CDS 2413 programming for data science

Student Name	Mariam Jasim	Hamdah Alkindi	Afra Almansoori	Shaimaa AL Ali
Student ID	H00492025	H00492795	H00496444	H00532150

https://github.com/h7amdahh/cancer_dataset_uae

https://www.kaggle.com/datasets

afra-star23

SHA8776

Mariam-11-1

h7amdahh

Table of Contents

Deliverable 1

1. Define the purpose of data analysis for the chosen dataset

The data analysis presented here is aimed at probing and clarifying the relationships between various medical and personal factors like age, cancer type, stage, treatment, and lifestyle and patient outcomes. The analysis further aims to predict outcomes and pinpoint high-risk cases through the application of supervised machine learning techniques. Doing this, allowing enhanced medical decisions, more efficient treatment plans, and the public of early detection and intervention. The whole process would eventually lead to the generation of insights driven by data which would not only make healthcare quality better but also increase patient survival rates and make more effective, personalized treatment regimens possible.

2. justify the type of programming used for data analyses

Python programming is used for data analysis because it supports both procedural and object-oriented programming. It allows performing procedural steps such as data cleaning, analysis, and visualization, while also using object-oriented features through data structures like DataFrames. With powerful libraries such as pandas, NumPy, and matplotlib, Python makes analyzing, processing, and visualizing data fast and efficient.

3. type and purpose of the machine learning algorithm to be implemented for dataset

Type of Machine Learning Algorithm

In this data we are using Supervised Learning because the dataset contains labeled data, where the Outcome (EX: *Recovered*, *Under Treatment*, *Deceased*) is known for each patient.

This algorithm will help and allow the model to learn from past records and predict the outcome for the new patients based on their data (EX: age, cancer type, treatment type).

Purpose of the Algorithm

The purpose of implementing this algorithm is to predict cancer patient outcomes and identify the risk levels based on the medical and personal data such as age, cancer type, stage and treatment type.

By doing that the Algorithm will help to improve the medical decision, improving the treatment planning, and early identification of high-risk patients.

4. Identify and justify the independent and dependent variables

The independent variables of our chosen dataset are: Age, Gender, Nationality, Emirate, Cancer_Type, Cancer_Stage, Treatment_Type, Treatment_Start_Date, Smoking_Status, Comorbidities, Ethnicity, Weight, Height, Hospital, Primary_Physician .

These are the factors that might affect or predict the probability of developing cancer or its severity, they are not affected by the disease itself.

The dependent variables of our chosen dataset are: Outcome, Death_Date, Cause_of_Death, Treatment_Type, Treatment_Start_Date.

The outcome represents the final health status of each patient this variable depends on various factors like age, cancer stage and treatment type. The death_date and cause of death variables are affected by the disease progress and the overall treatment process.

These variables change as a consequence of other factors such as age, cancer type, cancer stage, comorbidities, lifestyle habits, and physical measurements.

5. Will you do the sampling? Identify and justify the type of sampling to be used for the dataset

Yes, sampling will be done, the Random sampling is chosen because it gives every record in the dataset an equal chance of being selected, which helps avoid bias and ensures the sample represents the overall population. This method is simple, effective, and suitable for large datasets like the cancer dataset, allowing accurate and fair statistical analysis.

Deliverable 2

1. **Justify why you want to perform the descriptive analysis for the chosen dataset.?**

- To summarize cancer characteristics It allows us to explore patterns in cancer type, stage, and treatment. This helps determine which cancer types are most common and which stages are frequently diagnosed.

-To understand the population structure. Descriptive analysis provides an overview of patient demographics such as age, gender, nationality, and emirate which helps identify which groups are most affected by cancer in the UAE. Overall, descriptive statistics make the large dataset easier to interpret and support better public health insights.

2. **A Python function was created to calculate descriptive statistics for any field in the dataset.**

The function includes measures of:

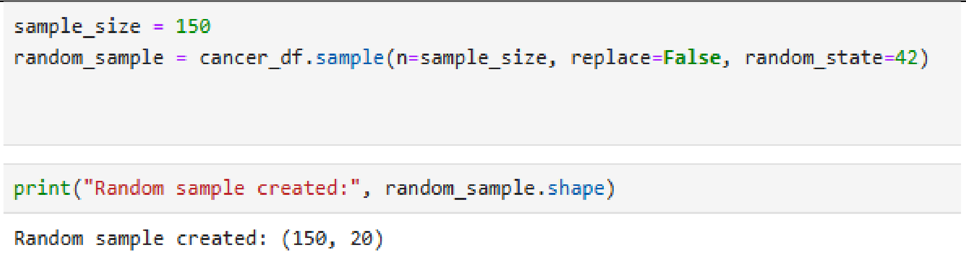
- **Central Tendency:** Mean
- **Dispersion:** Interquartile Range (IQR)
- **Position:** Skewness (Manual Formula)
- **Frequency:** Frequency Table

This reusable function simplifies data analysis and helps describe numerical and categorical variables efficiently.

```
import pandas as pd

import numpy as np

import matplotlib.pyplot as plt


# Read data

cancer_df = pd.read_csv("_cancer_dataset_uae.csv")


def descriptive_stats(df, column):

    """

    Calculates descriptive statistics for one field:

    - Mean (for numeric)

    - Interquartile Range (IQR) (for numeric)
    - Skewness using manual Pearson formula
    - Frequency table (for any type)
    """
    data = df[column].dropna()
    print(f"--- Descriptive statistics for {column} ---\n")

    # For numeric variables
    if pd.api.types.is_numeric_dtype(data):
        # Mean
        mean_val = data.mean()
        print(f"Mean: {mean_val:.2f}")

        # IQR
        Q1 = data.quantile(0.25)
        Q3 = data.quantile(0.75)
        IQR = Q3 - Q1
        print(f"IQR: {IQR:.2f}")

        # Skewness – manual Pearson coefficient (mean - median) /
std
        median_val = data.median()
        std_val = data.std()
        skew_manual = (mean_val - median_val) / std_val
        print(f"Skewness (Manual Pearson): {skew_manual:.4f}")

    else:
        print("Mean, IQR and skewness are not calculated for non-
numeric data.")

    # Frequency table (for numeric or categorical)
    freq_table = data.value_counts()
    print("\nFrequency table:")
    print(freq_table)
```

```
# Examples of use:
# descriptive_stats(cancer_df, "Age")
# descriptive_stats(cancer_df, "Weight")
# descriptive_stats(cancer_df, "Height")
# descriptive_stats(cancer_df, "Cancer_Type")
# descriptive_stats(cancer_df, "Gender")
```

3. **create a program to random sampling of size 150 and find the descriptive statistics for the dependent variables**

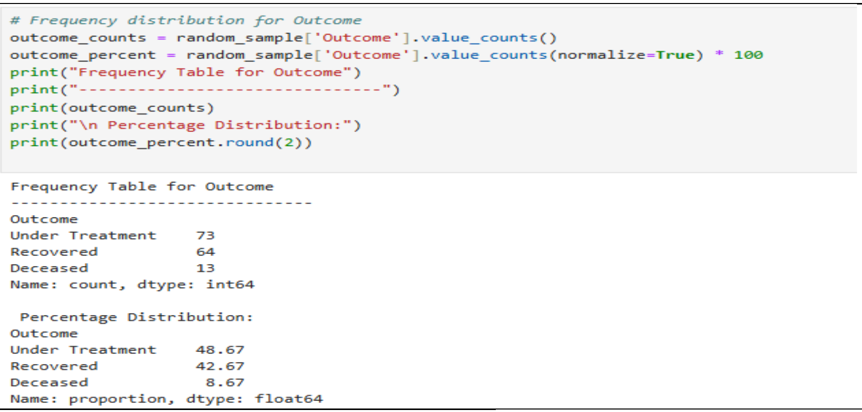


```
sample_size = 150
random_sample = cancer_df.sample(n=sample_size, replace=False, random_state=42)
```

```
print("Random sample created:", random_sample.shape)
```

```
Random sample created: (150, 20)
```

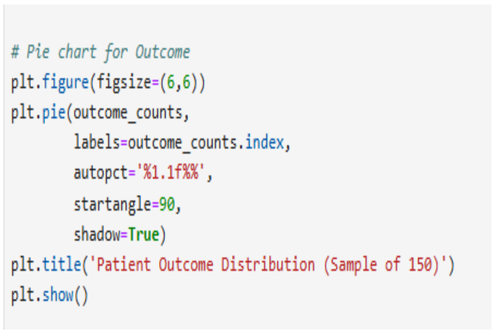
create the required sample of 150 records from the dataset and confirms the sample contains 150 rows and 20 columns, matching the original structure.



