GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY

PROJECT TITLE: Rainfall Prediction

MINOR PROJECT

ICT-497

Submitted in partial fulfilment of the requirements

for the award of the degree of

Bachelor of Technology

in

Computer Science Engineering

at

UNIVERSITY SCHOOL OF INFORMATION, COMMUNICATION AND TECHNOLOGY

Supervisor: Dr. Jyotsana Yadav

Submitted By: Bharat Sahay

Enrollment No: 01816403221

Programme: B.Tech. CSE 7th Sem

Table of Contents

DECLARATION

I, **Bharat Sahay** , a student of Bachelor of Technology in Computer Science & Engineering at the **University School of Information, Communication & Technology (USICT), Guru Gobind Singh Indraprastha University, New Delhi**, hereby declare that the work presented in this minor project report titled **"Lung Cancer Prediction using Machine Learning"** is an original and authentic effort undertaken by me under the technical guidance of **Dr. Jyotsna Yadav**.

I affirm that due acknowledgment has been made wherever necessary, and the information used in this project has been properly cited and referenced. The progress report has been checked by Turnitin Plagiarism Detection software at GGSIPU. The work is free from any sort of plagiarism, fabrication, falsification, or copyright issues, and the similarity of the contents identified is as per UGC norms.

I declare that the work in this project has not been submitted in full or in part for any diploma or degree course of this University or any other University/Institute to the best of my knowledge and belief. I shall be solely responsible for any copyright infringement or plagiarism, if any, in the said project or any dispute arising out of my work. My supervisor or anyone else shall not be held responsible for any full or partial violation of copyright/intellectual property rights/plagiarism issues. All the inputs and suggestions by the supervisor and SRC have been incorporated into the project report.

Bharat Sahay
B.Tech CSE Student

(Enrollment No: 01816403221)
University School of Information, Communication & Technology,
Guru Gobind Singh Indraprastha University,
Sector-16C, Dwarka, New Delhi, India-110078

Date-25/10/2024

UNIVERSITY SCHOOL OF INFORMATION, COMMUNICATION AND TECHNOLOGY

GURU GOBIND SINGH INDRAPRASTHA UNIVERSITY

Certificate of Originality

This is to certify that the work embodied in this project report titled **"Lung Cancer Detection using Machine Learning"** is an original piece of work carried out by **Bharat Sahay**, a student of Bachelor of Technology in Computer Science & Engineering at the **University School of Information, Communication & Technology (USICT), Guru Gobind Singh Indraprastha University**. This project has been completed under the supervision of **Dr. Jyotsna Yadav**.

The work has not been submitted, in part or in full, for any other degree or diploma at this or any other university, based on the declaration of the candidate.

It is certified that the work being submitted for this project report is free from any form of falsification, fabrication of results, data, claims, copyrights, and plagiarism, to the best of my knowledge and belief.

Dr. Jyotsna Yadav
Supervisor
Student
University School of Information, Communication & Technology
Guru Gobind Singh Indraprastha University,
New Delhi

Bharat Sahay
B.Tech CSE
01816403221

Date-25/10/2024

Abstract

This project focuses on developing a machine learning model for **Lung Cancer Prediction** using the **Survey of 234,580 Cases of lung cancer done by American Cancer Society (ACS) in 2015**.

Lung cancer remains one of **the leading causes of death** worldwide, primarily due to late-stage diagnosis. Early detection is crucial for improving survival rates, and machine learning (ML) offers a promising approach to automate and enhance the diagnostic process. This study explores the **application of machine learning techniques** for the detection of lung cancer using patient data, including clinical attributes and imaging features. **A variety of machine learning algorithms, including Random Forest, Gradient Boosting,** and Support Vector Machines, were implemented to build predictive models based on a dataset that includes both categorical and numerical features. The models were evaluated using common performance metrics such **as accuracy, precision, recall, F1-score, and ROC-AUC.** To address the challenge of class imbalance in the dataset, techniques such as **SMOTE (Synthetic Minority Over-sampling Technique)** were applied to enhance the classifier's performance.

The results indicate that **the Random Forest model achieved an accuracy of 0.84**, with potential improvements through fine-tuning and ensemble methods. This study demonstrates the potential of machine learning to assist healthcare professionals in early and accurate lung cancer diagnosis, thereby contributing to better patient outcomes and personalized treatment strategies. The model can be further improved through the inclusion of additional clinical and imaging data, as well as through advanced deep learning techniques.

