



UTM
UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING

SECB3203-01

PROGRAMMING FOR BIOINFORMATICS

TITLE:

Prediction of Lung Cancer using Recursive Feature Elimination

LECTURER:

DR. NIES HUI WEN

CLIENT:

Dr. Sharin Hazlin Binti Huspi

GROUP 4

GROUP MEMBERS:

NAME	MATRIC NUMBER
LEE RONG XIAN	A21EC0043
LU QI YAN	A21EC0049

ACKNOWLEDGEMENT

In preparing this project, we were in contact with many individuals whose invaluable contributions significantly enriched my understanding and thoughts. We would like to extend our heartfelt appreciation to our Programming For Bioinformatics lecturer, Dr. Nies Hui Wen, for her unwavering encouragement, guidance, constructive criticism, and friendship throughout this endeavour. Her expertise and insights have been instrumental in shaping the trajectory of this research.

We are equally grateful to my client, Dr. Sharin Hazlin Binti Huspi, for her generous guidance, valuable advice, and constant motivation. Her support has been instrumental in navigating the intricacies of this project, and her commitment to excellence has been a source of inspiration.

The collaborative efforts of these individuals, along with the countless researchers, academicians, and practitioners we have engaged with, have left an indelible mark on this work. Without their continued support, interest, and expertise, this project would not have reached the level of depth and quality as presented here.

ABSTRACT

Lung cancer poses a significant global health challenge, necessitating early detection for effective treatment. Leveraging advancements in bioinformatics and artificial intelligence, this research proposes the development of a predictive model for lung cancer utilizing Machine Learning techniques, specifically recursive elimination. With lung cancer being the leading cause of cancer-related deaths worldwide, this project aims to enhance prediction accuracy, focusing on the rapid and precise early detection of the disease.

The rise in lung cancer cases in Malaysia underscores the urgency of preventive measures and improved treatment strategies. Through the application of machine learning, this project seeks to contribute to the field by employing recursive feature elimination (RFE) for effective feature selection in the dataset. The objectives include predicting lung cancer based on patient symptoms, implementing RFE to systematically eliminate irrelevant features, and evaluating the effectiveness of RFE in selecting the most relevant features for accurate lung cancer prediction.

The problem statement addresses the challenges of predicting lung cancer and enhancing prediction model accuracy. The project scope involves symptom-based prediction, acknowledging potential symptom irrelevance, and concentrating on feature selection analysis for lung cancer prediction. The anticipated outcome is a novel RFE-based lung cancer prediction system, offering high-accuracy models for identifying individuals at risk of lung cancer. This research not only contributes to the early detection of lung cancer but also provides insights into the integration of artificial intelligence into clinical practices for automated diagnosis.

TABLE OF CONTENTS

	TITLE	PAGE
	ACKNOWLEDGEMENT	i
	ABSTRACT	ii
	TABLE OF CONTENTS	iii
	LIST OF TABLES	iv
	LIST OF FIGURES	v
	LIST OF APPENDICES	vi
CHAPTER 1	INTRODUCTION	1
1.1	Problem Background	1
1.2	Problem Statement	2
1.3	Objectives	2
1.4	Scopes	2
1.5	Conclusion	2
CHAPTER 2	DATA COLLECTION AND PRE-PROCESSING	3
2.1	Importing Dataset	3
2.1.1	Import Packages	3
2.1.2	Import Dataset	3
2.2	Data pre-processing and Exploratory data analysis	4
2.2.1	Data Formatting	4
2.2.2	Data Duplication Identifying and Handling	6
2.2.3	Missing Values Identifying and Handling	7
2.2.4	Data Normalization	8
2.2.5	Data Binning	10

2.2.6	Indicator Variable	10
2.2.7	Export preprocessed data	12
2.2.8	Descriptive Analysis	13
2.2.9	Mean, Variance, and Standard Deviation of AGE	14
2.2.10	Mean, Variance, and Standard Deviation of GENDER	15
2.2.11	Mean, Variance, and Standard Deviation of LUNG CANCER	16
2.2.12	Mode	17
2.2.13	Describe data(columns)	18
2.2.14	Boxplot	18
2.2.15	Correlation	19
2.3	Software and hardware requirements	22
CHAPTER 3	FLOWCHART OF PROPOSED APPROACH	23
3.1	Model Development	24
3.2	Model Evaluation	29
CHAPTER 4	RESULTS, TESTING, AND VALIDATION	32
4.1	Feedback	35
CHAPTER 5	CONCLUSION	36
CHAPTER 6	REFERENCES	36

LIST OF TABLES

TABLE NO.	TITLE	PAGE
Table 1.0	Training and Testing Set with respective accuracy	32

LIST OF FIGURES

FIGURE NO.	TITLE	PAGE
Figure 2.1	Import Packages	3
Figure 2.2	Import Dataset	3
Figure 2.3	Attributes	4
Figure 2.4	Total number of rows and columns	4
Figure 2.5	Columns of the dataset	5
Figure 2.6	Datatype for each attribute	5
Figure 2.7	Data Duplication Identifying and Handling	6
Figure 2.8	Columns of the dataset	6
Figure 2.9	Missing Values Identifying and Handling	7
Figure 2.10	Data Normalization	8
Figure 2.11	Column Rearrangement	9
Figure 2.12	Columns of the dataset	9
Figure 2.13	Data Binning	10
Figure 2.14	Indicator Variable	10
Figure 2.15	Columns of the dataset	11
Figure 2.16	Export preprocessed data	12
Figure 2.17	Import preprocessed datasets	13
Figure 2.18	Mean age	14
Figure 2.19	Variance of Age	14
Figure 2.20	Standard deviation of Age	15
Figure 2.21	Mean of Gender	15

Figure 2.22	Variance of Gender	16
Figure 2.23	Standard Deviation of Gender	16
Figure 2.24	Mean for Lung Cancer	16
Figure 2.25	Variance of Lung Cancer	17
Figure 2.26	Standard deviation of Lung Cancer	17
Figure 2.27	Mode	17
Figure 2.28	Columns of the dataset	18
Figure 2.29	Construction of boxplot	18
Figure 2.30	Boxplot of Age	19
Figure 2.31	Construction of Correlation	19
Figure 2.32	Result of Correlation	20
Figure 2.33	Construction of Heatmap	20
Figure 2.34	Heatmap	21
Figure 3.1	Flowchart	23
Figure 3.2	Import Packages	24
Figure 3.3	Replacement of categorical variables(Gender)	24
Figure 3.4	Replacement of categorial variables(Age group, Lung cancer)	25
Figure 3.5	Export Preprocessed Data	25
Figure 3.6	Dropping of Unwanted Columns	26
Figure 3.7	Model training	26
Figure 3.8	RFE implementation	27
Figure 3.9	Getting List of Features Selected	27
Figure 3.10	List of Features Selected	28
Figure 3.11	Getting Rank of Selected Features	28
Figure 3.12	Rank of Selected Features	29
Figure 3.13	StratifiedKFold Implementation	29

Figure 3.14	Cross-validation	30
Figure 3.15	Accuracy	30
Figure 3.16	Precision	30
Figure 3.17	Recall	31
Figure 4.1	Confusion Matrix Implementation	32
Figure 4.2	Region of Confusion Matrix	33
Figure 4.3	Region of Confusion Matrix	33
Figure 4.4	Confusion matrix's Accuracy	34
Figure 4.5	Confusion matrix's Precision	34
Figure 4.6	Confusion matrix's Recall	34

LIST OF APPENDICES

APPENDIX	TITLE	PAGE
Appendix A	Link For GitHub, E-Portfolio	38
Appendix B	Overall Discussion(Requested By Client)	39

Title: Prediction of Lung Cancer using Recursive Feature Elimination

1.0 Introduction

Lung cancer is a prevalent and potentially life-threatening disease, with early detection being crucial for successful treatment. Over the years, advancements in technology, particularly in the field of bioinformatics and artificial intelligence, have opened up new possibilities for more accurate and efficient skin cancer diagnosis. This proposal outlines a research project aimed at developing a predictive model for skin cancer using Machine Learning techniques, specifically recursive elimination.

1.1 Problem Background

Nowadays, Lung cancer is the leading cause of cancer deaths worldwide [2]. According to the American Cancer Society, lung cancer originates in the lungs where cells uncontrollably grow. The two main types are Non-Small Cell Lung Cancer (NSCLC) and Small Cell Lung Cancer (SCLC). NSCLC, comprising 80-85% of cases, includes adenocarcinoma, linked to smokers but also seen in non-smokers, squamous cell carcinoma associated with smoking, and large cell carcinoma that grows rapidly. SCLC, 10-15% of cases, known as oat cell cancer, spreads quickly beyond the lungs. While it responds well to chemotherapy and radiation therapy, recurrence is common.[2]

In Malaysia, The World Health Organisation (WHO) forecast that the prevalence of lung cancer in Malaysia could double by 2040. According to its estimation for past and future trends in terms of total cases per year, lung cancer cases increased from 4,403 in 2012 to 4,686 in 2018. The number is expected to increase to 9,679 by 2040. [4] Hence, it is crucial to discover methods for preventing and treating lung cancer.

Over the years, researchers have explored the application of machine learning to assist in the detection and prediction of cancer. In this project, we are going to focus on rapid and highly accurate early detection of the disease to ensure an increased survival rate. Thus, we are implementing one of the Machine learning approaches which is recursive feature elimination (RFE) for feature selection in our dataset[3] by eliminating the weakest feature until we get the number of features we want.

1.2 Problem Statement

1. How to predict lung cancer?
2. How to enhance the accuracy of lung cancer prediction models?
3. What is the effectiveness of RFE in selecting the most informative features for lung cancer prediction?

1.3 Objectives

1. To predict lung cancer based on the symptoms of the patients
2. To develop and apply the RFE technique to systematically eliminate irrelevant features to enhance the accuracy of lung cancer prediction models.
3. To evaluate the effectiveness of RFE in selecting the most relevant features for lung cancer prediction

1.4 Scopes

1. Assuming that the lung cancer prediction will be solely based on the symptoms in the dataset.
2. Some symptoms may be irrelevant to lung cancer.
3. Focusing on the analysis of feature selection for lung cancer prediction.