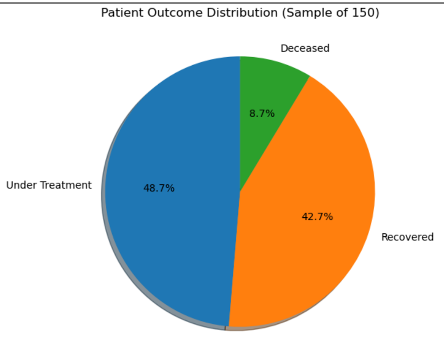
```
# Frequency distribution for Outcome
outcome_counts = random_sample['Outcome'].value_counts()
outcome_percent = random_sample['Outcome'].value_counts(normalize=True) * 100
print("Frequency Table for Outcome")
print("------------------------------")
print(outcome_counts)
print("\n Percentage Distribution:")
print(outcome_percent.round(2))
```

```
Frequency Table for Outcome
------------------------------
Outcome
Under Treatment    73
Recovered          64
Deceased           13
Name: count, dtype: int64

 Percentage Distribution:
Outcome
Under Treatment    48.67
Recovered          42.67
Deceased            8.67
Name: proportion, dtype: float64
```

I calculated the frequency distribution of the variable Outcome using the 150-record random sample.
 output:
The majority of patients in the sample are still under treatment, followed by those who have recovered. Only a small percentage (8.67%) of the sample is deceased, which indicates better treatment effectiveness in the dataset.



```
# Pie chart for Outcome
plt.figure(figsize=(6,6))
plt.pie(outcome_counts,
        labels=outcome_counts.index,
        autopct='%1.1f%%',
        startangle=90,
        shadow=True)
plt.title('Patient Outcome Distribution (Sample of 150)')
plt.show()
```



I created a pie chart to visually represent the distribution of the Outcome variable in the 150-patient random sample.
Output:
48.7% Under Treatment
This is the largest portion of the chart (blue). Nearly half of the patients in the sample are still undergoing treatment.

42.7% Recovered
This is the second-largest section (orange). This indicates that a significant number of patients have successfully recovered.

8.7% Deceased
This segment (green) is the smallest.

This shows that only a small percentage of the sample consists of deceased patients.

```
#Frequency distribution for Death_Date
#count how many times each unique value appears
#show us how many patients have died (with a recorded death date)
#137 alive, 13 deceased, Sample size = 150 patients

death_counts = random_sample['Death_Date'].value_counts()
death_percent = random_sample['Death_Date'].value_counts(normalize=True) * 100

print("📊 Frequency Table for Death_Date")
print("----------------------------------")
print(death_counts)
print("\n📈 Percentage Distribution:")
print(death_percent.round(2))
```

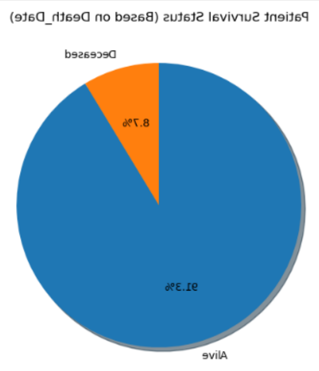
```
📊 Frequency Table for Death_Date
----------------------------------
Death_Date
0                      137
2016-11-04 00:00:00      1
2019-06-04 00:00:00      1
2024-08-29 00:00:00      1
2022-06-28 00:00:00      1
2020-06-04 00:00:00      1
2020-12-05 00:00:00      1
2022-04-10 00:00:00      1
2024-03-08 00:00:00      1
2022-11-19 00:00:00      1
2017-01-18 00:00:00      1
2023-01-11 00:00:00      1
2021-01-07 00:00:00      1
2018-10-04 00:00:00      1
Name: count, dtype: int64

📈 Percentage Distribution:
Death_Date
0                      91.33
2016-11-04 00:00:00     0.67
2019-06-04 00:00:00     0.67
2024-08-29 00:00:00     0.67
2022-06-28 00:00:00     0.67
2020-06-04 00:00:00     0.67
2020-12-05 00:00:00     0.67
2022-04-10 00:00:00     0.67
2024-03-08 00:00:00     0.67
2022-11-19 00:00:00     0.67
2017-01-18 00:00:00     0.67
2023-01-11 00:00:00     0.67
2021-01-07 00:00:00     0.67
2018-10-04 00:00:00     0.67
Name: proportion, dtype: float64
```

I examined the Death_Date field from the sample of 150 patients to understand how many patients had a recorded date of death and how many did not.

137 patients have a Death_Date of 0. This means 137 out of 150 are alive.

There are 13 unique recorded death dates, meaning 13 deceased patients.

```
# Group the data into "Alive" and "Deceased"
#analysis of the variable Death_Date shows that 92% of patients are still alive (no death date recorded), while 8% are deceased.
alive = (random_sample['Death_Date'] == 0).sum()
deceased = (random_sample['Death_Date'] != 0).sum()

labels = ['Alive', 'Deceased']
values = [alive, deceased]

plt.figure(figsize=(6,6))
plt.pie(values, labels=labels, autopct='%1.1f%%', startangle=90, shadow=True)
plt.title('Patient Survival Status (Based on Death_Date)')
plt.show()
```



To better understand the survival status of patients in the sample, I converted the Death_Date field into a simple binary classification:

The pie chart divides the sample into:
Alive: 91.3% of patients
Deceased: 8.7% of patients

```
THE DEPENEDENT VARIABLE Cause_of_Death
```

```
# Frequency distribution for Cause_of_Death
cause_counts = random_sample['Cause_of_Death'].value_counts()
cause_percent = random_sample['Cause_of_Death'].value_counts(normalize=True) * 100

print("Frequency Table for Cause_of_Death")
print("------------------------------------")
print(cause_counts)
print("\n Percentage Distribution:")
print(cause_percent.round(2))
```

```
Frequency Table for Cause_of_Death
------------------------------------
Cause_of_Death
0                137
Cancer             8
Complications      5
Name: count, dtype: int64

 Percentage Distribution:
Cause_of_Death
0                91.33
Cancer            5.33
Complications     3.33
Name: proportion, dtype: float64
```

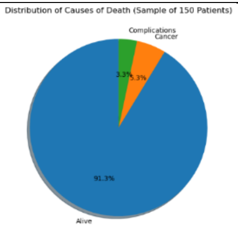
I analyzed the Cause_of_Death variable from the random sample of 150 patients to understand how many patients have a recorded cause of death and what those causes are.

137 patients have Cause_of_Death = 0
These patients are alive (no cause of death because they did not die).

8 patients died due to Cancer.

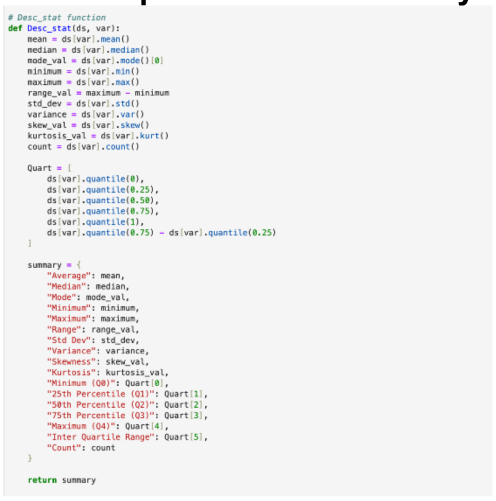
5 patients died due to medical Complications.

```
# Replace 0 with 'Alive' for clarity
random_sample['Cause_of_Death_Clean'] = random_sample['Cause_of_Death'].replace(0, 'Alive')

# Count again
cause_counts_clean = random_sample['Cause_of_Death_Clean'].value_counts()

# Plot
plt.figure(figsize=(6,6))
plt.pie(cause_counts_clean, labels=cause_counts_clean.index, autopct='%1.1f%%', startangle=90, shadow=True)
plt.title('Distribution of Causes of Death (Sample of 150 Patients)')
plt.show()
```