The project is implemented using a variety of tools and technologies, including **Python**, **Pandas**, **NumPy**, **Scikit-learn**, and **Matplotlib**, to ensure efficient data preprocessing, modeling, and visualization. Key steps include data preprocessing, exploratory data analysis (EDA), model selection, and performance evaluation. Several machine learning algorithms, such as **Linear Regression**, **Random Forest**, and **Neural Networks**, are applied and evaluated based on performance metrics like **Mean Squared Error (MSE)** and **R-squared**.

Introduction

Lung cancer is one of the most prevalent and deadly types of cancer worldwide, accounting for a significant number of cancer-related deaths each year. It is often diagnosed at an advanced stage, when the chances of successful treatment are considerably lower. As a result, improving early detection is essential for enhancing survival rates and enabling more effective treatment strategies.

Recent advancements in healthcare technologies, particularly in the field of data science and **machine learning (ML)**, have shown great promise in addressing this challenge. Machine learning, a subset of artificial intelligence (AI), enables computers to learn patterns from data and make predictions without explicit programming

This study focuses on utilizing machine learning techniques for the **detection of lung cancer**. The goal is to create predictive models that can accurately classify whether a patient is at risk of having lung cancer based on a set of input features such as demographic information, lifestyle factors, medical history, and symptom observations. To ensure robust and reliable results, a variety of classification algorithms, including **Random Forest (RF)**, **Gradient Boosting (GBM)**, and **Logistic Regression (LR)**, are explored and compared.

One of the major challenges in lung cancer prediction is the class imbalance problem, where the number of healthy patients significantly outweighs the number of cancer-positive cases. To address this, advanced techniques such as **Synthetic Minority Over-sampling Technique (SMOTE)** and **undersampling** are employed to balance the dataset and improve the model's ability to detect lung cancer. Additionally, model optimization techniques like **hyperparameter tuning** and **cross-validation** are used to enhance the accuracy and generalization of the classifiers.

Ultimately, this study contributes to the broader effort of improving cancer detection through AI and aims to provide insights into the future potential of machine learning in the medical field.

Technology Used

The **Lung Cancer Prediction using Machine Learning** project employs a range of modern tools and technologies to build an efficient and accurate machine learning model for cancer prediction. The project is developed using **Python** due to its extensive library support and ease of use in data processing and model development. The following technologies and libraries were utilized:

1. **Pandas**: This powerful data manipulation library is used to handle and preprocess the dataset. Pandas enables efficient cleaning, normalization, and transformation of the rainfall data, making it ready for modeling.
2. **NumPy**: NumPy provides support for high-performance numerical computations. It is used in this project for efficient handling of large datasets and performing mathematical operations, which are crucial in the model training process.
3. **Scikit-learn**: Scikit-learn is a machine learning library that simplifies the implementation of various algorithms. In this project, models such as **Linear Regression**, **Random Forest**, and **Neural Networks** are implemented using Scikit-learn. It also offers tools for model evaluation through metrics like **Mean Squared Error (MSE)** and **R-squared**.
4. **Matplotlib**: For data visualization, Matplotlib is used to generate insightful graphs and heatmaps. These visualizations help compare predicted and actual **cancer result** values, providing clear insights into model performance and future trends.
5. **Seaborn**: In addition to Matplotlib, Seaborn is employed for advanced visualization tasks, such as generating correlation matrices and detailed heatmaps, making exploratory data analysis (EDA) more intuitive.
6. **Jupyter Notebook**: This interactive environment is used for the development and execution of the project code. It allows for easy iteration during the model development process, enabling real-time visualization of results and facilitating rapid prototyping.

By integrating these technologies, the project ensures a robust and scalable solution for lung cancer prediction , making the most of advanced machine learning techniques and effective data handling practices. The combination of these tools enables the creation of an accurate model that can assist in detecting lung cancer probability in early stage .

Work Done

The development of the **Lung Cancer Detection using Machine Learning** project has primarily involved data preprocessing, exploratory data analysis, and model selection. Below is a summary of the work completed thus far:

1. Data Preprocessing:

- Cleaned the dataset by handling missing values to ensure the data was complete and reliable for model training.
- Normalized the dataset to standardize the range of rainfall values, making it suitable for machine learning algorithms.
- Split the data into training and testing sets for accurate model evaluation.

