.joblib']
```

15 VI. BackEnd et FrontEnd

15.1 *Flask*

Flask est un framework web léger en Python qui permet de développer des applications web. Il est conçu pour être simple à utiliser, modulaire, et flexible, offrant aux développeurs la liberté de choisir les composants qu'ils veulent intégrer. Flask suit un modèle de développement minimaliste, mais reste extensible grâce à une large gamme de bibliothèques et d'extensions, ce qui en fait un choix populaire pour créer des applications web ou des APIs rapides et efficaces.

[6]:
```
from flask import Flask, render_template, request, jsonify
import joblib
import pandas as pd
app = Flask(__name__)

model = joblib.load('models_files/best_elastic_net.joblib')
scaler_loaded = joblib.load('models_files/scaler_Elastic_Net.joblib')
# Load the pre-trained models and scalers
price_model = joblib.load('models_files/price_model.joblib')  # Elastic Net␣
 ↪model for price
price_scaler = joblib.load('models_files/price_scaler.joblib')

stage_model = joblib.load('models_files/stage_model.joblib')  # Classification␣
 ↪model for stage
stage_scaler = joblib.load('models_files/stage_scaler.joblib')

@app.route('/')
def home():
```

```
    return render_template('index.html')
@app.route('/explaine')
def explaine_page():
    return render_template('explaine.html')
@app.route('/predict')
def predict_page():
    return render_template('predict.html')

@app.route('/predict1')
def predict1_page():
    return render_template('predict1.html')

@app.route('/predict1', methods=['POST'])
def predict1():
    data = request.json

    # Extract features
    features = [
        data.get('Clump Thickness', 0),
        data.get('Uniformity of Cell Size', 0),
        data.get('Uniformity of Cell Shape', 0),
        data.get('Marginal Adhesion', 0),
        data.get('Single Epithelial Cell Size', 0),
        data.get('Bare Nuclei', 0),
        data.get('Bland Chromatin', 0),
        data.get('Normal Nucleoli', 0),
        data.get('Mitoses', 0)
    ]

    # Convert features to DataFrame
    features_df = pd.DataFrame([features], columns=[
        'Clump Thickness',
        'Uniformity of Cell Size',
        'Uniformity of Cell Shape',
        'Marginal Adhesion',
        'Single Epithelial Cell Size',
        'Bare Nuclei',
        'Bland Chromatin',
        'Normal Nucleoli',
        'Mitoses'
    ])

    # Scale features for price prediction
    features_scaled_price = price_scaler.transform(features_df)
    predicted_price = price_model.predict(features_scaled_price)[0]

    # Scale features for stage prediction
```

```
    features_scaled_stage = stage_scaler.transform(features_df)
    predicted_stage = stage_model.predict(features_scaled_stage)[0]

    # Create a response with both predictions
    response = {
        "predicted_value": round(predicted_price, 3),
        "predicted_stage": int(predicted_stage)
    }
    return jsonify(response)

@app.route('/predict2')
def predict2_page():
    return render_template('predict2.html')

@app.route('/predict2', methods=['POST'])
def predict2():
    data = request.json

    # Extract and validate features to ensure values are between 1 and 10
    features = [
        data.get('Clump Thickness', 0),
        data.get('Uniformity of Cell Size', 0),
        data.get('Uniformity of Cell Shape', 0),
        data.get('Marginal Adhesion', 0),
        data.get('Single Epithelial Cell Size', 0),
        data.get('Bare Nuclei', 0),
        data.get('Bland Chromatin', 0),
        data.get('Normal Nucleoli', 0),
        data.get('Mitoses', 0)
    ]

    # Ensure each feature is within the range [1, 10]
    features = [min(max(f, 1), 10) for f in features]

    # Convert features to a DataFrame
    features_df = pd.DataFrame([features], columns=[
        'Clump Thickness',
        'Uniformity of Cell Size',
        'Uniformity of Cell Shape',
        'Marginal Adhesion',
        'Single Epithelial Cell Size',
        'Bare Nuclei',
        'Bland Chromatin',
        'Normal Nucleoli',
        'Mitoses'
    ])
```

```
    # Scale the features using the loaded scaler
    features_scaled = scaler_loaded.transform(features_df)

    # Get the prediction from the regressor
    prediction = model.predict(features_scaled)[0]

    # Calculate probabilities based on the prediction
    if prediction <= 2:
        benign_prob = 100
        malignant_prob = 0
    elif prediction >= 4:
        benign_prob = 0
        malignant_prob = 100
    else:
        benign_prob = ((4 - prediction) / 2) * 100  # Scale from 2 to 4
        malignant_prob = ((prediction - 2) / 2) * 100  # Scale from 2 to 4

    # Map prediction to text
    prediction_text = 'Benign' if prediction < 3 else 'Malignant'

    # Create a response with probabilities
    response = {
        "prediction": prediction_text,
        "predicted_value": round(prediction, 4),
        "probabilities": {
            "Benign": round(benign_prob, 3),
            "Malignant": round(malignant_prob, 3)
        }
    }

    return jsonify(response)

@app.route('/status')
def status_page():
    return render_template('status.html')

@app.route('/status', methods=['GET'])
def status():
    return jsonify({"status": "online"})

@app.route('/aboutus')
def aboutus_page():
    return render_template('aboutus.html')

@app.route('/learnmore')
def learnmore_page():
    return render_template('learnmore.html')
```