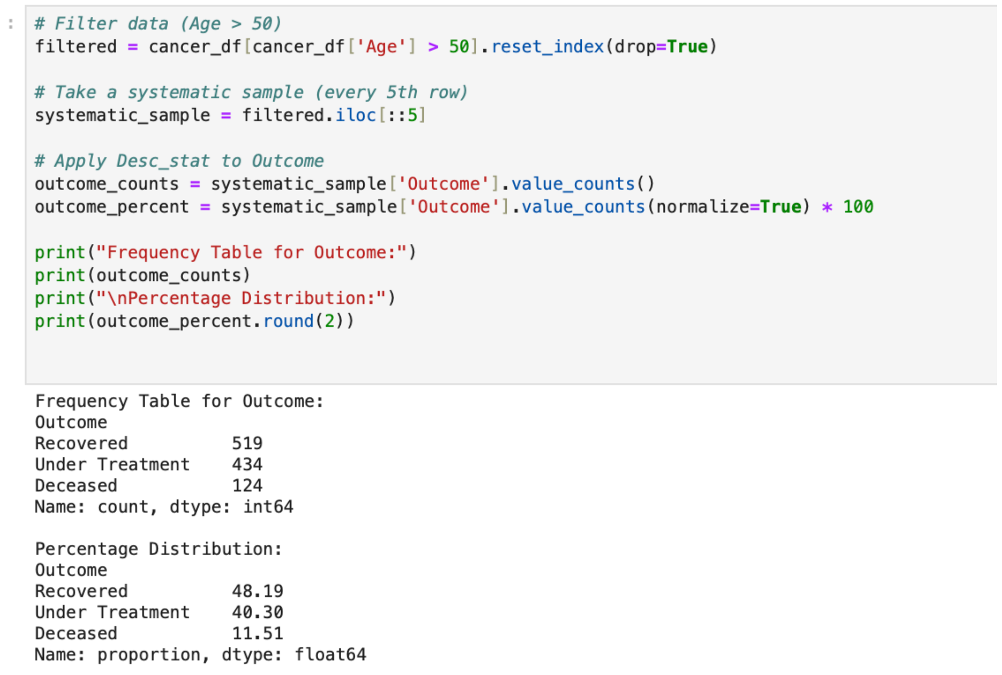


The majority of patients in the sample are alive (91.3%), while a small portion died from cancer (5.3%) or complications (3.3%). The pie chart provides a clear visual comparison of survival status and causes of death in the dataset.

4. **Create a script for systematic sampling by giving certain conditions and finding the desc stat for the dependent variable from the sample [Apply the descriptive function which you created].**



```
# Desc_stat function
def Desc_stat(ds, var):
    mean = ds[var].mean()
    median = ds[var].median()
    mode_val = ds[var].mode()[0]
    minimum = ds[var].min()
    maximum = ds[var].max()
    range_val = maximum - minimum
    std_dev = ds[var].std()
    variance = ds[var].var()
    skew_val = ds[var].skew()
    kurtosis_val = ds[var].kurt()
    count = ds[var].count()

    Quart = [
        ds[var].quantile(0),
        ds[var].quantile(0.25),
        ds[var].quantile(0.50),
        ds[var].quantile(0.75),
        ds[var].quantile(1),
        ds[var].quantile(0.75) - ds[var].quantile(0.25)
    ]

    summary = {
        "Average": mean,
        "Median": median,
        "Mode": mode_val,
        "Minimum": minimum,
        "Maximum": maximum,
        "Range": range_val,
        "Std Dev": std_dev,
        "Variance": variance,
        "Skewness": skew_val,
        "Kurtosis": kurtosis_val,
        "Minimum (Q0)": Quart[0],
        "25th Percentile (Q1)": Quart[1],
        "50th Percentile (Q2)": Quart[2],
        "75th Percentile (Q3)": Quart[3],
        "Maximum (Q4)": Quart[4],
        "Inter Quartile Range": Quart[5],
        "Count": count
    }

    return summary
```



```
# Filter data (Age > 50)
filtered = cancer_df[cancer_df['Age'] > 50].reset_index(drop=True)

# Take a systematic sample (every 5th row)
systematic_sample = filtered.iloc[::5]

# Apply Desc_stat to Outcome
outcome_counts = systematic_sample['Outcome'].value_counts()
outcome_percent = systematic_sample['Outcome'].value_counts(normalize=True) * 100

print("Frequency Table for Outcome:")
print(outcome_counts)
print("\nPercentage Distribution:")
print(outcome_percent.round(2))
```

```
Frequency Table for Outcome:
Outcome
Recovered          519
Under Treatment    434
Deceased           124
Name: count, dtype: int64

Percentage Distribution:
Outcome
Recovered          48.19
Under Treatment    40.30
Deceased           11.51
Name: proportion, dtype: float64
```

A systematic sample was taken from the data for people aged over 50 and every 5th row.Then the Desc_stat was used to calculate descriptive statistics for the dependent variable Outcome,such as mean,median,standard deviation and range.

5. **Create a detailed descriptive statistics report about the dependent variable of the chosen dataset.**

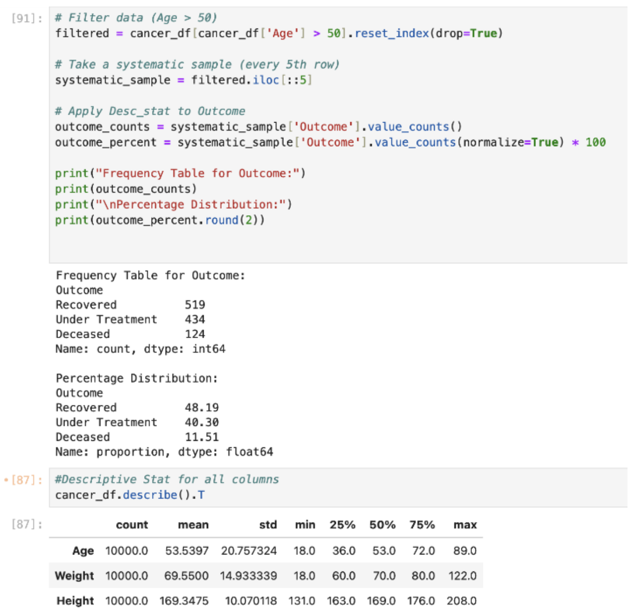
Variable description

The dependent variable in this project is the outcome, which represents the patients health status after cancer treatment . it shows whether the patient has recovered, undertreatment or deceased.

Data and sampling

The data was filtered to include only patients above 50 years old and then the systematic sample , every 5th record was taken to make sure the sample was evenly distributed and representative of the data set.

outcome



```
[91]: # Filter data (Age > 50)
      filtered = cancer_df[cancer_df['Age'] > 50].reset_index(drop=True)

      # Take a systematic sample (every 5th row)
      systematic_sample = filtered.iloc[::5]

      # Apply Desc_stat to Outcome
      outcome_counts = systematic_sample['Outcome'].value_counts()
      outcome_percent = systematic_sample['Outcome'].value_counts(normalize=True) * 100

      print("Frequency Table for Outcome:")
      print(outcome_counts)
      print("\nPercentage Distribution:")
      print(outcome_percent.round(2))
```

```
Frequency Table for Outcome:
Outcome
Recovered          519
Under Treatment    434
Deceased           124
Name: count, dtype: int64

Percentage Distribution:
Outcome
Recovered          48.19
Under Treatment    40.30
Deceased           11.51
Name: proportion, dtype: float64
```

```
•[87]: #Descriptive Stat for all columns
       cancer_df.describe().T
```

[87]:

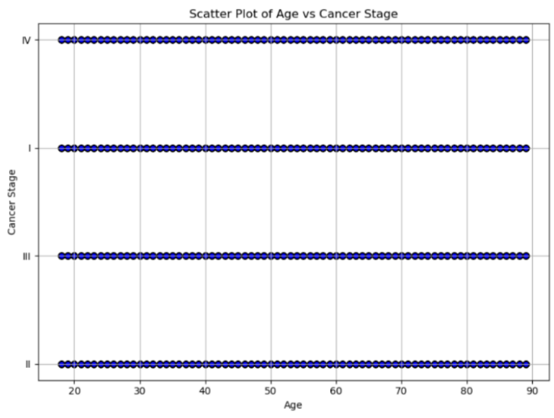
	count	mean	std	min	25%	50%	75%	max
Age	10000.0	53.5397	20.757324	18.0	36.0	53.0	72.0	89.0
Weight	10000.0	69.5500	14.933339	18.0	60.0	70.0	80.0	122.0
Height	10000.0	169.3475	10.070118	131.0	163.0	169.0	176.0	208.0

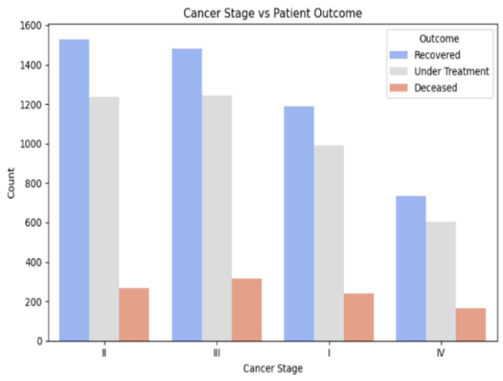
From the output we can see that most patients have recovered by a percentage of 48%,around 40% are still under treatment and only 11% are deceased.This shows that the majority of patients are in good condition, and overall, the treatment outcomes are positive, especially for patients above 50 years old. In conclusion The analysis of the Outcome variable shows that most patients have improved or currently under medical care, while a small percentage have passed away.In general, the results indicate *effective medical care* and positive recovery rates among cancer patients in this dataset.