2. Exploratory Data Analysis (EDA):

- Conducted an in-depth analysis of the historical rainfall data to identify patterns, trends, and seasonality.
- Visualized geographical distribution of rainfall using heatmaps and identified correlations between rainfall and different regions of India.
- Summarized rainfall trends over the years to understand the data better before applying machine learning models.

3. Model Development:

- Implemented several machine learning algorithms, including **Random Forest, Gradient Boosting, XGBoost , Naive Bayes Classifiers** to predict future rainfall based on the historical dataset.
- Trained each model on the processed data, comparing performance across various algorithms.
- Focused on tuning hyperparameters to enhance model accuracy.

4. Model Evaluation:

- Evaluated the performance of the models using metrics such as **Evaluation Metrix Mean Squared Error (MSE) and R-squared (R^2)** to determine their predictive power.
- Visualized the predicted versus actual rainfall to gauge the effectiveness of the models and to pinpoint areas for improvement.

This work has established a strong foundation for further model refinement and real-time Cancer Prediction , ensuring the project is on track for successful completion.

Dataset Description:

1. Dataset link = https://www.kaggle.com/datasets/akashnath29/lung-cancer-dataset

2. Detailed Patient Profiles for Lung Cancer Risk Assessment and Analysis

3. This dataset is an invaluable asset in the realm of Health Care, providing a structured foundation for the development of cancer detection models. This dataset exemplifies **the variety of symptoms of Lung Cancer**. Each category within the dataset—**'GENDER', 'AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY', 'PEER_PRESSURE', 'CHRONIC_DISEASE', 'FATIGUE', 'ALLERGY', 'WHEEZING', 'ALCOHOL_CONSUMING', 'COUGHING', 'SHORTNESS_OF_BREATH', 'SWALLOWING_DIFFICULTY', 'CHEST_PAIN'**— • For some of the subdivisions data is from 1950 to 2015.

Dataset Composition

The Lung Cancer Dataset includes a diverse array of symptoms essential for comprehensive analysis and model development. The primary categories of data are as follows:

1. Patient Demographics

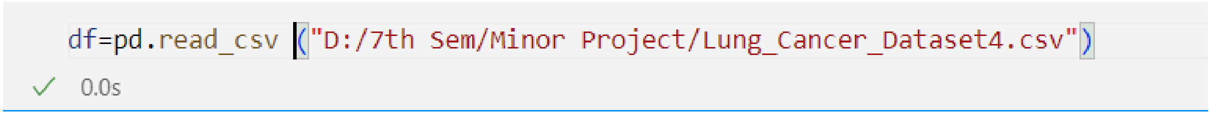
Age: Provides the age at diagnosis, enabling analysis of age-related incidence and outcomes.
Gender: Includes information on patient gender, facilitating gender-based studies.
Smoking Status: Categorized as current smoker, former smoker, or non-smoker, this data is critical for evaluating the impact of smoking on lung cancer risk and progression.

2. Medical History

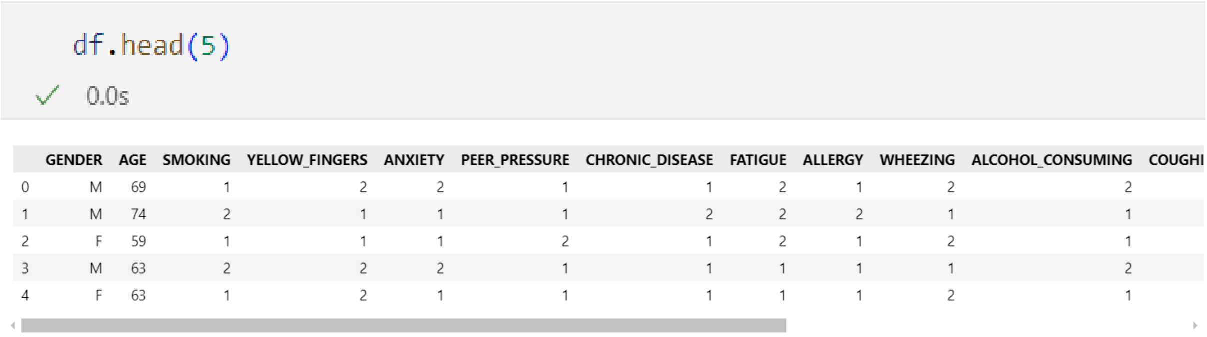
Comorbidities: Details additional health issues such as chronic obstructive pulmonary disease (COPD), which are relevant for treatment planning and prognosis.