1.5 Conclusion

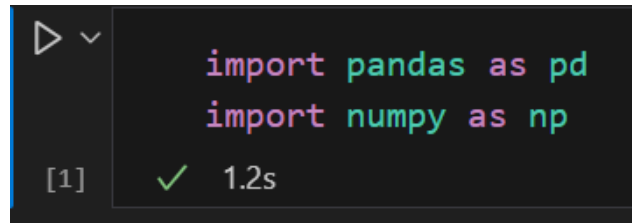
In conclusion, the outcome of this project is the development of an RFE-based lung cancer prediction system that includes high-accuracy prediction models for detecting whether a patient is at risk of lung cancer. Besides, insights into the potential of AI for automated lung cancer diagnosis and its integration into clinical practice will be obtained through this project.

2.0 Data collection and pre-processing

2.1 Importing dataset

2.1.1 Import Packages

The necessary packages such as pandas and numpy are imported. These packages are commonly used in the analysis and manipulation of data in Python.



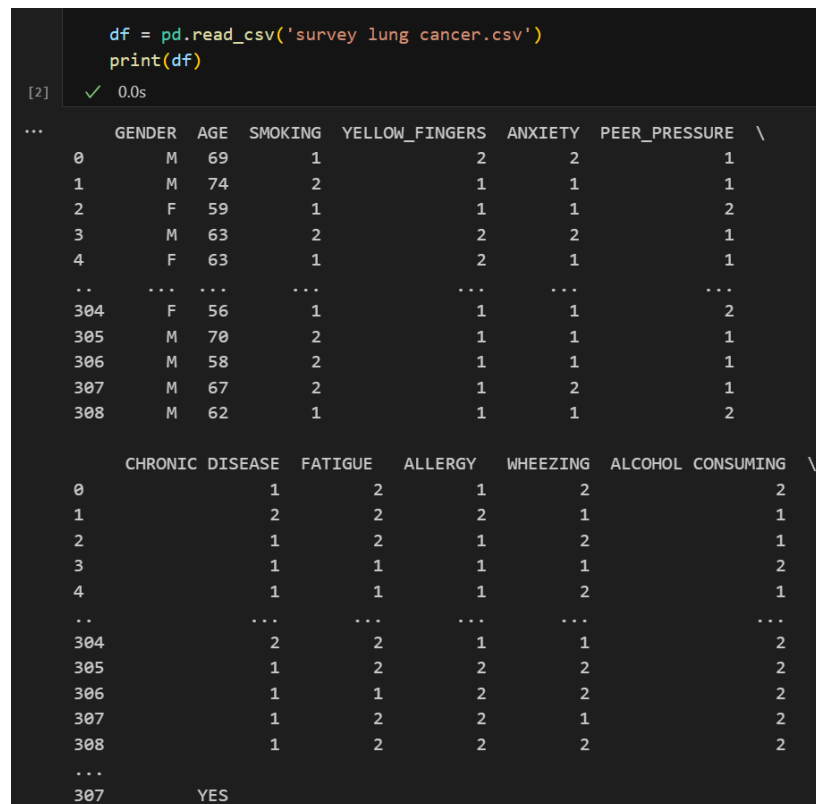
```
import pandas as pd
import numpy as np
```

[1] ✓ 1.2s

Figure 2.1 Import Packages

2.1.2 Import Dataset

The lung cancer dataset is imported by reading a CSV file. The `print(df)` method is used to display the imported dataset.



```
df = pd.read_csv('survey_lung_cancer.csv')
print(df)
```

[2] ✓ 0.0s

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	\
0	M	69	1	2	2	1	1
1	M	74	2	1	1	1	1
2	F	59	1	1	1	2	2
3	M	63	2	2	2	1	1
4	F	63	1	2	1	1	1
...
304	F	56	1	1	1	2	2
305	M	70	2	1	1	1	1
306	M	58	2	1	1	1	1
307	M	67	2	1	2	1	1
308	M	62	1	1	1	2	2
	CHRONIC_DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL_CONSUMING	\	
0	1	2	1	2	2	2	
1	2	2	2	1	1	1	
2	1	2	1	2	2	1	
3	1	1	1	1	2	2	
4	1	1	1	2	1	1	
...	
304	2	2	1	1	2	2	
305	1	2	2	2	2	2	
306	1	1	2	2	2	2	
307	1	2	2	1	2	2	
308	1	2	2	2	2	2	
...	
307	YES						

Figure 2.2 Import Dataset

2.2 Data pre-processing and Exploratory data analysis

2.2.1 Data Formatting

To ensure all the attributes are successfully imported, the attributes are displayed using `print(df.head())`.

```
[3] ✓ 0.0s
```

...	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	\
0	M	69	1	2	2	1	
1	M	74	2	1	1	1	
2	F	59	1	1	1	2	
3	M	63	2	2	2	1	
4	F	63	1	2	1	1	
	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COUGHING	\
0		1	2	1	2	2	2
1		2	2	2	1	1	1
2		1	2	1	2	1	2
3		1	1	1	1	2	1
4		1	1	1	2	1	2
	SHORTNESS OF BREATH	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER			
0		2	2	2	YES		
1		2	2	2	YES		
2		2	1	2	NO		
3		1	2	2	NO		
4		2	1	1	NO		

Figure 2.3 Attributes

Then, the total number of rows and columns in the dataset is identified using the `df.shape` method. There are 309 rows and 16 columns indicating that the dataset is successfully imported.

```
[4] ✓ 0.0s
```

```
... (309, 16)
```

Figure 2.4 Total number of rows and columns

The columns of the dataset are then displayed using `df.columns` method.

```
print(df.columns)

[5] ✓ 0.0s

... Index(['GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',
        'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',
        'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
        'SWALLOWING DIFFICULTY', 'CHEST PAIN', 'LUNG_CANCER'],
        dtype='object')
```

Figure 2.5 Columns of the dataset

The datatype for each attribute is identified using `df.dtypes` method.

```
#Identify data type
print(df.dtypes)

[6] ✓ 0.0s

... GENDER      object
    AGE      int64
    SMOKING    int64
    YELLOW_FINGERS int64
    ANXIETY    int64
    PEER_PRESSURE int64
    CHRONIC DISEASE int64
    FATIGUE    int64
    ALLERGY    int64
    WHEEZING   int64
    ALCOHOL CONSUMING int64
    COUGHING   int64
    SHORTNESS OF BREATH int64
    SWALLOWING DIFFICULTY int64
    CHEST PAIN int64
    LUNG_CANCER object
    dtype: object
```

Figure 2.6 Datatype for each attribute

2.2.2 Data Duplication Identifying and Handling

The duplicated data are identified using Python's built-in method, `df.duplicated()`. The total number of duplicated data is displayed using `df.duplicated().sum()` method. There are 33 duplicate values shown in Figure. The duplicate values are dropped and the updated data frame is printed. From Figure 10, there are 276 rows and 16 columns.

```
# IDENTIFY DUPLICATE VALUES
print('Total duplicate values: ', df.duplicated().sum())
data_duplicate = pd.DataFrame(df)
data = data_duplicate.drop_duplicates()
print(data)

✓ 0.0s
```

Figure 2.7 Data Duplication Identifying and Handling

```
Total duplicate values: 33
GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
0      M    69      1              2          2            1
1      M    74      2              1          1            1
2      F    59      1              1          1            2
3      M    63      2              2          2            1
4      F    63      1              2          1            1
..    ...    ...    ...            ...    ...            ...
279    F    59      1              2          2            2
280    F    59      2              1          1            1
281    M    55      2              1          1            1
282    M    46      1              2          2            1
283    M    60      1              2          2            1

CHRONIC  DISEASE  FATIGUE  ALLERGY  WHEEZING  ALCOHOL  CONSUMING  \
0          1      2        1        2          2            2
1          2      2        2        1          1            1
2          1      2        1        2          2            1
3          1      1        1        1          1            2
4          1      1        1        2          2            1
..    ...    ...    ...    ...    ...            ...
279          1      1        2        2          2            1
280          2      2        2        1          1            1
281          1      2        2        1          1            1
282          1      1        1        1          1            1
283          1      2        1        2          2            2

COUGHING  SHORTNESS OF BREATH  SWALLOWING DIFFICULTY  CHEST PAIN  \
0          2              2              2            2
1          1              2              2            2
2          2              2              1            2
3          1              1              2            2
4          2              2              1            1
..    ...    ...    ...    ...            ...
279          2              1              2            1
280          1              2              1            1
281          1              2              1            2
282          1              1              2            2
283          2              2              2            2

LUNG_CANCER
0      YES
1      YES
2      NO
3      NO
4      NO
..    ...
279    YES
280    NO
281    NO
282    NO
283    YES

[276 rows x 16 columns]
```

Figure 2.8 Columns of the dataset

2.2.3 Missing Values Identifying and Handling

The missing values are identified using the method, `.isnull().sum()`. The output shows that there are no null values in the dataset.

```
# Check for null value
print(data.isnull().sum())
```

[8] ✓ 0.0s

...	GENDER	0
	AGE	0
	SMOKING	0
	YELLOW_FINGERS	0
	ANXIETY	0
	PEER_PRESSURE	0
	CHRONIC_DISEASE	0
	FATIGUE	0
	ALLERGY	0
	WHEEZING	0
	ALCOHOL_CONSUMING	0
	COUGHING	0
	SHORTNESS_OF_BREATH	0
	SWALLOWING_DIFFICULTY	0
	CHEST_PAIN	0
	LUNG_CANCER	0
	dtype: int64	

Figure 2.9 Missing Values Identifying and Handling

2.2.4 Data Normalization

The ratio variable which is age is converted into a floating point for easier analysis. The Normalized Age is generated as a new column in the data frame. Among the normalization techniques, Min-Max scaling is implemented which is represented by the method `MinMaxScaler()`.