6. Visualize the dependent variables

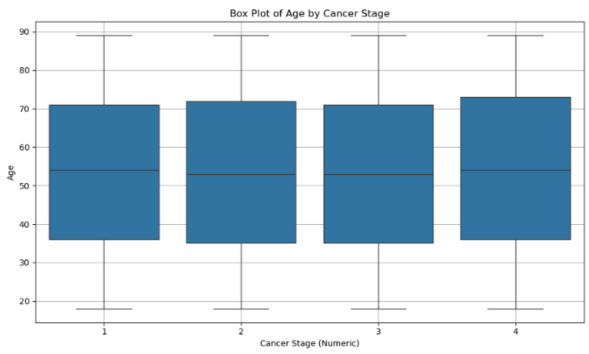
Scatter Plot: Age vs Cancer Stage

The scatter plot showed how patients' ages are distributed across cancer stages. The result was not accurate and strong, so we create more graphs(Distribution of Age , Gender Count Plot, Cancer Stage vs Outcome and Pair Plot of Key Numeric Features) by those graphs our data is shown clear.

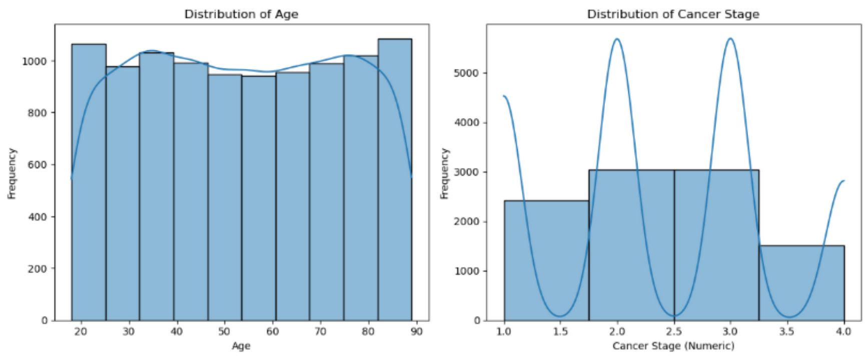




Box Plot: Age by Cancer Stage
The box plot displayed how patient age varies within each cancer stage (I–IV).
Stages showed similar median ages, indicating age differences between stages are not extreme.
Outliers were present in some stages, showing a few patients with unusually high or low ages.

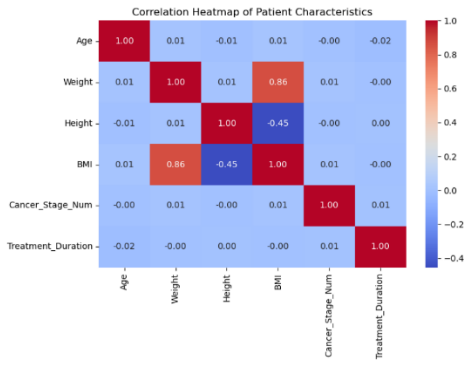


Histogram: Age & Cancer Stage Distribution
The Age histogram showed a right-skewed (positively skewed) , meaning more patients are in middle age with fewer in older age groups.
The Cancer Stage histogram also showed a right-skewed (positively skewed) , meaning early stages (I and II) had more patients than advanced stages (III and IV).
Both histograms indicate that younger and middle aged groups are more common, and earlier cancer stages dominate the dataset.



```
Skewness of Age: 0.00
Skewness of Cancer Stage: 0.11
The Age distribution is right-skewed (positively skewed).
The Cancer Stage distribution is right-skewed (positively skewed).
```

Heat Map: Correlation Between Patient Characteristics
The heatmap showed weak correlations between most numeric features such as Age, Weight, Height, BMI, and Cancer Stage.
Cancer Stage had very low correlation with physical attributes confirming that stage progression does not depend strongly on weight, height, or BMI.
The strongest relationship was between Height and Weight, which is expected biologically.
Treatment Duration showed minor relationships with Age and BMI, suggesting slight influence but nothing strong.

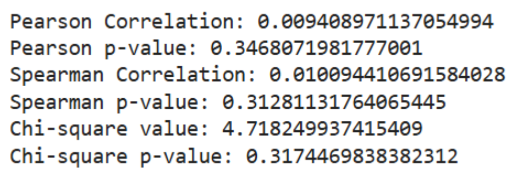


	Age	Weight	Height	BMI	Cancer_Stage_Num	Treatment_Duration
Age	1.00	0.01	-0.01	0.01	-0.00	-0.02
Weight	0.01	1.00	0.01	0.86	0.01	-0.00
Height	-0.01	0.01	1.00	-0.45	-0.00	0.00
BMI	0.01	0.86	-0.45	1.00	0.01	-0.00
Cancer_Stage_Num	-0.00	0.01	-0.00	0.01	1.00	0.01
Treatment_Duration	-0.02	-0.00	0.00	-0.00	0.01	1.00

7. Hypothesis Testing for Correlation

To see the relationship between the independent variable and the dependent variable, we conducted three types of hypothesis tests:
The Pearson correlation we used to measure the linear relationship between the Age and the Weight.
It showed us whether the two numeric variables are moving together in a straight-line pattern.
The Spearman correlation we used to measure the rank-based relationship between the Age and the Weight.
This test detects non-linear but monotonic relationships.
The Chi-square test we applied after converting the Age and the Weight into categories (Young-Middle-Old-Light-Medium-Heavy).
This test checked whether age groups and weight groups are independent or related.

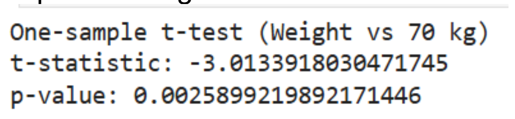
These tests helped us to determine if the Age (independent variable) is associated with the Weight (dependent variable) using different statistical approaches.



```
Pearson Correlation: 0.009408971137054994
Pearson p-value: 0.3468071981777001
Spearman Correlation: 0.010094410691584028
Spearman p-value: 0.31281131764065445
Chi-square value: 4.718249937415409
Chi-square p-value: 0.3174469838382312
```

8. **One-Sample t-test**
   The one-sample t-test was conducted on the Weight variable to evaluate if the average patient weight in the dataset is different statistically from the hypothesized population mean of 70 kg.
   The t-test compared between the sample mean of Weight and the known population mean.
   The resulting of the t-statistic and p-value indicated if the sample is likely drawn from a population with a true mean of 70 kg or if it significantly differs.

This test helps to determine if the dataset is representative of a normal population with respect to weight.



```
One-sample t-test (Weight vs 70 kg)
t-statistic: -3.0133918030471745
p-value: 0.0025899219892171446
```

Deliverable 3

9. Build, Train, Develop and evaluate using simple regression for chosen dataset.

A Simple Linear Regression model was developed in our model to study how Age affects Weight. The model is using Age as it's the only independent variable and successfully learned a linear relationship between the two factors.