3. Clinical Data

Vital Signs: Records of blood pressure, heart rate, respiratory rate, and other vital signs at diagnosis and during treatment.

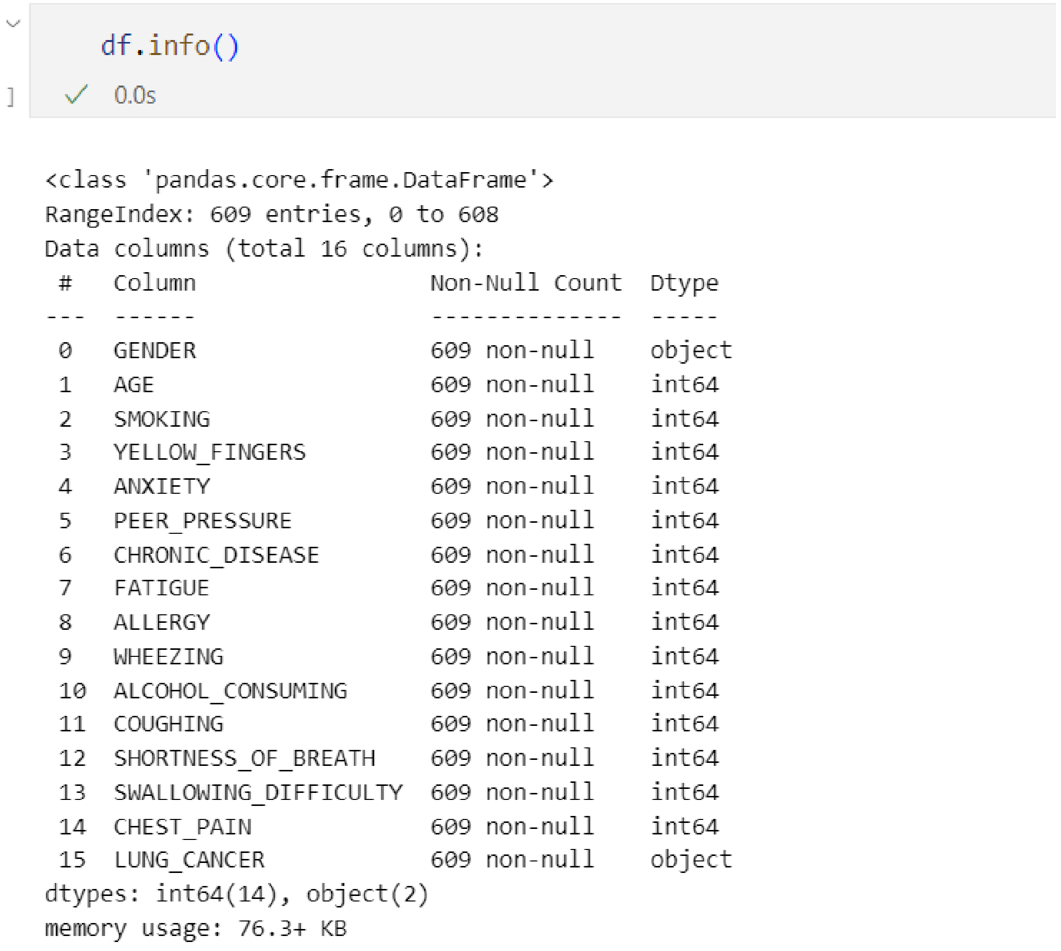


```
df=pd.read_csv ("D:/7th Sem/Minor Project/Lung_Cancer_Dataset4.csv")
```



```
df.head(5)
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC_DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL_CONSUMING	COUGHI
0	M	69	1	2	2	1	1	2	1	2	2	
1	M	74	2	1	1	1	2	2	2	1	1	
2	F	59	1	1	1	2	1	2	1	2	1	
3	M	63	2	2	2	1	1	1	1	1	2	
4	F	63	1	2	1	1	1	1	1	2	1	

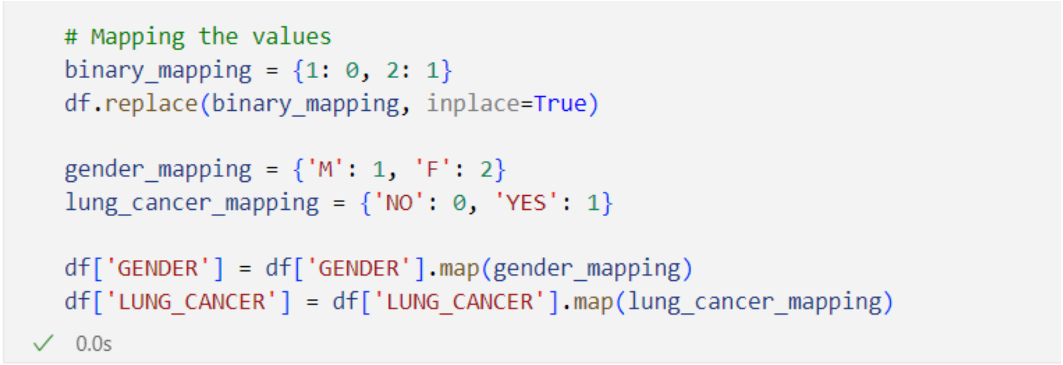