The formula for data normalization is shown below:

$$x' = (x - x_{min}) / (x_{max} - x_{min})$$

```
# Data normalization (centering/scaling)
from sklearn.preprocessing import MinMaxScaler

## Copy data
normalized_data = data.copy()
column = 'Normalized_Age'

# Applying normalization technique
normalized_data[column] = MinMaxScaler().fit_transform(
    np.array(data['AGE']).reshape(-1,1)
)
print(normalized_data['Normalized_Age'])
```

[9] ✓ 1.3s

```
... 0      0.727273
     1      0.803030
     2      0.575758
     3      0.636364
     4      0.636364
     ...
    279     0.575758
    280     0.575758
    281     0.515152
    282     0.378788
    283     0.590909
Name: Normalized_Age, Length: 276, dtype: float64
```

Figure 2.10 Data Normalization

The attributes are rearranged to ensure that the normalized age and age are next to each other.

```
[10] # Adjust the column arrangement
new_order = [0,1,16,2,3,4,5,6,7,8,9,10,11,12,13,14,15]
adjusted_data = pd.DataFrame(normalized_data[normalized_data.columns[new_order]])
print(adjusted_data)
✓ 0.0s
```

Figure 2.11 Column Rearrangement

	GENDER	AGE	Normalized_Age	SMOKING	YELLOW_FINGERS	ANXIETY	\
0	M	69	0.727273	1	2	2	
1	M	74	0.803030	2	1	1	
2	F	59	0.575758	1	1	1	
3	M	63	0.636364	2	2	2	
4	F	63	0.636364	1	2	1	
..	
279	F	59	0.575758	1	2	2	
280	F	59	0.575758	2	1	1	
281	M	55	0.515152	2	1	1	
282	M	46	0.378788	1	2	2	
283	M	60	0.590909	1	2	2	

	PEER_PRESSURE	CHRONIC_DISEASE	FATIGUE	ALLERGY	WHEEZING	\
0	1	1	2	1	2	
1	1	2	2	2	1	
2	2	1	2	1	2	
3	1	1	1	1	1	
4	1	1	1	1	2	
..	
279	2	1	1	2	2	
280	1	2	2	2	1	
281	1	1	2	2	1	
282	1	1	1	1	1	
283	1	1	2	1	2	

	ALCOHOL_CONSUMING	COUGHING	SHORTNESS_OF_BREATH	SWALLOWING_DIFFICULTY	\
0	2	2	2	2	
1	1	1	2	2	
2	1	2	2	1	
3	2	1	1	2	
4	1	2	2	1	
..	
279	1	2	1	2	
280	1	1	2	1	
281	1	1	2	1	
282	1	1	1	2	
283	2	2	2	2	

	CHEST_PAIN	LUNG_CANCER
0	2	YES
1	2	YES
2	2	NO
3	2	NO
4	1	NO
..
279	1	YES
280	1	NO
281	2	NO
282	2	NO
283	2	YES

[276 rows x 17 columns]

Figure 2.12 Columns of the dataset

2.2.5 Data Binning

The original data values are divided into small intervals known as bins and then they are replaced by a general value calculated for that bin. This has a smoothing effect on the input data and may also reduce the chances of overfitting in the case of small datasets. For our data frame, a total of 4 bins were created, grouped into 3 age groups.

```
## Data Binning for AGE variable

bins = [20, 40, 60, 100]
group_names = ["[20-40] years", "[41-60] years", "[60+] years"]
adjusted_data["AGE GROUPS"] = pd.cut(
    adjusted_data["AGE"], bins, labels=group_names)

age_group = adjusted_data['AGE GROUPS']
preprocessed_data = adjusted_data.drop(columns = 'AGE GROUPS')
preprocessed_data.insert(loc=2, column='AGE GROUPS', value=age_group)
print(preprocessed_data)
```

[11] ✓ 0.0s

Figure 2.13 Data Binning

2.2.6 Indicator Variable

The indicator variable is set for 3 attributes namely, Gender, Age Groups, and Lung Cancer. The columns are then rearranged and displayed.

```
## INDICATOR VARIABLES (for convert categorical data to indicator variables)
df_dc = pd.get_dummies(
    preProcessed_data,
    columns = ['GENDER', 'AGE GROUPS', 'LUNG_CANCER'], dtype=int)

# Adjust the column arrangement
new_order1 = [15,16,0,1,17,18,19,2,3,4,5,6,7,8,9,10,11,12,13,14,20,21]
preProcessed_data1 = pd.DataFrame(df_dc[df_dc.columns[new_order1]])
print(preProcessed_data1)
```

✓ 0.0s Python

Figure 2.14 Indicator Variable

	GENDER_F	GENDER_M	AGE	Normalized_Age	AGE_GROUPS_[20-40] years	\
0	0	1	69	0.727273		0
1	0	1	74	0.803030		0
2	1	0	59	0.575758		0
3	0	1	63	0.636364		0
4	1	0	63	0.636364		0
..
279	1	0	59	0.575758		0
280	1	0	59	0.575758		0
281	0	1	55	0.515152		0
282	0	1	46	0.378788		0
283	0	1	60	0.590909		0

	AGE_GROUPS_[41-60] years	AGE_GROUPS_[60+] years	SMOKING	\
0	0	1	1	
1	0	1	2	
2	1	0	1	
3	0	1	2	
4	0	1	1	
..	
279	1	0	1	
280	1	0	2	
281	1	0	2	
282	1	0	1	
283	1	0	1	

	YELLOW_FINGERS	ANXIETY	...	FATIGUE	ALLERGY	WHEEZING	\
0	2	2	...	2	1	2	
1	1	1	...	2	2	1	
2	1	1	...	2	1	2	
3	2	2	...	1	1	1	
4	2	1	...	1	1	2	
..	
279	2	2	...	1	2	2	
280	1	1	...	2	2	1	
281	1	1	...	2	2	1	
282	2	2	...	1	1	1	
283	2	2	...	2	1	2	

	ALCOHOL-CONSUMING	COUGHING	SHORTNESS OF BREATH	SWALLOWING DIFFICULTY	\
0	2	2		2	2
1	1	1		2	2
2	1	2		2	1
3	2	1		1	2
4	1	2		2	1
..
279	1	2		1	2
280	1	1		2	1
281	1	1		2	1
282	1	1		1	2
283	2	2		2	2

	CHEST PAIN	LUNG_CANCER_NO	LUNG_CANCER_YES
0	2	0	1
1	2	0	1
2	2	1	0
3	2	1	0
4	1	1	0
..
279	1	0	1
280	1	1	0
281	2	1	0
282	2	1	0
283	2	0	1

[276 rows x 22 columns]

Figure 2.15 Columns of the dataset

2.2.7 Export preprocessed data

The preprocessed dataset ID output as a CSV file.

```
## Export data frame to csv file
out_csv = "preprocessed_lungcancerdata.csv"
preprocessed_data1.to_csv(out_csv)
✓ 0.0s
```

Figure 2.16 Export preprocessed data

Exploratory Data Analysis

To understand the dataset, Exploratory Data Analysis(EDA) is the most important first step in data analysis. It is the approach for us to understand the general patterns of our dataset. Thus, to summarize our dataset, we have included the following descriptive analysis and all the correlations.

2.2.8 Descriptive Analysis

Firstly, we import our preprocessed datasets in the format of CSV file into Python. The dataset is shown as the output of the code, which is shown in the diagram below. At the same time, we drop an unrelated column in the dataset.

```
# Import csv file into python
df = pd.read_csv('preprocessed_lungcancerdata.csv')

# drop unrelated column
df = df.drop(df.columns[0],axis=1)
print(df)
```

✓ 0.0s

	GENDER_F	GENDER_M	AGE	Normalized_Age	AGE GROUPS_[20-40] years	\
0	0	1	69	0.727273		0
1	0	1	74	0.803030		0
2	1	0	59	0.575758		0
3	0	1	63	0.636364		0
4	1	0	63	0.636364		0
..
271	1	0	59	0.575758		0
272	1	0	59	0.575758		0
273	0	1	55	0.515152		0
274	0	1	46	0.378788		0
275	0	1	60	0.590909		0

	AGE GROUPS_[41-60] years	AGE GROUPS_[60+] years	SMOKING	\
0		0	1	1
1		0	1	2
2		1	0	1
3		0	1	2
4		0	1	1
..
271		1	0	1
272		1	0	2
273		1	0	2
274		1	0	1
275		1	0	1
...				
274	2	1	0	
275	2	0	1	

[276 rows x 22 columns]

Figure 2.17 Import preprocessed datasets

2.2.9 Mean, Variance, and Standard Deviation of AGE

We will next proceed to analyze the age based on the age groups. To do this, we will calculate the mean, variance, and standard deviation of age and each age group. By using the `.mean()`, we get the output as shown below. It can be seen that the mean for Age is 62.909 which is nearly 63. Besides that, by comparing the means of each age group, we can conclude that the age group of 60+ has the highest mean which is 0.609.

```
# Mean for the age and age groups columns
age = df[['AGE', 'AGE GROUPS_[20-40] years', 'AGE GROUPS_[41-60] years', 'AGE GROUPS_[60+] years']].mean()
print(age)
```

AGE	62.909420
AGE GROUPS_[20-40] years	0.010870
AGE GROUPS_[41-60] years	0.380435
AGE GROUPS_[60+] years	0.608696

dtype: float64

Figure 2.18 Mean age

Next, we calculate the variance for the age and age groups column. The variance of the age column is 70.213584 which means that much data is far from the mean and every other data from the dataset. This is because the higher the variance, the more the data spread out from the mean and among themselves. Meanwhile, the variance for the age groups of 20-40, 41-60, and 60+ are 0.010791, 0.236561, and 0.239051.

```
# Variance for the age and age groups columns
age = df[['AGE', 'AGE GROUPS_[20-40] years', 'AGE GROUPS_[41-60] years', 'AGE GROUPS_[60+] years']].var()
print(age)
```

AGE	70.213584
AGE GROUPS_[20-40] years	0.010791
AGE GROUPS_[41-60] years	0.236561
AGE GROUPS_[60+] years	0.239051

dtype: float64

Figure 2.19 Variance of Age

The standard deviation of the age column is 8.379355. Meanwhile, the standard deviation for the age groups of 20-40, 41-60, and 60+ are 0.103877, 0.486376, and 0.488929.


```
# Std deviation for the age and age groups columns
age = df[['AGE', 'AGE GROUPS_[20-40] years', 'AGE GROUPS_[41-60] years', 'AGE GROUPS_[60+] years']].std()
print(age)
```

AGE	8.379355
AGE GROUPS_[20-40] years	0.103877
AGE GROUPS_[41-60] years	0.486376
AGE GROUPS_[60+] years	0.488929

dtype: float64

Figure 2.20 Standard deviation of Age

2.2.10 Mean, Variance, and Standard Deviation of GENDER

Next, we will be calculating the mean, variance, and standard deviation of gender which are the values for males and females in the dataset. As usual, calculate the mean for genders first. GENDER_M is 0.514493 which is larger than GENDER_F 0.485507. Thus, it can be concluded that males are more than females in this dataset.

```
# Mean for the gender columns (female and male)
gender = df[['GENDER_F', 'GENDER_M']].mean()
print(gender)
```