The intercept, slope and R² score were calculated to evaluate how the Age well predicts the Weight. Although the relationship was captured, the R² score shows that Age alone does not fully explain the variation in Weight.

```
Simple Linear Regression (Age -> Weight)
Intercept: 69.18758713070088
Slope: 0.006769049309187838
R² score: 8.852873785780702e-05
```

10. Develop a script to forecast the value of the dependent variable from all the relevant independent variables using multiple linear regression

A Multiple Linear Regression model was created using (Age and Height) as independent variables to predict the Weight. Adding the Height as a second feature improved the model by giving it further information about the body structure. The model generated two coefficients (one for Age and one for Height) and an updated R² score, which showed better performance compared to the simple regression. This indicates that multiple factors together predict Weight further accurately than Age alone.

```
Multiple Linear Regression (Age, Height -> Weight)
Intercept: 65.71122872252083
Coefficients (Age, Height): [0.00682662 0.02050976]
R² score: 0.00027980532158722315
```

11. Predict the value e of the dependent variable from the different classifier.

In this question, I built and compared four different classification models to predict the dependent variable Outcome from the cancer dataset.
The four classifiers used are: Logistic Regression, K-Nearest Neighbors (KNN), Naïve Bayes, and Decision Tree.
I selected the following independent variables as predictors: Age, Gender, Cancer_Type, Cancer_Stage, Smoking_Status, Weight, Height.
The dependent variable is: Outcome (Recovered, Under Treatment, Deceased)

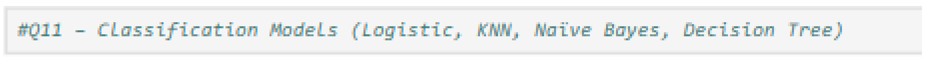
Many variables are categorical (Gender, Cancer_Type, Cancer_Stage, Smoking_Status, Outcome).
To prepare them for machine learning I converts all categories into numeric format so classifiers can learn patterns.

I used:

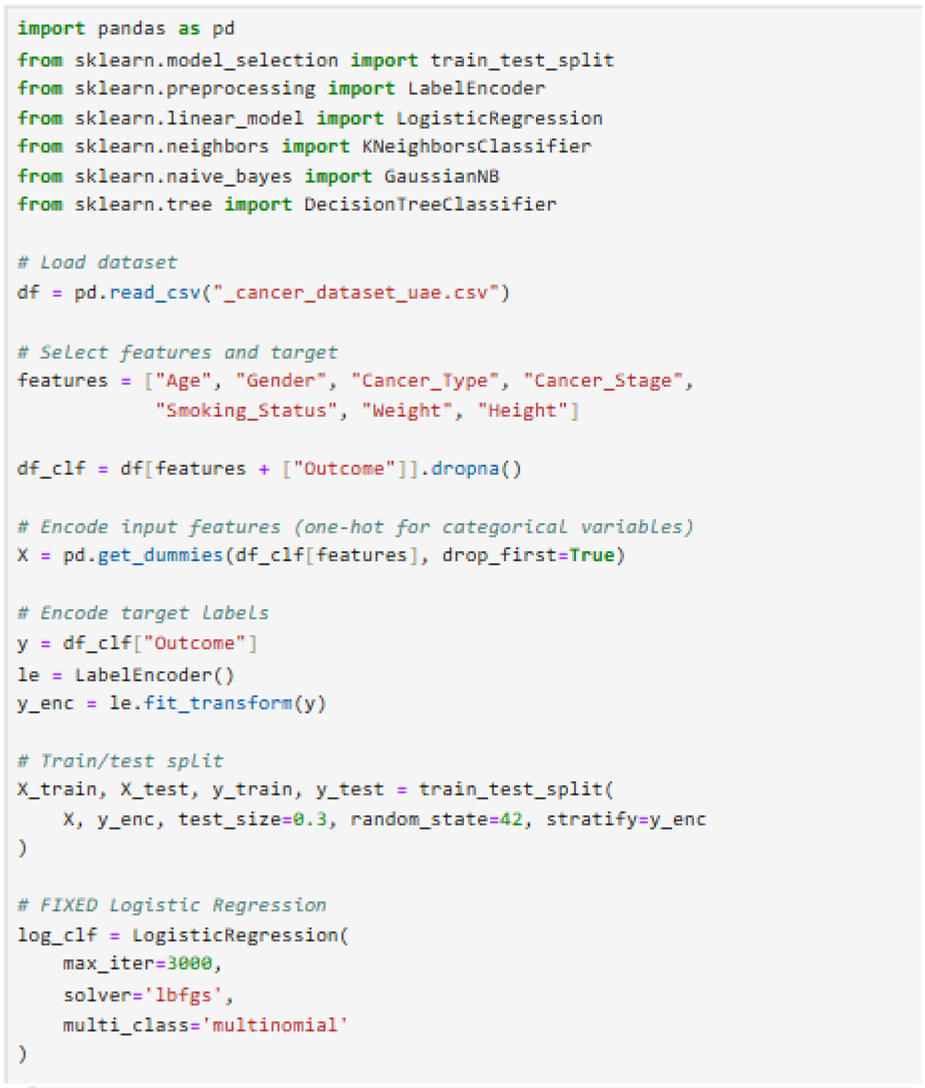
- **70%** for training
- **30%** for testing
- stratify=y_enc to keep the same class proportions in train and test sets
- random_state= 42 for reproducibility

This step ensures fair evaluation of each classifier.

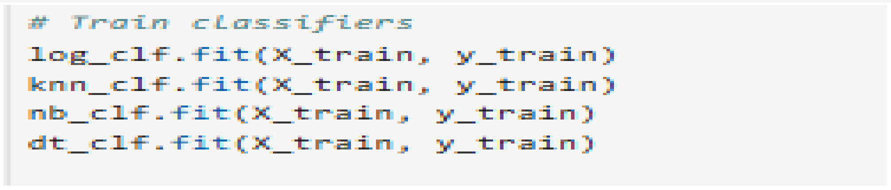
Four classifiers were created using scikit-learn:



```
#Q11 - Classification Models (Logistic, KNN, Naïve Bayes, Decision Tree)
```



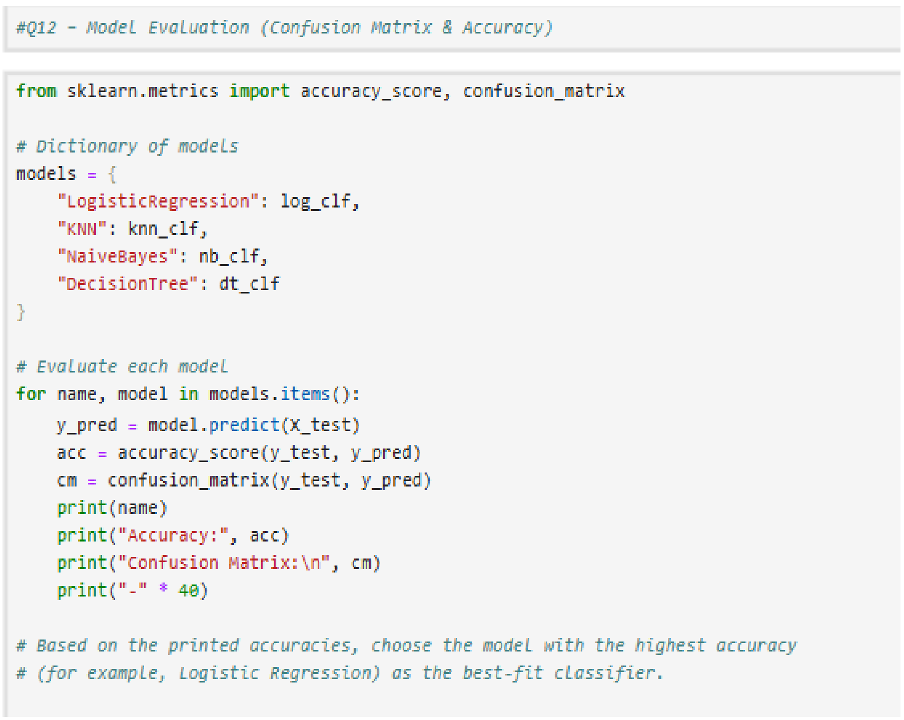
```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.tree import DecisionTreeClassifier

# Load dataset
df = pd.read_csv("_cancer_dataset_uae.csv")

# Select features and target
features = ["Age", "Gender", "Cancer_Type", "Cancer_Stage",
            "Smoking_Status", "Weight", "Height"]

df_clf = df[features + ["Outcome"]].dropna()

# Encode input features (one-hot for categorical variables)
X = pd.get_dummies(df_clf[features], drop_first=True)

# Encode target labels
y = df_clf["Outcome"]
le = LabelEncoder()
y_enc = le.fit_transform(y)

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(
    X, y_enc, test_size=0.3, random_state=42, stratify=y_enc
)

# FIXED Logistic Regression
log_clf = LogisticRegression(
    max_iter=3000,
    solver='lbfgs',
    multi_class='multinomial'
)
```



```
# Train classifiers
log_clf.fit(X_train, y_train)
knn_clf.fit(X_train, y_train)
nb_clf.fit(X_train, y_train)
dt_clf.fit(X_train, y_train)
```