```
df.info()
```

✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 609 entries, 0 to 608
Data columns (total 16 columns):
 #   Column                 Non-Null Count  Dtype 
---  ------                 --------------  ----- 
 0   GENDER                 609 non-null    object
 1   AGE                    609 non-null    int64 
 2   SMOKING                609 non-null    int64 
 3   YELLOW_FINGERS         609 non-null    int64 
 4   ANXIETY                609 non-null    int64 
 5   PEER_PRESSURE          609 non-null    int64 
 6   CHRONIC_DISEASE        609 non-null    int64 
 7   FATIGUE                609 non-null    int64 
 8   ALLERGY                609 non-null    int64 
 9   WHEEZING               609 non-null    int64 
 10  ALCOHOL_CONSUMING      609 non-null    int64 
 11  COUGHING               609 non-null    int64 
 12  SHORTNESS_OF_BREATH    609 non-null    int64 
 13  SWALLOWING_DIFFICULTY  609 non-null    int64 
 14  CHEST_PAIN             609 non-null    int64 
 15  LUNG_CANCER            609 non-null    object
dtypes: int64(14), object(2)
memory usage: 76.3+ KB
```

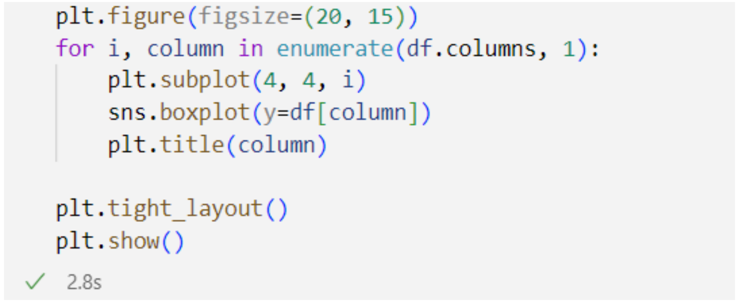
Preprocessing Data : Hot Labeling



```
# Mapping the values
binary_mapping = {1: 0, 2: 1}
df.replace(binary_mapping, inplace=True)

gender_mapping = {'M': 1, 'F': 2}
lung_cancer_mapping = {'NO': 0, 'YES': 1}

df['GENDER'] = df['GENDER'].map(gender_mapping)
df['LUNG_CANCER'] = df['LUNG_CANCER'].map(lung_cancer_mapping)
```