GENDER_F	0.485507
GENDER_M	0.514493

dtype: float64

Figure 2.21 Mean of Gender

For gender, males and females have the same value for each variance and standard deviation. The variance is 0.250698 for both genders while the standard deviation is 0.500698 for both genders.

```
# variance for the gender columns (female and male)
gender = df[['GENDER_F', 'GENDER_M']].var()
print(gender)
```

```
GENDER_F    0.250698
GENDER_M    0.250698
dtype: float64
```

Figure 2.22 Variance of Gender

```
# Std deviation for the gender columns (female and male)
gender = df[['GENDER_F', 'GENDER_M']].std()
print(gender)
```

```
GENDER_F    0.500698
GENDER_M    0.500698
dtype: float64
```

Figure 2.23 Standard Deviation of Gender

2.2.11 Mean, Variance, and Standard Deviation of LUNG CANCER

Then, we will do the lung cancer result, which indicates whether the dataset suffers from lung cancer or not. It is indicated with Yes or No. To compare the results of lung cancer in the dataset, we first calculated the mean. From the result, it can be seen that the mean of Yes is more than No. Thus, there are more datasets that suffer from lung cancer.

```
# Mean for the lung cancer columns (yes or no)
lung_cancer = df[['LUNG_CANCER_NO', 'LUNG_CANCER_YES']].mean()
print(lung_cancer)
```

```
LUNG_CANCER_NO    0.137681
LUNG_CANCER_YES    0.862319
dtype: float64
```

Figure 2.24 Mean for Lung Cancer

For lung cancer results, males and females have the same value for each variance and standard deviation. The variance is 0.119157 for both results while the standard deviation is 0.345191 for both Yes and No results.

```
# Variance for the lung cancer columns (yes or no)
lung_cancer = df[['LUNG_CANCER_NO', 'LUNG_CANCER_YES']].var()
print(lung_cancer)
```

LUNG_CANCER_NO	0.119157
LUNG_CANCER_YES	0.119157
dtype: float64	

Figure 2.25 Variance of Lung Cancer

```
# Std variance for the lung cancer columns (yes or no)
lung_cancer = df[['LUNG_CANCER_NO', 'LUNG_CANCER_YES']].std()
print(lung_cancer)
```

LUNG_CANCER_NO	0.345191
LUNG_CANCER_YES	0.345191
dtype: float64	

Figure 2.26 Standard deviation of Lung Cancer

2.2.12 Mode

To know which columns are the most frequent in the dataset, we investigate the mode for Age, Age group 20-40, 40-60, 60+, female, male, and the lung result of No and Yes. The result of the mode is shown as follows.

```
# Mode of the columns
df[['AGE', 'AGE_GROUPS_[20-40] years', 'AGE_GROUPS_[41-60] years', 'AGE_GROUPS_[60+] years', 'GENDER_F', 'GENDER_M', 'LUNG_CANCER_NO', 'LUNG_CANCER_YES']].mode()
```

	AGE	AGE_GROUPS_[20-40] years	AGE_GROUPS_[41-60] years	AGE_GROUPS_[60+] years	GENDER_F	GENDER_M	LUNG_CANCER_NO	LUNG_CANCER_YES
0	64	False	False	True	False	True	False	True

Figure 2.27 Mode

2.2.13 Describe data(columns)

Our next step to have a better understanding of our data, will be using `data.describe()`. The data is summarized in the table below. The table includes the total number of datasets, the mean value, the standard deviation value, minimum and maximum for each column. More than that, the percentile values are also shown as the rows of 25%, 50%, and 75%.

```
# Describe the columns
df[['AGE', 'AGE_GROUPS_[20-40] years', 'AGE_GROUPS_[41-60] years', 'AGE_GROUPS_[60+] years', 'GENDER_F', 'GENDER_M',
    'LUNG_CANCER_NO', 'LUNG_CANCER_YES']].describe(include="all")
```

	AGE	AGE_GROUPS_[20-40] years	AGE_GROUPS_[41-60] years	AGE_GROUPS_[60+] years	GENDER_F	GENDER_M	LUNG_CANCER_NO	LUNG_CANCER_YES
count	276.000000	276.000000	276.000000	276.000000	276.000000	276.000000	276.000000	276.000000
mean	62.909420	0.010870	0.380435	0.608696	0.485507	0.514493	0.137681	0.862319
std	8.379355	0.103877	0.486376	0.488929	0.500698	0.500698	0.345191	0.345191
min	21.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	57.750000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000
50%	62.500000	0.000000	0.000000	1.000000	0.000000	1.000000	0.000000	1.000000
75%	69.000000	0.000000	1.000000	1.000000	1.000000	1.000000	0.000000	1.000000
max	87.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000	1.000000

Figure 2.28 Columns of the dataset

2.2.14 Boxplot

To improve our analysis in determining lung cancer prediction, we then plotted a box plot for the age in the dataset. Firstly, import the required matplotlib.pyplot package to plot the boxplot.

```
import matplotlib.pyplot as plt
plt.boxplot(df['AGE'])
plt.title("Boxplot of Age")
```

Figure 2.29 Construction of boxplot

The boxplot of age is shown below. The yellow line in the boxplot is the median of the ages which is 62.5. We can also depict the interquartile range as we know the 25th percentile is at 57.75 and the 75th percentile is at 69, thus the interquartile range is 11.25. It can also be seen that there are 4 outliers.

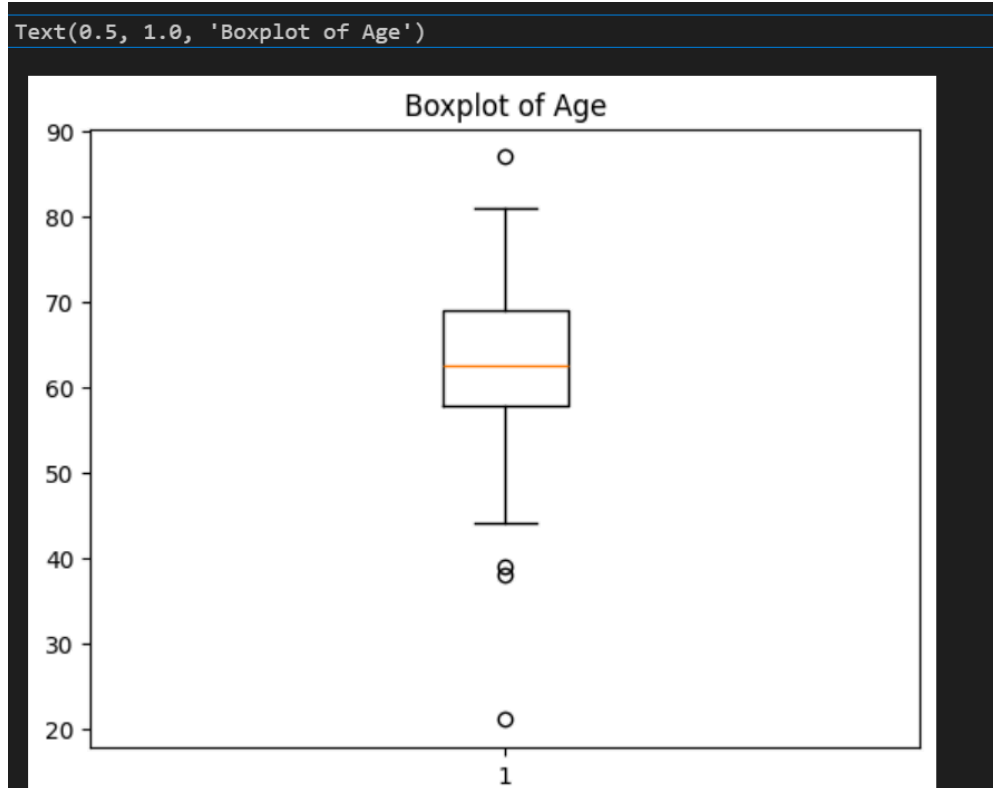


Figure 2.30 Boxplot of Age

2.2.15 Correlation

Last but not least, we do correlation as the final step of exploratory data analysis. Correlation is performed to measure the statistical relationship between two variables or quantities. To perform correlation, we will be using the `corr()` function. This `corr()` function is Pearson Correlation. The output of the function is shown as follow.

```
# DATA.CORRELATION
data=pd.read_csv('preprocessed_lungcancerdata.csv')

# DROP UNRELATED COLUMN
data = data.drop(data.columns[0], axis = 1)
corr_data = data.corr()
print(corr_data)
```

Figure 2.31 Construction of Correlation

	GENDER_F	GENDER_M	...	LUNG_CANCER_NO	LUNG_CANCER_YES
GENDER_F	1.000000	-1.000000	...	0.053666	-0.053666
GENDER_M	-1.000000	1.000000	...	-0.053666	0.053666
AGE	0.013120	-0.013120	...	-0.106305	0.106305
Normalized_Age	0.013120	-0.013120	...	-0.106305	0.106305
AGE_GROUPS_[20-40] years	0.037997	-0.037997	...	0.059524	-0.059524
AGE_GROUPS_[41-60] years	0.000325	-0.000325	...	0.076748	-0.076748
AGE_GROUPS_[60+] years	-0.008396	0.008396	...	-0.088993	0.088993
SMOKING	-0.041131	0.041131	...	-0.034878	0.034878
YELLOW_FINGERS	0.202506	-0.202506	...	-0.189192	0.189192
ANXIETY	0.152032	-0.152032	...	-0.144322	0.144322
PEER_PRESSURE	0.261427	-0.261427	...	-0.195086	0.195086
CHRONIC_DISEASE	0.189925	-0.189925	...	-0.143692	0.143692
FATIGUE	0.079020	-0.079020	...	-0.160078	0.160078
ALLERGY	-0.150174	0.150174	...	-0.333552	0.333552
WHEEZING	-0.121047	0.121047	...	-0.249054	0.249054
ALCOHOL_CONSUMING	-0.434264	0.434264	...	-0.294422	0.294422
COUGHING	-0.120228	0.120228	...	-0.253027	0.253027
SHORTNESS_OF_BREATH	0.052893	-0.052893	...	-0.064407	0.064407
SWALLOWING_DIFFICULTY	0.048959	-0.048959	...	-0.268940	0.268940
CHEST_PAIN	-0.361547	0.361547	...	-0.194856	0.194856
LUNG_CANCER_NO	0.053666	-0.053666	...	1.000000	-1.000000
LUNG_CANCER_YES	-0.053666	0.053666	...	-1.000000	1.000000