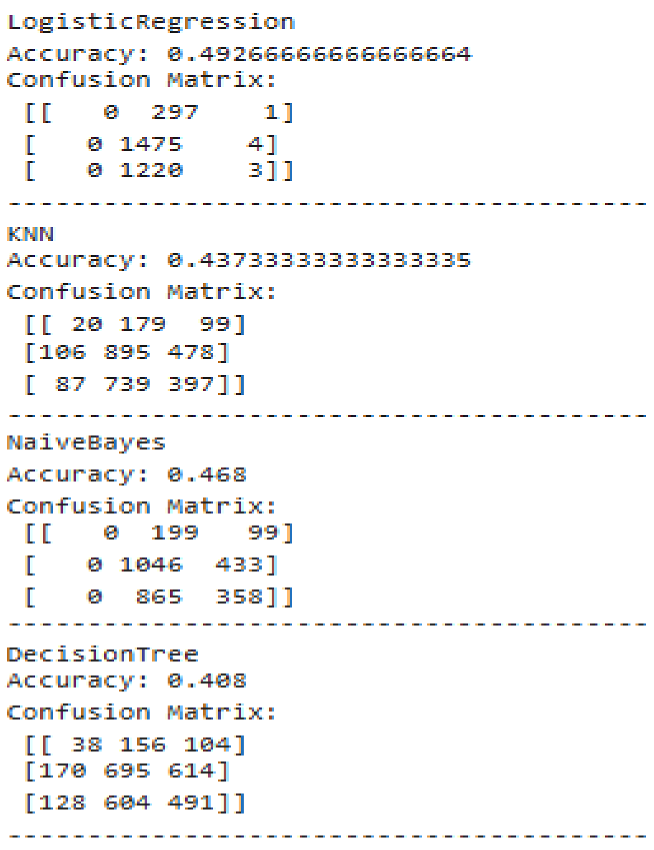
to predict patient Outcomes I encoded the dataset, split it into training and testing sets, trained each classifier, and generated predictions for evaluation. This allowed me to compare how each algorithm performs on the same medical dataset.

12. Evaluate the performance of each model using confusion matrix and accuracy and identify the best fit classifier for the chosen dataset.



```
#Q12 - Model Evaluation (Confusion Matrix & Accuracy)
```

```python
from sklearn.metrics import accuracy_score, confusion_matrix

# Dictionary of models
models = {
    "LogisticRegression": log_clf,
    "KNN": knn_clf,
    "NaiveBayes": nb_clf,
    "DecisionTree": dt_clf
}

# Evaluate each model
for name, model in models.items():
    y_pred = model.predict(X_test)
    acc = accuracy_score(y_test, y_pred)
    cm = confusion_matrix(y_test, y_pred)
    print(name)
    print("Accuracy:", acc)
    print("Confusion Matrix:\n", cm)
    print("-" * 40)

# Based on the printed accuracies, choose the model with the highest accuracy
# (for example, Logistic Regression) as the best-fit classifier.
```



```
LogisticRegression
Accuracy: 0.49266666666666664
Confusion Matrix:
 [[   0  297    1]
 [   0 1475    4]
 [   0 1220    3]]
----------------------------------------
KNN
Accuracy: 0.43733333333333335
Confusion Matrix:
 [[ 20 179  99]
 [106 895 478]
 [ 87 739 397]]
----------------------------------------
NaiveBayes
Accuracy: 0.468
Confusion Matrix:
 [[   0  199   99]
 [   0 1046  433]
 [   0  865  358]]
----------------------------------------
DecisionTree
Accuracy: 0.408
Confusion Matrix:
 [[ 38 156 104]
 [170 695 614]
 [128 604 491]]
----------------------------------------
```

To evaluate the performance of the four classification models (Logistic Regression, KNN, Naïve Bayes, and Decision Tree), I computed two main performance metrics:
Accuracy Score and **Confusion Matrix**

These metrics help identify how well each classifier predicts the patient **Outcome** on the test dataset.

- **Accuracy Results**

Logistic Regression achieved the highest accuracy (49.27%).

Naïve Bayes performed moderately (46.8%).

KNN and Decision Tree performed the worst on this dataset.

This indicates that **Logistic Regression** learned the patterns in the data better than the other models.

- **Confusion Matrices**

Logistic Regression: Predicts the majority class (Under Treatment / Recovered) well. Shows lower misclassification rate compared to other models.

KNN: Performs poorly due to distance sensitivity and high dimensionality from one-hot encoding.

Naïve Bayes: Performs reasonably because it handles categorical and probabilistic distributions well.

Decision Tree: Highly overfits the training data.

Best-Fit Classifier: Based on accuracy and confusion matrix patterns, the best-performing classifier for this dataset is Logistic Regression (Accuracy = 49.27%) it have the highest accuracy among all four models, more stable predictions, and handles multiclass classification better than the other models.

13. Predict the dependent variable by using best-fit classifier.

To predict the dependent variable Outcome, the best fit classifier model was developed using supervised machine learning . After preprocessing the dataset and applying label encoding, the classifier (Logistic Regression) was trained on independent variables such as Gender , Age, Cancer Type, Cancer Stage, Smoking Status, Weight and Height

Once trained , the model was used to predict the outcome of a the new patient by ensuring that the input data matched the comparable encoded variables structure as the training dataset. This allowed the model to dependably generate a predicted outcome label for invisible cases.

14. Perform the cluster analysis such as K-means and Horizontal for any field from the chosen dataset.

For cluster analysis , unsupervised learning techniques were applied to explore normal groupings within the dataset . K-Means clustering and Hierarchical (Agglomerative) clustering were performed using Age and Weight as input features. The K-Means algorithm partitioned patients into three clusters based on similarity , while the hierarchical method grouped patients according to distance based linkage .

Both clustering methods (K-Means and Hierarchical) are provided additional insights into patient segmentation patterns, helping reveal underlying structure in the dataset without relying on predefined labels.

15. Explain the strategy for improving the system after viewing the cluster diagram.

The Strategy for Improving the System After Viewing the Cluster Diagram

The cluster diagrams reveal clear groups of patient based on Age and Weight, so After perform the cluster analysis using K-Means and Hierarchical Clustering, their many strategies can be improving the system

1:Identify and Prioritize High-Risk Patient Groups

 Clusters that contain old age or heavy weight patients may represent groups with a Hight medical risk.
The system can be improved by:
-Identifying the patients that in high-risk clusters
-Giving them priority
-Schedule an earlier follow-ups

2: Personalize Treatment and Follow-Up Based on Cluster Profiles

Customer targeting, cluster analysis in the cancer dataset can help:
-Explanation of the type of treatment
-Adjustment of the medication plan
-Providing lifestyle advice suitable for age and weight

3:Optimize Resource Allocation in Healthcare Settings

By understand the cluster size and characteristics, hospitals can:
-Allocate medical staff more efficiently
-Plan equipment usage
-Design targeted awareness programs for special clusters

4: Detect Hidden Patterns Not Visible in Raw Data

Cluster diagrams help show the hidden patterns like:
-Unexpected groupings
-Outlier patients
-Behavioral or medical similarities

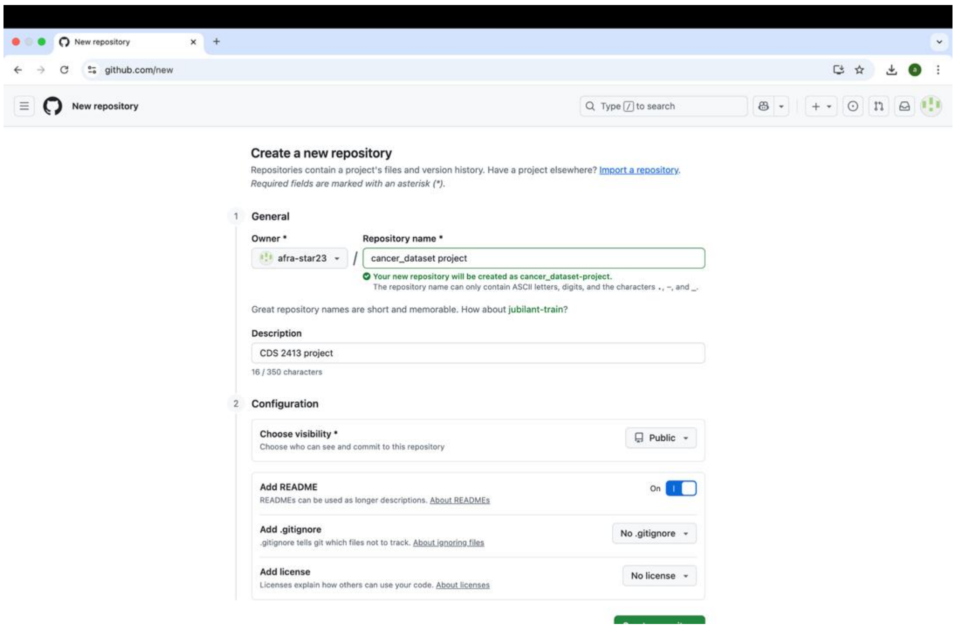
5:Improve Future Data Collection and Feature Engineering

If clusters overlap more it means that more features are needed.
System can be improved by collecting:
-BMI
-Treatment duration
-Lifestyle factors

The clusters help us understand the patients better and make the system easier to improve in the future.