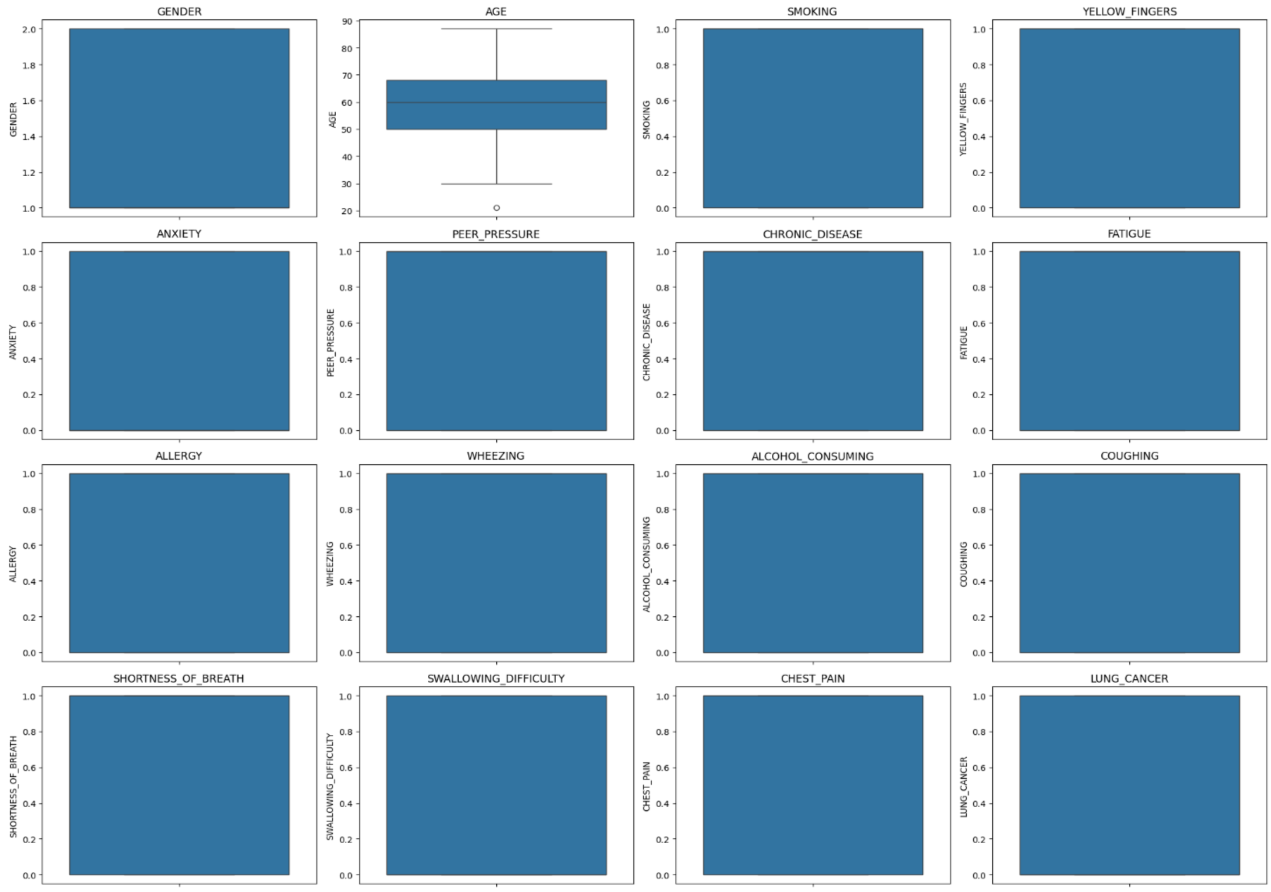
✓ 0.0s

Plotting box plots for each feature :

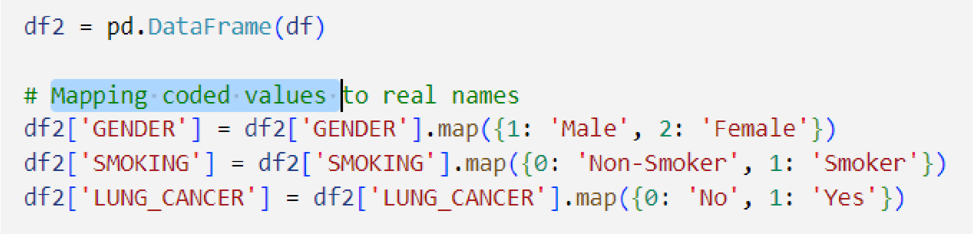


```
plt.figure(figsize=(20, 15))
for i, column in enumerate(df.columns, 1):
    plt.subplot(4, 4, i)
    sns.boxplot(y=df[column])
    plt.title(column)

plt.tight_layout()
plt.show()
```