[22 rows x 22 columns]

Figure 2.32 Result of Correlation

We then went ahead and created a heatmap to show the predictors' correlation coefficient. The values on the graph are represented by a heatmap with various hues. The lighter hues correspond to higher values, while the darker hues correspond to lower values. The values stand for the correlation between each column and the others. The data that are highly correlated with one another are represented by the high value. Therefore, the correlation between the columns is larger the higher the value. We import the necessary seaborn and matplotlib packages to accomplish this. The figure below displays the heatmap.

```
import matplotlib.pyplot as mp
import seaborn as sns

mp.figure(figsize=(15,15))
sns.heatmap(data.corr(),annot=True,linewidth=0.5,fmt='0.2f')
mp.title('Correlation Coefficient of Lung Cancer Predictors')
mp.show()
```

Figure 2.33 Construction of Heatmap

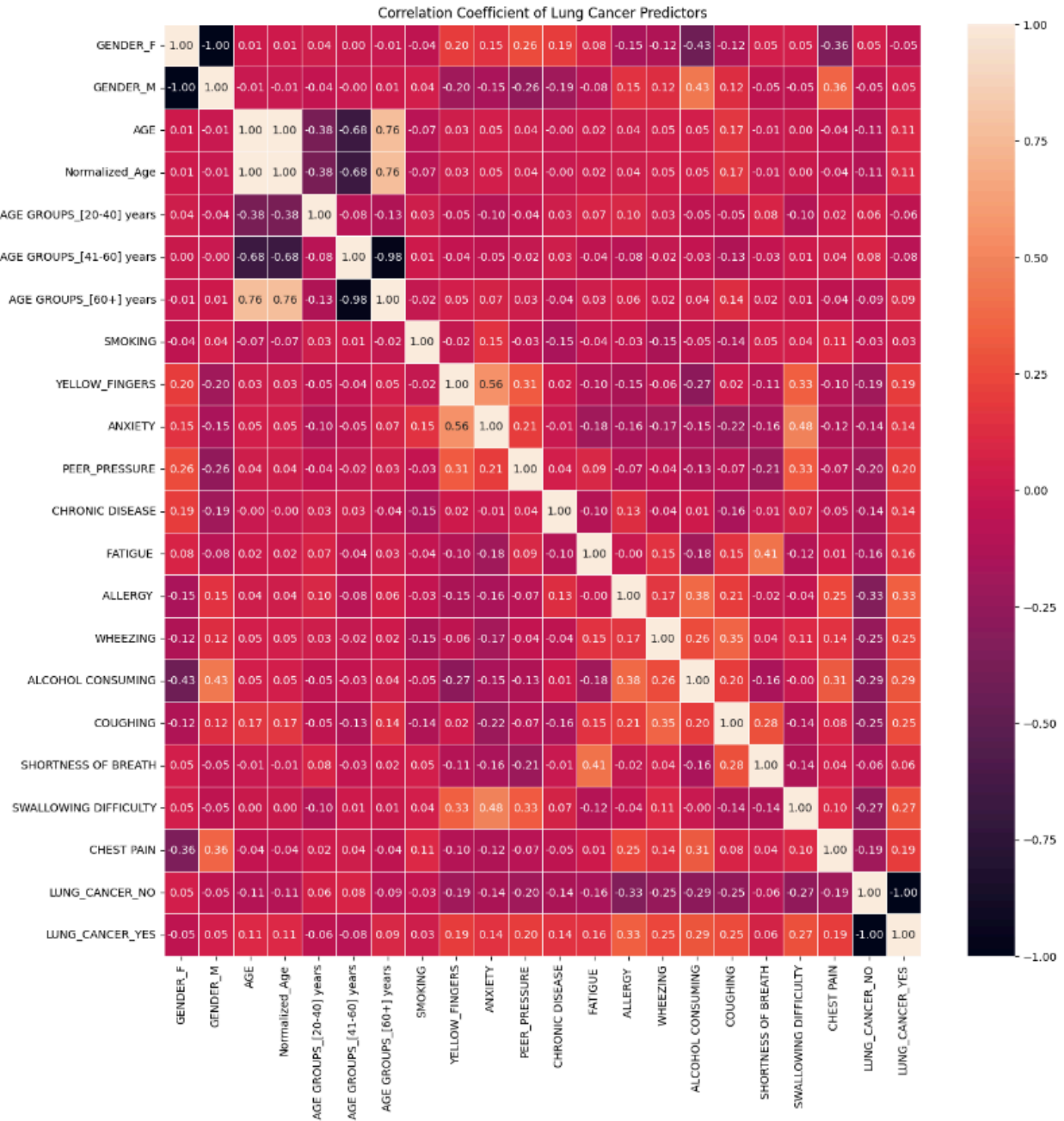


Figure 2.34 Heatmap

2.3 Software and hardware requirements

1. Software requirements

- Windows 11
- GitHub
- Visual Studio Code

2. Hardware requirements

Component	Specification
Processor	Intel Core i7 or AMD RYZEN 7 5000X
RAM	16GB DDR5-6400
Solid State Drive or Hard Disk Drive	512GB
Cache Memory	16MB
GPU	8GB

3.0 Flowchart of proposed approach

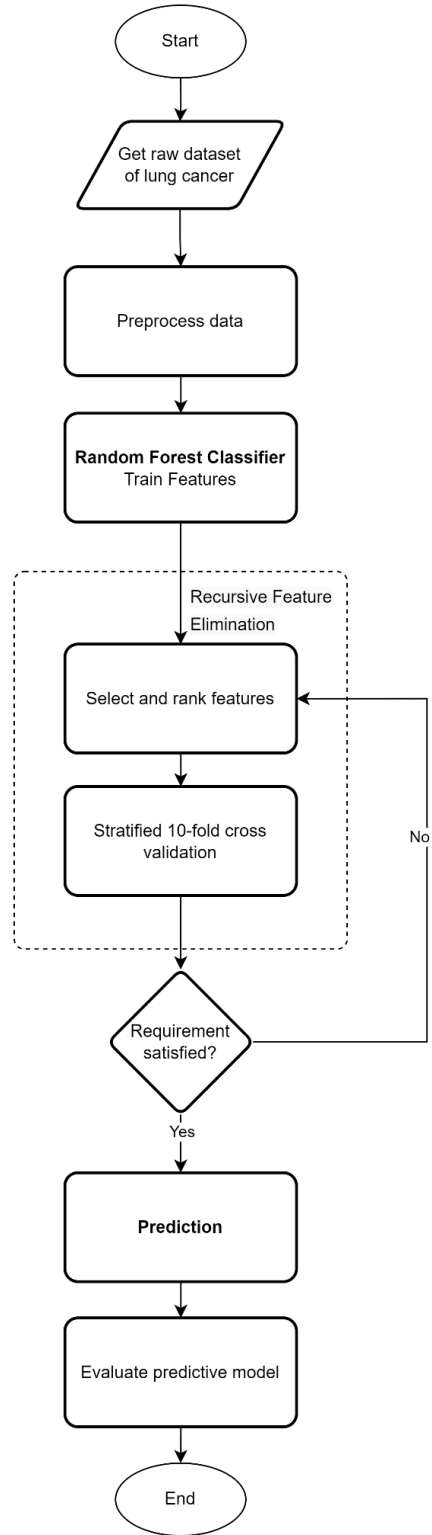


Figure 3.1 Flowchart

3.1 Model Development

Next, we will be moving to model development. This model development is to help us to have a better understanding of the relationship between the different variables to estimate the outcome. We would like to predict future observations from our imported data through the developed model. Firstly, we need to import all the required packages.

```
from sklearn.feature_selection import RFE
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.model_selection import StratifiedKFold
from sklearn.model_selection import cross_val_score
from sklearn import metrics
import seaborn as sn      "sn" is not accessed
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np      "np" is not accessed
```

✓ 0.9s

Figure 3.2 Import Packages

To make the data or model easier to identify, we first need to modify the preprocessed data. The function `.replace` is used to modify the data. Firstly, we replace the categorical variables of males and females into 1 and 0 respectively which are binary forms so that we can easily identify them.

```
preProcessed_data['GENDER'].replace(['M', 'F'],[1,0],inplace=True)
print(preProcessed_data.head(5))
```

✓ 0.0s

	GENDER	AGE	AGE GROUPS	...	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER
0	1	69	[60+] years	...	2	2	YES
1	1	74	[60+] years	...	2	2	YES
2	0	59	[41-60] years	...	1	2	NO
3	1	63	[60+] years	...	2	2	NO
4	0	63	[60+] years	...	1	1	NO

[5 rows x 18 columns]

Figure 3.3 Replacement of categorical variables(Gender)

Then, we modify the age and lung cancer data. The age ranges are divided into three categories: 0 (20–40 years), 1 (41–60 years), and 2 (60+ years). In the meanwhile, binary 0 or 1 is used in place of the yes and no in the lung cancer column. It is shown as follows.

```
# AGE / AGE GROUPS / NORMALIZED_AGE are describing same thing using diff ways, so may include 1 (AGE GROUPS) is enough?
preProcessed_data['AGE_GROUPS'].replace(["[20-40] years", "[41-60] years", "[60+] years"],[0,1,2], inplace = True)
print(preProcessed_data.head(5))
preProcessed_data['LUNG_CANCER'].replace(['YES','NO'],[1,0], inplace = True)
print(preProcessed_data.head(5))
```

✓ 0.0s

	GENDER	AGE	AGE_GROUPS	...	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER
0	1	69	2	...	2	2	YES
1	1	74	2	...	2	2	YES
2	0	59	1	...	1	2	NO
3	1	63	2	...	2	2	NO
4	0	63	2	...	1	1	NO

[5 rows x 18 columns]

	GENDER	AGE	AGE_GROUPS	...	SWALLOWING DIFFICULTY	CHEST PAIN	LUNG_CANCER
0	1	69	2	...	2	2	1
1	1	74	2	...	2	2	1
2	0	59	1	...	1	2	0
3	1	63	2	...	2	2	0
4	0	63	2	...	1	1	0

[5 rows x 18 columns]