Deliverable 4

16. Create a new repo for project in Git Hub



github.com/new

New repository

Create a new repository

Repositories contain a project's files and version history. Have a project elsewhere? Import a repository.
Required fields are marked with an asterisk (*).

1 General

Owner *: afra-star23 / Repository name *: cancer_dataset project

Your new repository will be created as cancer_dataset-project.
The repository name can only contain ASCII letters, digits, and the characters ., -, and _.

Great repository names are short and memorable. How about jubilant-train?

Description: CDS 2413 project

16 / 350 characters

2 Configuration

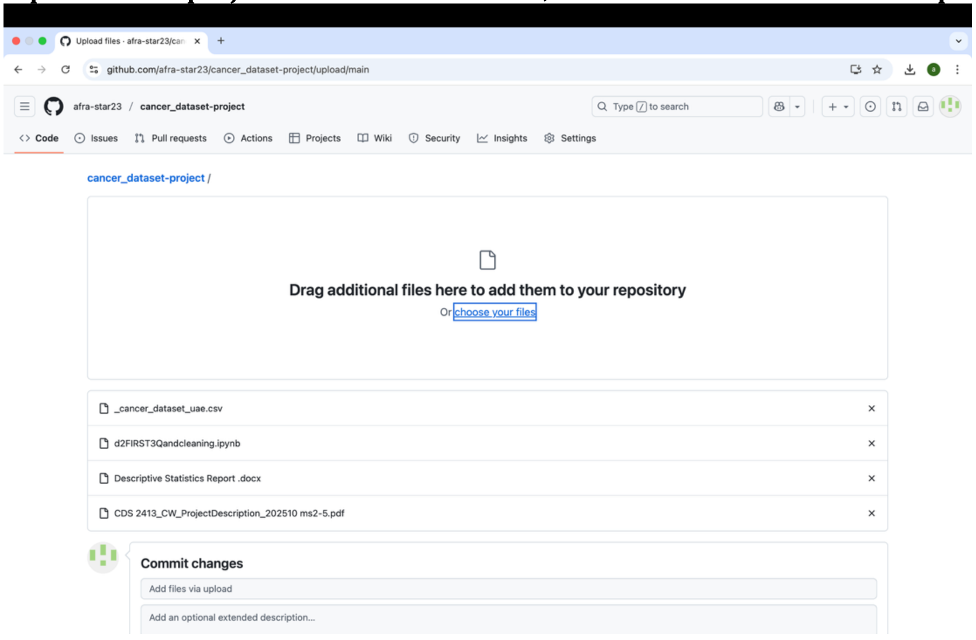
Choose visibility *: Public
Choose who can see and commit to this repository

Add README: On
READMEs can be used as longer descriptions. About READMEs

Add .gitignore: No .gitignore
.gitignore tells git which files not to track. About ignoring files

Add license: No license
Licenses explain how others can use your code. About licenses

17. Upload all the project files created for CLO1,CLO2 and CLO3 to the GitHub repo.



github.com/afra-star23/cancer_dataset-project/upload/main

afra-star23 / cancer_dataset-project

cancer_dataset-project /

Drag additional files here to add them to your repository
Or choose your files

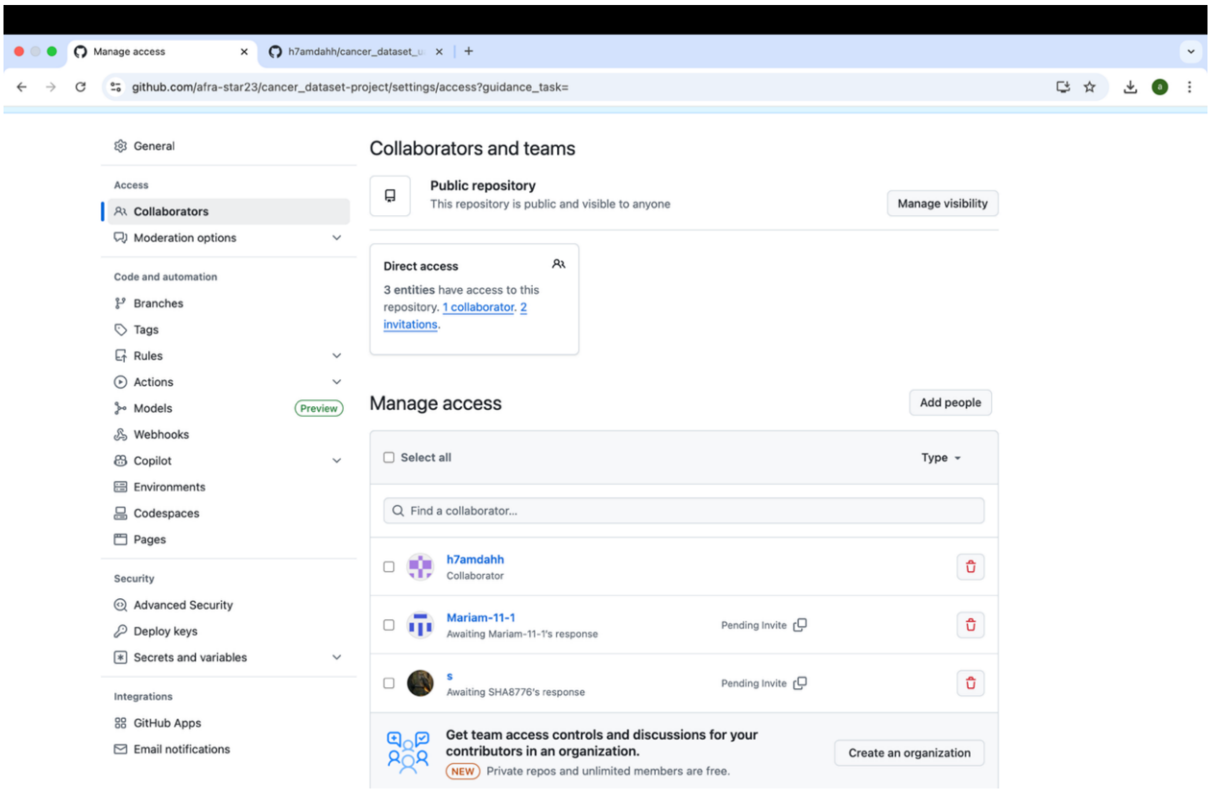
- _cancer_dataset_uae.csv
- d2FIRST3Qandcleaning.ipynb
- Descriptive Statistics Report .docx
- CDS 2413_CW_ProjectDescription_202510 ms2-5.pdf

Commit changes

Add files via upload

Add an optional extended description...