✓ 2.8s

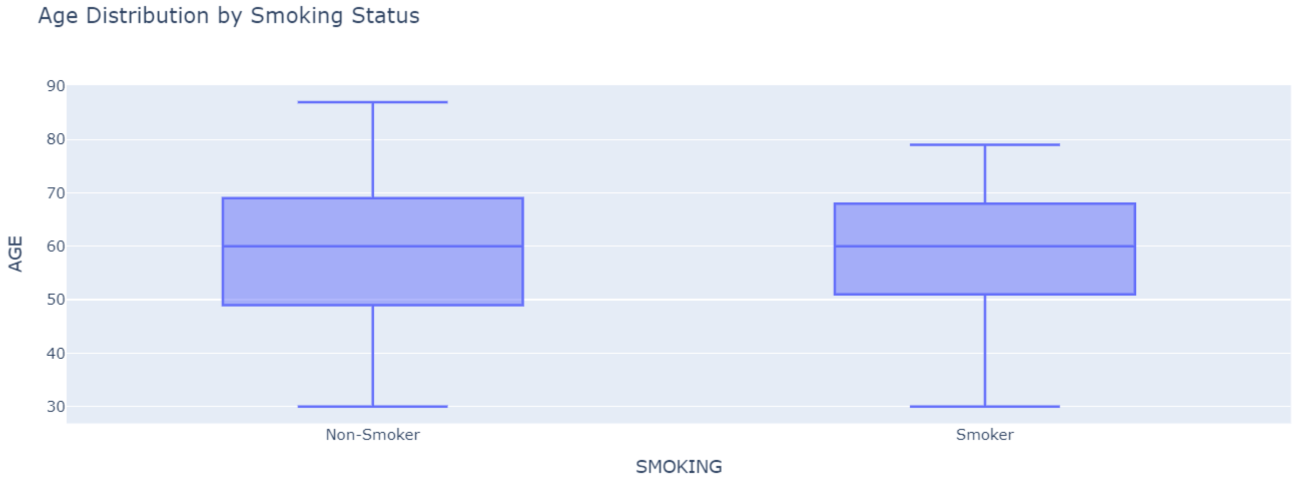


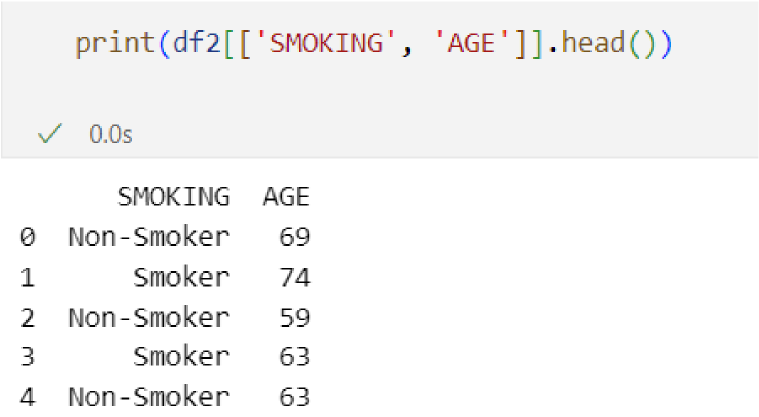
Preprocessing Data : Mapping coded values



```
df2 = pd.DataFrame(df)

# Mapping coded values to real names
df2['GENDER'] = df2['GENDER'].map({1: 'Male', 2: 'Female'})
df2['SMOKING'] = df2['SMOKING'].map({0: 'Non-Smoker', 1: 'Smoker'})
df2['LUNG_CANCER'] = df2['LUNG_CANCER'].map({0: 'No', 1: 'Yes'})
```

Box Plot : Age Distribution by Smoking Status



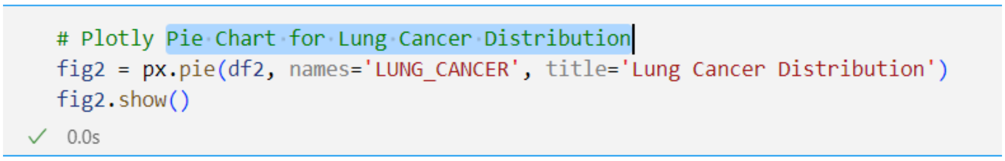


```
print(df2[['SMOKING', 'AGE']].head())
```

✓ 0.0s

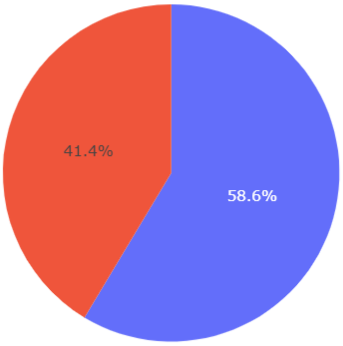
```
      SMOKING  AGE
0  Non-Smoker   69
1      Smoker   74
2  Non-Smoker   59
3      Smoker   63
4  Non-Smoker   63
```