Figure 3.4 Replacement of categorical variables(Age group, Lung cancer)

The updated dataset is exported into modeldev_lungcancerdata.csv.

```
out_csv = "modeldev_lungcancerdata.csv"
preProcessed_data.to_csv(out_csv)
```

✓ 0.0s

Figure 3.5 Export Preprocessed Data

Next, we imported the updated dataset. From the imported dataset, we identify the features that are not required for future analysis. These features include AGE, NORMALIZED_AGE, and LUNG_CANCER. These columns are then dropped by using the function `.drop()`. The remaining columns are shown in the output as follows.

```
lung_data = pd.read_csv('modeldev_lungcancerdata.csv')
X = lung_data.copy()

# drop SAMPLE column
X_2=X.drop(X.columns[0],axis=1)
X_3=X_2.drop(X_2.columns[1],axis=1)
X_4=X_3.drop(X_3.columns[2],axis=1)
X_5=X_4.drop(X_4.columns[15],axis=1)
print(X_5.columns)
y = lung_data.pop('LUNG_CANCER')
```

✓ 0.0s

```
Index(['GENDER', 'AGE GROUPS', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',
      'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',
      'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',
      'SWALLOWING DIFFICULTY', 'CHEST PAIN'],
      dtype='object')
```

Figure 3.6 Dropping of Unwanted Columns

Then, we train the model. The `train_test_split` function is used to split the dataset into a train set (70%) and a test set (30%). The number of data for each set is shown in the output below.

```
# TRAIN MODEL
X_train, X_test, y_train, y_test = train_test_split(X_5,y,test_size = 0.3, shuffle=False)
X_train.shape, X_test.shape
```

✓ 0.0s

```
((193, 15), (83, 15))
```

Figure 3.7 Model training

We set `step=1` and `n_features_to_select=10` and utilized `RandomForestClassifier` as the estimator to carry out the recursive feature elimination approach. `Step=1` indicated that one feature would be eliminated at each iteration, while `n_features_to_select=10` indicated that the selection process would end once ten features had been chosen. Next, we used a training set to train the

model. The features that were chosen, together with the sort of data they included, were displayed below.

```
# perform RFE via RandomForestClassifier
rf = RandomForestClassifier(random_state=0)
selector = RFE(estimator=rf, step=1, n_features_to_select=10)
selector = selector.fit(X_train, y_train)
#print('Feature selected Column: X_5.columns[selector.support_]')
#print("Feature ranking: ", selector.ranking_)

feature = X_train.columns[selector.get_support()]
print(feature)

✓ 0.7s

Index(['AGE_GROUPS', 'YELLOW_FINGERS', 'ANXIETY', 'PEER_PRESSURE',
      'CHRONIC_DISEASE', 'FATIGUE ', 'ALLERGY ', 'ALCOHOL_CONSUMING',
      'SHORTNESS OF BREATH', 'CHEST PAIN'],
      dtype='object')
```

Figure 3.8 RFE implementation

We've rearranged them into a data frame for a presentable look and easy comprehension of the chosen features. It printed True if the features were selected and False otherwise.

```
# Get list of features selected
features_kept = pd.DataFrame({'columns' : X_5.columns, 'Kept':selector.support_})
features_kept

✓ 0.0s
```

Figure 3.9 Getting List of Features Selected

	columns	Kept
0	GENDER	False
1	AGE GROUPS	True
2	SMOKING	False
3	YELLOW_FINGERS	True
4	ANXIETY	True
5	PEER_PRESSURE	True
6	CHRONIC DISEASE	True
7	FATIGUE	True
8	ALLERGY	True
9	WHEEZING	False
10	ALCOHOL CONSUMING	True
11	COUGHING	False
12	SHORTNESS OF BREATH	True
13	SWALLOWING DIFFICULTY	False
14	CHEST PAIN	True

Figure 3.10 List of Features Selected

The features rating is shown below. It makes use of the selector.ranking function. The symptoms, the estimated best-selected attributes, would be ranked 1st.

```
# Get list of features ranking
rank = pd.DataFrame({'columns' : X_5.columns, 'Kept': selector.ranking_})
rank
✓ 0.0s
```

Figure 3.11 Getting Rank of Selected Features

	columns	Kept
0	GENDER	3
1	AGE GROUPS	1
2	SMOKING	5
3	YELLOW_FINGERS	1
4	ANXIETY	1
5	PEER_PRESSURE	1
6	CHRONIC DISEASE	1
7	FATIGUE	1
8	ALLERGY	1
9	WHEEZING	2
10	ALCOHOL CONSUMING	1
11	COUGHING	6
12	SHORTNESS OF BREATH	1
13	SWALLOWING DIFFICULTY	4
14	CHEST PAIN	1

Figure 3.12 Rank of Selected Features

3.2 Model Evaluation

Firstly, we will be conducting Stratified 10-fold cross-validation by using StratifiedKFold. In the StratifiedKFold, we set the `n_splits` to 10. This is to divide our datasets 10 times. We do not want the order of features to be shuffled, thus we set the `shuffle` to false. At the same time, we do not want the order of features to be changed, thus we set the `random_state` to none.

```
# Conduct Stratified 10-fold cross validation
skf = StratifiedKFold(n_splits=10, shuffle=False, random_state=None)
✓ 0.0s
```

Figure 3.13 StratifiedKFold Implementation

Cross_val_score tests and trains the model on several dataset folds. Therefore, cross-validation can be used to determine the model performance of the dataset. Additionally, it is employed to encourage model generalization and avoid overfitting. The data to fit the model is denoted by the parameter rf, and the model's target is X_5. The score parameter determines the metric to be used, whereas the number of splits to be used (cv), where the stratifiedKFold is displayed above, is set to skf.

```
# Cross val score is used to prevent over-fitting and promote model generalisation
n_scores = cross_val_score(rf, X_5, y, scoring = 'accuracy', cv=skf)
print('Mean score is %0.2f' % (n_scores.mean()))
✓ 1.0s
Mean score is 0.89
```

Figure 3.14 Cross-validation

Then, we calculate the accuracy score. This score is used to calculate the accuracy of predicting lung cancer symptoms. The accuracy score is 0.939759 which is near to 0.94.

```
# Predict the accuracy of lung cancer's symptoms
y_pred = selector.predict(X_test)
print("Accuracy: ", metrics.accuracy_score(y_test, y_pred))
✓ 0.0s
Accuracy: 0.9397590361445783
```

Figure 3.15 Accuracy

Next, the precision score and recall score are also calculated.

```
# Predict the precision of lung cancer's symptoms
y_pred = selector.predict(X_test)
print("Precision: ", metrics.precision_score(y_test, y_pred))
✓ 0.0s
Precision: 0.9577464788732394
```

Figure 3.16 Precision


```
#Predict the recall of Lung cancer's symptoms
y_pred = selector.predict(X_test)
print("Recall: ", metrics.recall_score(y_test, y_pred))
✓ 0.0s
Recall:  0.9714285714285714
```

Figure 3.17 Recall

4.0 Results, Testing, and Validation

In this project, the reason that a train set (70%) and a test set (30%) are used to train the model is that they contribute to the highest accuracy of our predictive model. Therefore, it is proven in the table below:

Train set	Test set	Accuracy
0.10	0.90	0.89
0.20	0.80	0.84
0.30	0.70	0.89
0.40	0.60	0.90
0.50	0.50	0.91
0.60	0.40	0.93
0.70	0.30	0.94
0.80	0.20	0.91
0.90	0.10	0.89

Table 1 Training and Testing Set with respective accuracy

In this project, a confusion matrix is applied to evaluate the predictive model. It can help to show how many predictions are correct and how many are incorrect.

```
# Evaluate predictive model using Confusion Matrix 2 data_pred pd.DataFrame({'y_pred':y_pred})
data_pred = pd.DataFrame({'y_pred':y_pred})
data_actual = pd.DataFrame({'y_actual':y})
data_merged = pd.merge(data_actual, data_pred, left_index=True, right_index=True)
print(data_merged, '\n\n')

confusion_matrix = pd.crosstab(data_merged['y_actual'], data_merged['y_pred'], rownames=['Actual'], colnames=['Predicted'])
print(confusion_matrix)

sns.heatmap(confusion_matrix, annot=True)
plt.show()
```

Figure 4.1 Confusion Matrix Implementation

The result is shown in the figure below. From the figure, we can see that there is 1 in the True Negative region, 16 in the False Positive region, 11 in the False Negative region, and 55 in the True Positive region.

```

      y_actual y_pred
0         1      1
1         1      1
2         0      1
3         0      1
4         0      1
..      ...    ...
78        1      1
79        1      1
80        0      1
81        1      0
82        1      1

[83 rows x 2 columns]

Predicted  0   1
Actual
0          1  16
1         11  55

```

Figure 4.2 Region of Confusion Matrix

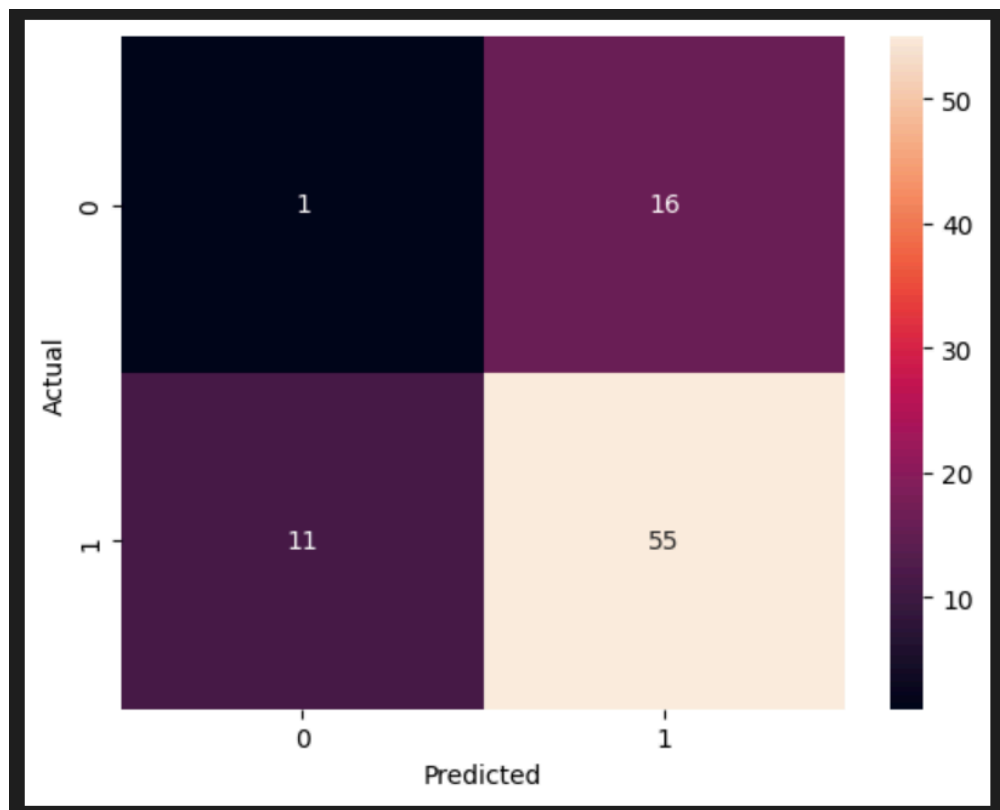


Figure 4.3 Region of Confusion Matrix

Then, we calculate the confusion matrix's accuracy. It is shown that the confusion matrix's accuracy is 0.6746987951807228.

```
# (True positive + True Negative) / Total predictions
print("Confusion Matrix's Accuracy: ", (55+1)/83)
✓ 0.0s

Confusion Matrix's Accuracy:  0.6746987951807228
```