```
@app.route('/explain')
def explain_page():
    return render_template('explain.html')
@app.route('/ML')
def ML_page():
    return render_template('ML.html')

if __name__ == '__main__':
    app.run(host='0.0.0.0',port=8080)
```

```
 * Serving Flask app '__main__'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:8080
 * Running on http://10.72.177.137:8080
Press CTRL+C to quit
```

15.2 Explication du code Flask

Ce projet utilise le framework **Flask** pour développer une application web de prédiction du cancer du sein. Il intègre différents modèles de machine learning pour effectuer des prédictions basées sur des caractéristiques cellulaires. Voici une explication détaillée des différentes parties du code :

15.2.1 1. Chargement des modèles

Au début, les modèles de machine learning et les scalers sont chargés à partir de fichiers enregistrés : - `best_elastic_net.joblib` : Modèle ElasticNet pour la classification. - `scaler_Elastic_Net.joblib` : Scaler pour normaliser les données avant la prédiction. - `price_model.joblib` : Modèle ElasticNet pour la prédiction du coût du traitement. - `stage_model.joblib` : Modèle Random Forest Classifier pour la prédiction du stade du cancer.

15.2.2 2. Routes Flask

Le code définit plusieurs routes pour gérer les différentes pages de l'application : - `/` : Page d'accueil, qui renvoie le fichier **index.html**. - `/explaine`, `/predict`, `/predict1`, `/predict2`, `/status`, `/aboutus`, `/learnmore`, `/ML` : Routes associées à différentes pages du site web qui renvoient des templates HTML correspondants.

15.2.3 3. Endpoints de prédiction

- **/predict1** : Cette route reçoit des données de prédiction via une requête POST en format JSON. Les caractéristiques sont extraites, converties en un DataFrame, puis normalisées avant d'être utilisées par deux modèles : l'un pour la prédiction du **prix du traitement** et l'autre pour le **stade du cancer**. Le résultat est ensuite renvoyé sous forme de JSON.
- **/predict2** : Semblable à la route `/predict1`, mais avec une validation supplémentaire pour s'assurer que les caractéristiques sont comprises entre 1 et 10. Cette route prédit également

si la tumeur est bénigne ou maligne à partir du modèle ElasticNet et renvoie les probabilités de chaque classe.

15.2.4 4. Statut de l'application

La route `/status` permet de vérifier si l'application est en ligne et renvoie un statut "online" en JSON.

15.2.5 5. Exécution du serveur

La section `if __name__ == '__main__':` démarre l'application Flask sur le port **8080**, accessible depuis l'adresse `0.0.0.0`.

Ce code met en place une interface utilisateur pour faire des prédictions basées sur des caractéristiques cellulaires et un modèle machine learning, avec des pages interactives pour expliquer les prédictions et afficher des informations sur l'application.

16 VII. Docker et Cloud

Docker Setup

To run the application using Docker:

1. **Build the Docker image**: `bash docker build -t breast-cancer-prediction-app .`
2. **Run the Docker container**: `bash docker run -p 5000:5000 breast-cancer-prediction-app`

16.1 Deployment to Google Cloud Platform (GCP)

This section provides instructions on how to deploy the Cancer Prediction App using a Docker image on Google Cloud Platform.

16.1.1 Prerequisites

Before you begin, ensure you have the following:

- A Google Cloud account. If you don't have one, you can sign up here.
- Google Cloud SDK installed on your local machine.
- Docker installed on your local machine.
- A GCP project created.

16.1.2 Step 1: Build the Docker Image

1. Navigate to the server directory where your `Dockerfile` is located.

   ```
   cd server
   ```

2. Build the Docker image. Replace `your-image-name` with a name for your image.

   ```
   docker build -t your-image-name .
   ```

16.1.3 Step 2: Tag the Docker Image

Tag the Docker image for Google Container Registry (GCR). Replace `YOUR_PROJECT_ID` with your actual GCP project ID.

```
docker tag your-image-name gcr.io/YOUR_PROJECT_ID/your-image-name
```

16.1.4 Step 3: Push the Docker Image to Google Container Registry

Authenticate with Google Cloud:

```
gcloud auth login
```

Set your project ID:

```
gcloud config set project YOUR_PROJECT_ID
```

Push the Docker image to GCR:

```
docker push gcr.io/YOUR_PROJECT_ID/your-image-name
```

16.1.5 Step 4: Deploy the Docker Image to Google Cloud Run

1. Deploy the image to Cloud Run. This command will create a new service named `your-service-name` from the Docker image. Replace the service name and image name as needed.

   ```
   gcloud run deploy your-service-name --image gcr.io/YOUR_PROJECT_ID/your-image-name --platf
   ```

2. Follow the prompts to select a region and allow unauthenticated invocations (if desired).

16.1.6 Step 5: Access Your Deployed App

Once the deployment is complete, you will receive a URL where your Cancer Prediction App is hosted. You can access your app by navigating to this URL in your web browser.

16.1.7 Cleanup

To avoid incurring charges, remember to delete your Cloud Run service when you no longer need it:

```
gcloud run services delete your-service-name --platform managed
```

17 VIII. Conclusion

Le déploiement de l'application de prédiction du cancer sur Google Cloud Platform (GCP) en utilisant Docker et Cloud Run permet une solution évolutive et facilement accessible pour prédire le type de tumeur, le coût de traitement et le stade du cancer. Grâce à l'utilisation de modèles de machine learning comme l'ElasticNet pour la prédiction du coût et du type de tumeur, ainsi que le Random Forest Classifier pour le stade du cancer, l'application fournit des prédictions précises et fiables.

L'intégration de Docker et de Google Cloud Run permet une gestion simplifiée des déploiements et garantit que l'application peut être facilement mise à l'échelle en fonction des besoins, tout en offrant une disponibilité continue à travers un environnement cloud sécurisé.

En résumé, ce projet démontre l'efficacité de la combinaison des technologies de machine learning, Docker et Cloud Computing pour développer des applications de santé intelligentes et accessibles à grande échelle.