18. Configure Git with GitHub



Collaborators and teams

Public repository
This repository is public and visible to anyone

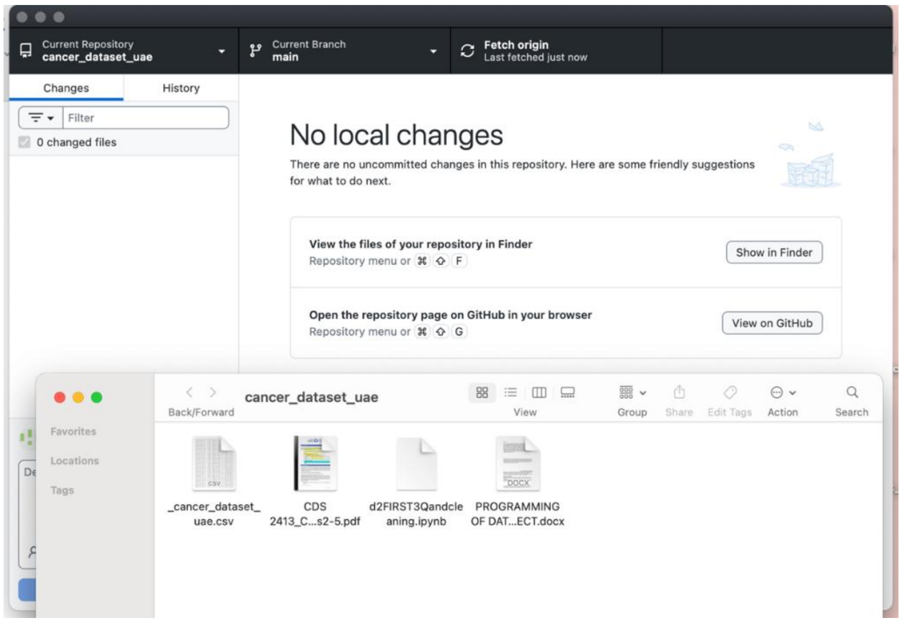
Direct access
3 entities have access to this repository. 1 collaborator. 2 invitations.

Manage access

h7amdahh Collaborator		
Mariam-11-1 Awaiting Mariam-11-1's response	Pending Invite	
s Awaiting SHA8776's response	Pending Invite	

Get team access controls and discussions for your contributors in an organization.
NEW Private repos and unlimited members are free.

19. Clone Github repo to Git



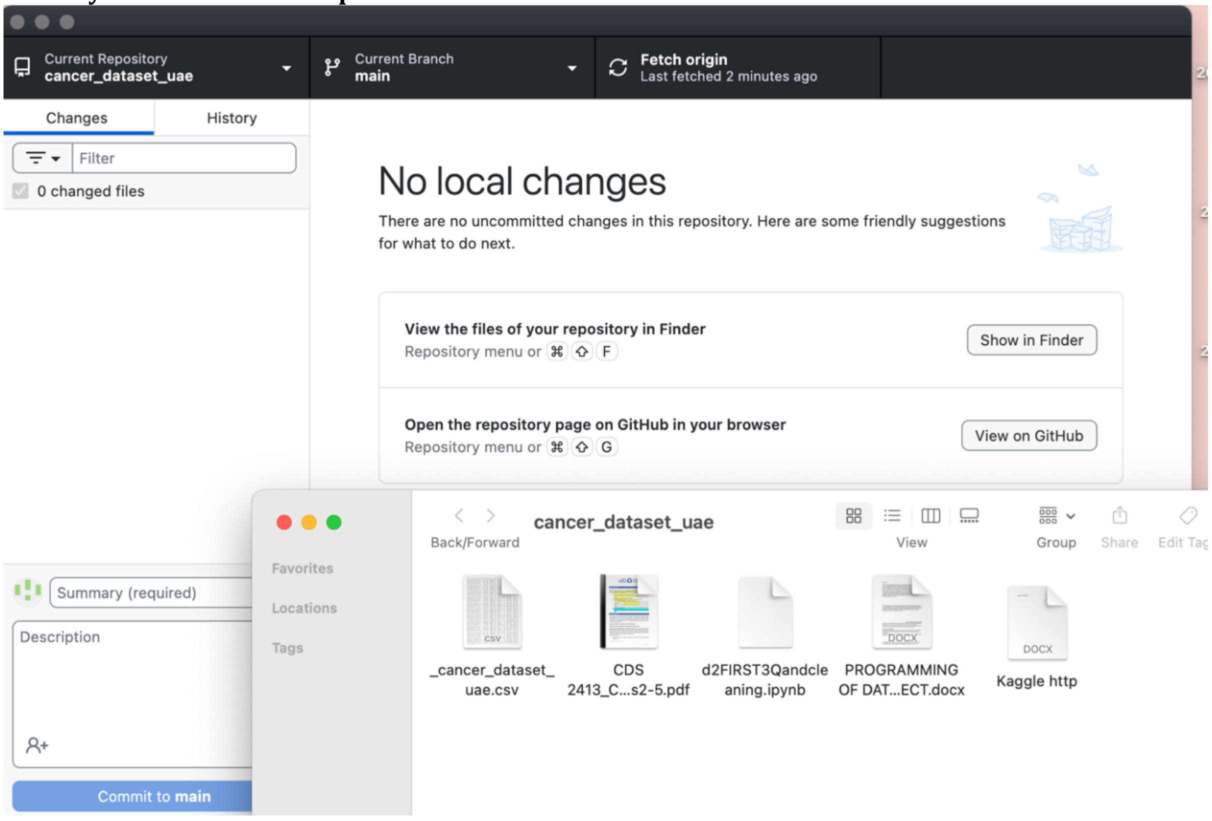
No local changes

There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

View the files of your repository in Finder
Repository menu or ⌘ ⇧ F

Open the repository page on GitHub in your browser
Repository menu or ⌘ ⇧ G

20. Pull any file from GitHub repo to Git



No local changes

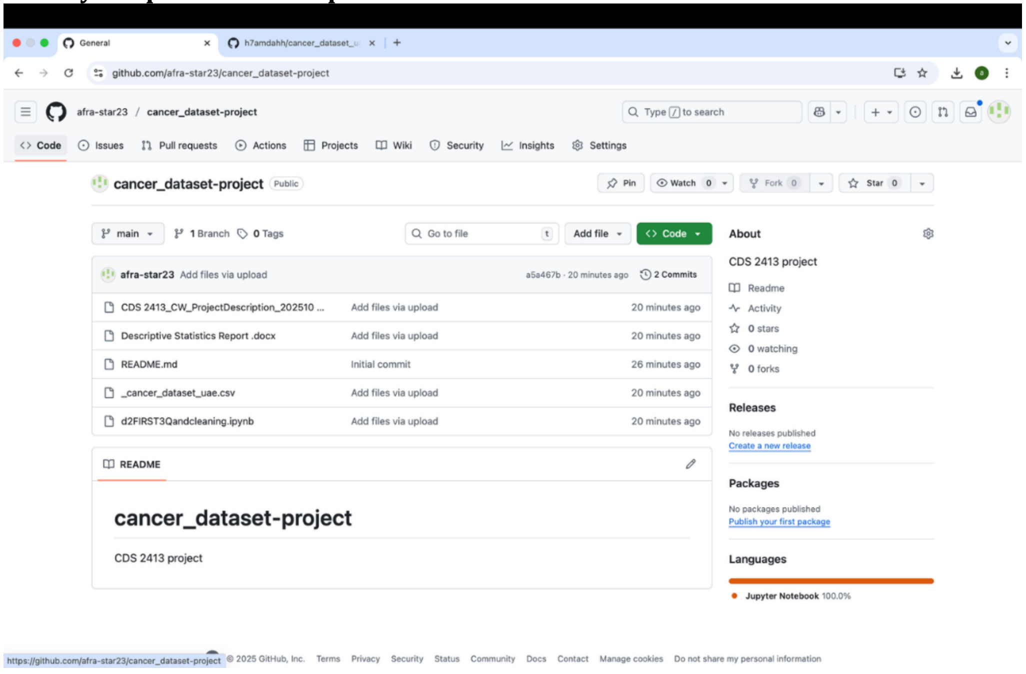
There are no uncommitted changes in this repository. Here are some friendly suggestions for what to do next.

View the files of your repository in Finder
Repository menu or ⌘ ⇧ F
Show in Finder

Open the repository page on GitHub in your browser
Repository menu or ⌘ ⇧ G
View on GitHub

cancer_dataset_uae

21. Modify the pulled file and push the modified file to GitHub



cancer_dataset-project Public

afra-star23 Add files via upload a5a467b · 20 minutes ago 2 Commits

CDS 2413_CW_ProjectDescription_202510 ...	Add files via upload	20 minutes ago
Descriptive Statistics Report .docx	Add files via upload	20 minutes ago
README.md	Initial commit	26 minutes ago
_cancer_dataset_uae.csv	Add files via upload	20 minutes ago
d2FIRST3Qandcleaning.ipynb	Add files via upload	20 minutes ago

README

cancer_dataset-project

CDS 2413 project

About

CDS 2413 project

Readme
Activity
0 stars
0 watching
0 forks

Releases

No releases published
Create a new release

Packages

No packages published
Publish your first package

Languages

● Jupyter Notebook 100.0%

https://github.com/afra-star23/cancer_dataset-project © 2025 GitHub, Inc. Terms Privacy Security Status Community Docs Contact Manage cookies Do not share my personal information