Figure 4.4 Confusion matrix's Accuracy

At the same time, the precision and recall obtained are 0.7746478873239436 and 0.8333333333333334 respectively.

```
# True positive / (True Negative + False Positive)
print("Confusion Matrix's Precision: ", 55/(55+16))
✓ 0.0s

Confusion Matrix's Precision:  0.7746478873239436
```

Figure 4.5 Confusion matrix's Precision

```
# True positive / (True Positive + False Negative)
print("Confusion Matrix's Recall: ", 55/(55+11))
✓ 0.0s

Confusion Matrix's Recall:  0.8333333333333334
```

Figure 4.6 Confusion matrix's Recall

4.1 Feedback

Feedback from Dr. Sharin Hazlin Binti Huspi

FEEDBACK:

Overall, the group has done a good job in learning and applying what they need to do for the project. It was easy to guide them as they were able to understand what they needed to do. They were also interested in the area, thus they were able to understand the process and develop the algorithm.

However, they still need to learn to discuss their findings. I think there are still areas to learn in doing the discussion analysis of the results.

CLIENT NAME	Sharin Hazlin Huspi		
SIGNATURE	Shmie	DATE	11/02/2024

5.0 Conclusion

In conclusion, our model achieved a high accuracy score which 0.94. This shows that the model has good performance. Thus, we can conclude that we have achieved our objectives for this project. Our project used the method of recursive feature elimination to predict lung cancer. Other than that, we also used a confusion matrix to evaluate our model. In the end, we can conclude the 10 important features that are the symptoms related to lung cancer. The 10 important features include age groups, yellow fingers, anxiety, peer pressure, chronic disease, fatigue, allergy, alcohol consuming, shortness of breath, and chest pain. By knowing these 10 important features, we can increase the possibility of predicting lung cancer. This achieved our objectives which were to have a method that can enhance the accuracy and effectiveness of lung cancer prediction models. At the same time, it can increase the chances of realizing the presence of lung cancer and increase the chances of successful treatment. Survival might be significantly impacted by an early diagnosis. This is because it might guarantee a treatment that is both effective and less likely to spread (Georgia, 2022). Additionally, research indicates that individuals who receive an early diagnosis of lung cancer have a 20-year survival probability of up to 80% (Mount Sinai Health System, 2022).

6.0 References

1. American Cancer Society. What Is Lung Cancer? Retrieved from <https://www.cancer.org/cancer/types/lung-cancer/about/what-is.html>
2. Georgia I. Salvaryn. (2022, April 30). Early lung cancer detection improves survival rates. Healio. Retrieved January 17, 2023, from <https://www.healio.com/news/hematology-oncology/20220421/early-lung-cancer-detection-improves-survival-rates>
3. Mount Sinai Health System. (2022, November 22). Lung cancer screening dramatically increases long-term survival rate. Mount Sinai Health System. Retrieved January 17, 2023, from <https://www.mountsinai.org/about/newsroom/2022/lung-cancer-screening-dramatically-increases-long-term-survival-rate#:~:text=The%20results%20show%20that%20patients,diagnosed%20at%20an%20early%20stage>

4. Mayo Clinic (2023). Lung cancer - Symptoms and causes. Retrieved from <https://www.mayoclinic.org/diseases-conditions/lung-cancer/symptoms-causes/syc-20374620#:~:text=Lung%20cancer%20is%20the%20leading,of%20cigarettes%20you%27ve%20smoked>
5. M. A.Bhat (2021). Lung Cancer. Retrieved from [Lung Cancer \(kaggle.com\)](#)
6. R. Vethasalam, J. Ibrahim (2023, November 5). 95% of lung cancer cases detected late (Poll Inside) Retrieved from <https://www.thestar.com.my/news/nation/2023/11/05/95-of-lung-cancer-cases-detected-late#:~:text=The%20World%20Health%20Organisation%20>

7.0 Appendix

Appendix A Link For GitHub, E-Portfolio

- a. GitHub Link:

https://github.com/NiesHW/SECB3203_P4B/tree/main/Group_Project/Group_4

- b. E-Portfolio Link(LEE RONG XIAN):

<https://eportfolio.utm.my/user/lee-rong-xian/secb3203-01-programming-for-bioinformati>
[cs](#)

- c. E-Portfolio Link(LU QI YAN):

Appendix B Overall Discussion(Requested By Client)



FACULTY OF COMPUTING

SECB3203-01

PROGRAMMING FOR BIOINFORMATICS

TITLE:

Prediction of Lung Cancer using Recursive Feature Elimination

LECTURER:

DR. NIES HUI WEN

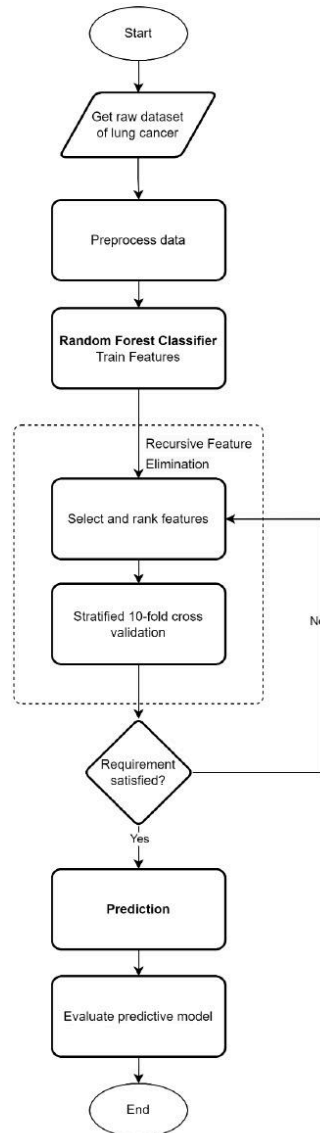
GROUP 4

GROUP MEMBERS:

NAME	MATRIC NUMBER
LEE RONG XIAN	A21EC0043
LU QI YAN	A21EC0049

Discussion

Throughout the project, we are following closely the flowchart of our project as a guide. The flowchart of the proposed approach is shown below.



1.0 Get raw dataset of the lung cancer

The dataset we use in this project can be found at

<https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer>.

The raw dataset consists of 309 rows and 16 columns. The 16 features are as below:

```
GENDER      object
AGE         int64
SMOKING      int64
YELLOW_FINGERS int64
ANXIETY      int64
PEER_PRESSURE int64
CHRONIC_DISEASE int64
FATIGUE      int64
ALLERGY      int64
WHEEZING     int64
ALCOHOL_CONSUMING int64
COUGHING     int64
SHORTNESS_OF_BREATH int64
SWALLOWING_DIFFICULTY int64
CHEST_PAIN   int64
LUNG_CANCER  object
dtype: object
```

2.0 Data Preprocessing

The first step in data processing is to handle the duplicated data. A total of 33 duplicated data in the dataset are identified and dropped. This is because the duplicates are taking up unnecessary storage space and may lead to inconsistencies during the model training. By removing the duplicated data, we are able to ensure a higher accuracy of the data analysis result.

```
Total duplicate values: 33
GENDER  AGE  SMOKING  YELLOW_FINGERS  ANXIETY  PEER_PRESSURE  \
0      M   69      1             2             2             1
1      M   74      2             1             1             1
2      F   59      1             1             1             2
3      M   63      2             2             2             1
4      F   63      1             2             1             1
...     ...   ...      ...             ...             ...
279     F   59      1             2             2             2
280     F   59      2             1             1             1
281     M   55      2             1             1             1
282     M   46      1             2             2             1
283     M   60      1             2             2             1
```

The second step is to handle the missing values as the null values would result in data inconsistencies. However, there are no null values in this dataset. Next, we do data normalization on the “Age” feature. The method used is Min-Max scaling which transforms the data to a range

between 0 and 1(change integer to floating point datatype), making it easier to interpret the relative importance of values.

```
0      0.727273
1      0.803030
2      0.575758
3      0.636364
4      0.636364
...
279    0.575758
280    0.575758
281    0.515152
282    0.378788
283    0.590909
Name: Normalized_Age, Length: 276, dtype: float64
```

Then, we do data binning where we group the “age” feature into 3 categories which are “20-40 years old”, “41-60 years old” and “above 60 years old”. It is shown that the continuous data value which is “age” is grouped into discrete intervals. This is to reduce the complexity of data and make it more manageable and easier to analyze. Lastly, we create the indicator variables for our dataset whereby we convert the categorical data into numerical values. The indicator variable is set for 3 attributes namely, Gender, Age Groups, and Lung Cancer. As we can visualize in the result below, the boolean values (True and False) are converted into integer values (1 and 0) for the 3 features: Gender, Age Groups, and Lung Cancer. Besides, it is shown that the continuous numerical data which is “Age” and “Normalized Age” remain the same. Setting indicator variables is to ease the data analysis process and to ensure a clearer representation of the dataset.

```

GENDER_F GENDER_M AGE Normalized_Age AGE_GROUPS_[20-40] years \
0 0 1 69 0.727273 0
1 0 1 74 0.808080 0
2 1 0 59 0.575758 0
3 0 1 63 0.636364 0
4 1 0 63 0.636364 0
... ..
279 1 0 59 0.575758 0
280 1 0 59 0.575758 0
281 0 1 55 0.515152 0
282 0 1 46 0.378788 0
283 0 1 60 0.590909 0

AGE_GROUPS_[41-60] years AGE_GROUPS_[60+] years SMOKING \
0 0 1
1 0 1 2
2 1 0 1
3 0 1 2
4 0 1 1
... ..
279 1 0 1
280 1 0 2
281 1 0 2
282 1 0 1
283 1 0 1

YELLOW_FINGERS ANXIETY ... FATIGUE ALLERGY WHEEZING \
0 2 2 ... 2 1 2
1 1 1 ... 2 2 1
2 1 1 ... 2 1 2
3 2 2 ... 1 1 1
4 2 1 ... 1 1 2
... ..
279 2 2 ... 1 2 2
280 1 1 ... 2 2 1
281 1 1 ... 2 2 1
282 2 2 ... 1 1 1
283 2 2 ... 2 1 2

ALCOHOL-CONSUMING COUGHING SHORTNESS OF BREATH SWALLOWING DIFFICULTY \
0 2 2 2
1 1 1 2
2 1 2 2
3 2 1 1
4 1 2 2
... ..
279 1 2 1 2
280 1 1 2 1
281 1 1 2 1
282 1 1 1 2
283 2 2 2 2

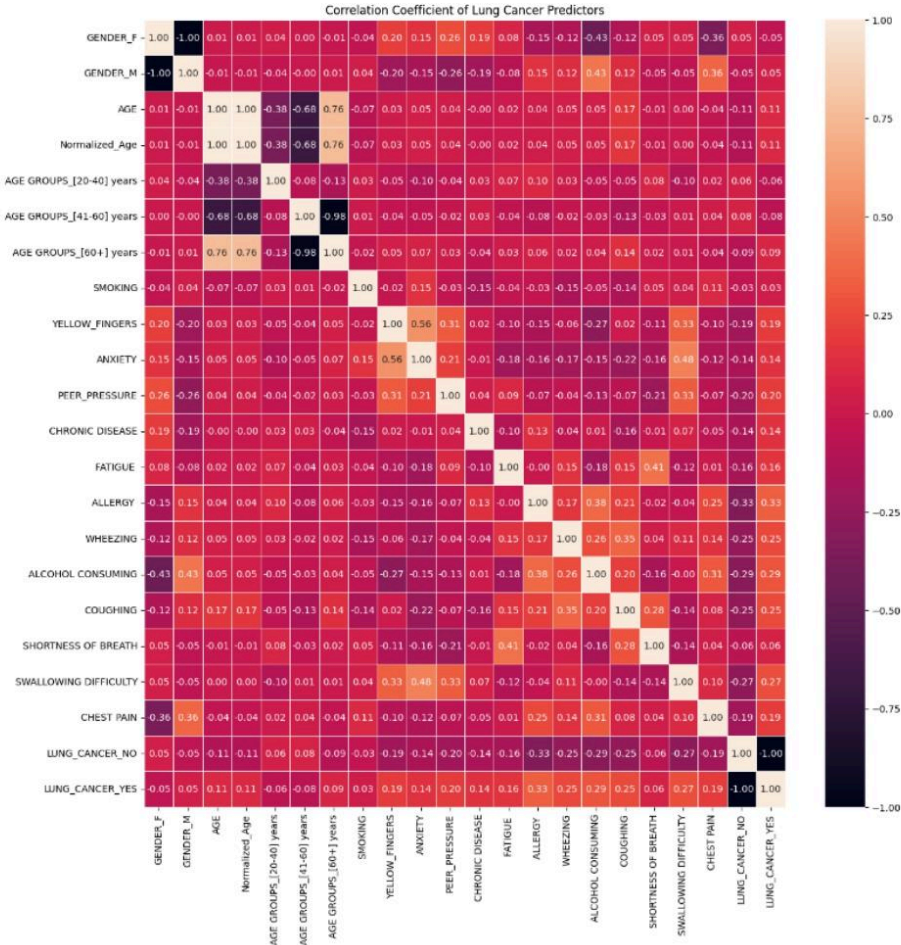
CHEST PAIN LUNG_CANCER_NO LUNG_CANCER_YES
0 2 0 1
1 2 0 1
2 2 1 0
3 2 1 0
4 1 1 0
... ..
279 1 0 1
280 1 1 0
281 2 1 0
282 2 1 0
283 2 0 1

```

[276 rows x 22 columns]

In a nutshell, we have obtained 276 rows and 22 columns after data preprocessing.

Next, we produced a heatmap to show the predictors' correlation coefficient. This is for us to visualize the strength of associations between data variables in a way clearer.



3.0 Model Development

Firstly, we identify the features that are not required for future analysis. These features include AGE, NORMALIZED_AGE, and LUNG_CANCER. Hence, these columns are dropped. This is because AGE, NORMALIZED_AGE and AGE GROUPS represent the same information, keeping them might cause the data to be redundant whereas LUNG_CANCER is not a relevant

predictor or we can say the target variable for the analysis. Then, we are using the 15 features below to train the model.

```
Index(['GENDER', 'AGE GROUPS', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY',  
      'PEER_PRESSURE', 'CHRONIC DISEASE', 'FATIGUE ', 'ALLERGY ', 'WHEEZING',  
      'ALCOHOL CONSUMING', 'COUGHING', 'SHORTNESS OF BREATH',  
      'SWALLOWING DIFFICULTY', 'CHEST PAIN'],  
      dtype='object')
```

In this case, we split the dataset into a train set (70%) and a test set (30%) because this contributes to the highest accuracy of our predictive model.

3.1 Select and rank features

We use RandomForestClassifier as the estimator to carry out the recursive feature elimination. The number of features to select is set to 10 because it provides enough data for training and testing and it also may result in a reliable performance in our model. The eliminated features are represented by false whereas 10 relevant features are represented by true.

	columns	Kept
0	GENDER	False
1	AGE GROUPS	True
2	SMOKING	False
3	YELLOW_FINGERS	True
4	ANXIETY	True
5	PEER_PRESSURE	True
6	CHRONIC DISEASE	True
7	FATIGUE	True
8	ALLERGY	True
9	WHEEZING	False
10	ALCOHOL CONSUMING	True
11	COUGHING	False
12	SHORTNESS OF BREATH	True
13	SWALLOWING DIFFICULTY	False
14	CHEST PAIN	True

Then, the features are ranked. This helps us to identify the most relevant features that cause lung cancer.

	columns	Kept
0	GENDER	3
1	AGE GROUPS	1
2	SMOKING	5
3	YELLOW_FINGERS	1
4	ANXIETY	1
5	PEER_PRESSURE	1
6	CHRONIC DISEASE	1
7	FATIGUE	1
8	ALLERGY	1
9	WHEEZING	2
10	ALCOHOL CONSUMING	1
11	COUGHING	6
12	SHORTNESS OF BREATH	1
13	SWALLOWING DIFFICULTY	4
14	CHEST PAIN	1

3.2 Stratified 10-fold cross-validation

Then, we evaluate the model performance by using the Stratified 10-fold cross-validation.

The evaluation result is as below.

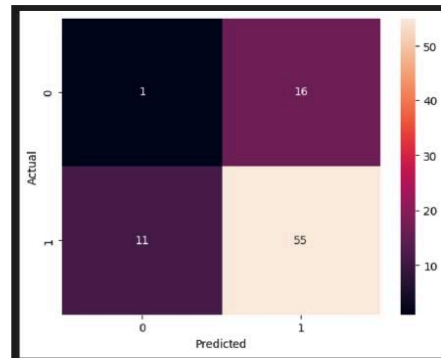
Metrics	Score
Mean score	0.89
Accuracy	0.94
Precision	0.96
Recall	0.97

In this project, the reason that a train set (70%) and a test set (30%) are used to train the model is that they contribute to the highest accuracy of our predictive model. Therefore, it is proven in the table below:

Train set	Test set	Accuracy
0.10	0.90	0.89
0.20	0.80	0.84
0.30	0.70	0.89
0.40	0.60	0.90
0.50	0.50	0.91
0.60	0.40	0.93
0.70	0.30	0.94
0.80	0.20	0.91
0.90	0.10	0.89

4.0 Predictive model evaluation

In this project, a confusion matrix is applied to evaluate the predictive model. It can help to show how many predictions are correct and how many are incorrect. The result is shown in the figure below. From the figure, we can see that there is 1 in the True Negative region, 16 in the False Positive region, 11 in the False Negative region, and 55 in the True Positive region.



The scores of the key metrics of the confusion matrix are shown below.

Metrics	Score
Accuracy	0.67
Precision	0.77
Recall	0.83

To wrap up the discussion, it is proven that the project successfully achieved its objectives by developing an accurate lung cancer prediction model, applying Recursive Feature Elimination (RFE) to systematically eliminate irrelevant features, and demonstrating the effectiveness of RFE in selecting the most relevant features for enhanced predictive performance.

5.0 Reflection

This project requires us to understand the dataset that we have chosen and to understand the algorithm that we have chosen to do. From this, we learned how to find suitable datasets online. At the same time, it is important to find a related thesis or scholarly paper that can help us to understand the algorithms. By doing this, we can fully understand the algorithms that we choose which is the algorithm for Recursive Features Elimination. After understanding and referring to the sources that we found, we learned to modify the algorithm based on our requirements. In this project, what we do is play around with the parameters in the algorithm so that we can get the highest accuracy with the specific value of the parameters. In summary, this research has given us practical experience and insightful knowledge in the field of machine learning, with a particular emphasis on the application of recursive feature elimination (RFE). Through this project, we have improved our coding abilities and become more skilled at creating algorithms and composing code specifically for the RFE method.

Feedback from Dr Sharin Hazlin Binti Huspi

FEEDBACK:

Overall, the group has done a good job in learning and applying what they need to do for the project. It was easy to guide them as they were able to understand what they needed to do. They were also interested in the area, thus they were able to understand the process and develop the algorithm.

However they still need to learn to discuss their findings. I think there are still areas to learn in doing the discussion analysis of the results.

CLIENT NAME	Sharin Hazlin Huspi		
SIGNATURE	<i>Shmie</i>	DATE	11/02/2024