Pie Chart : Lung Cancer Distribution

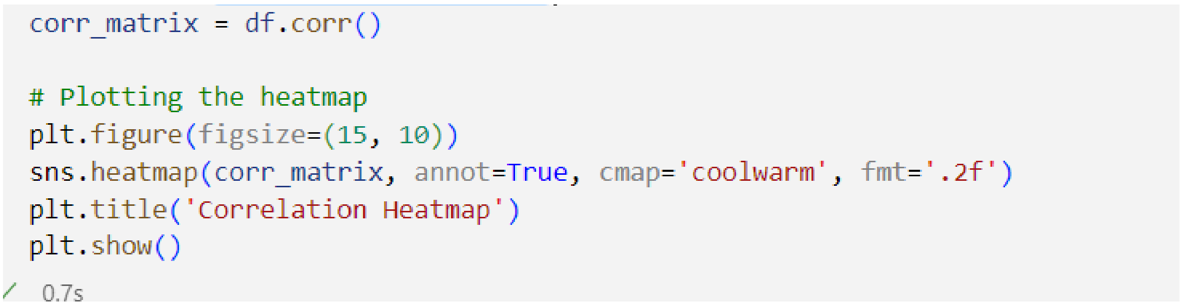


```
# Plotly Pie Chart for Lung Cancer Distribution
fig2 = px.pie(df2, names='LUNG_CANCER', title='Lung Cancer Distribution')
fig2.show()
```

✓ 0.0s

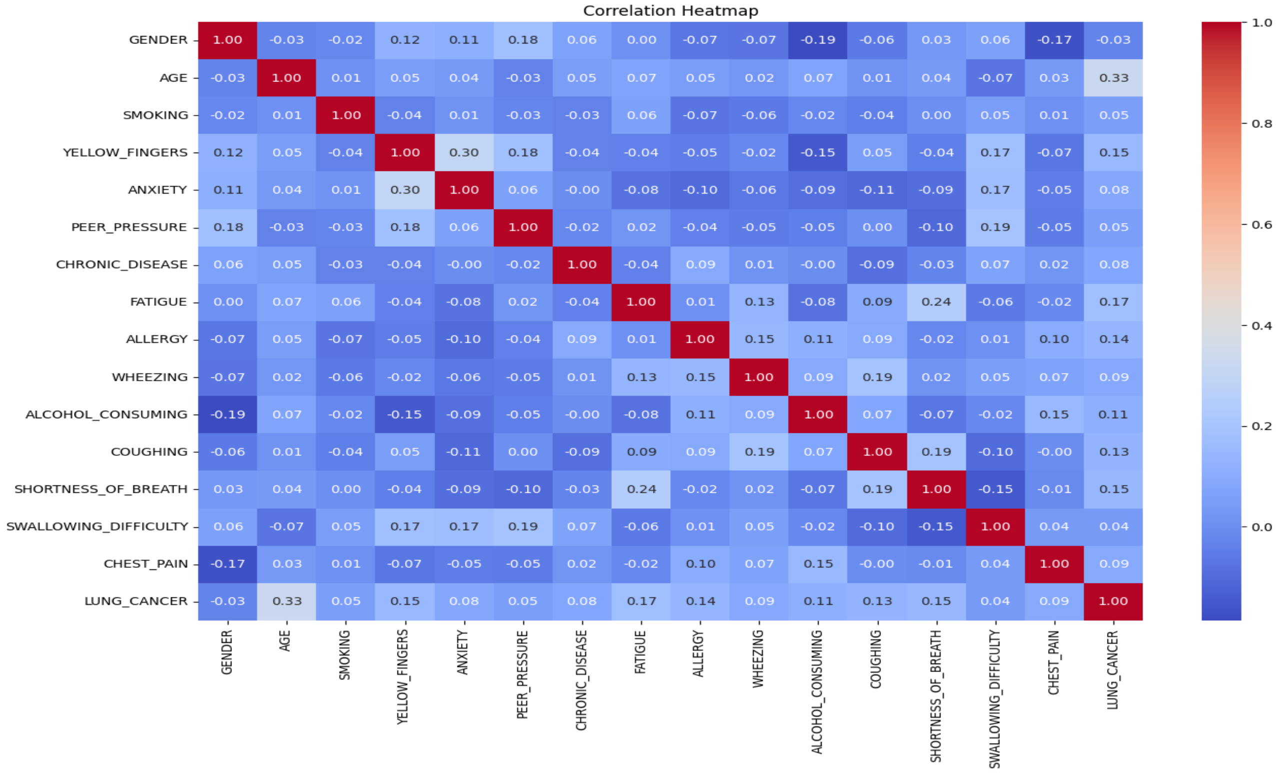


The Correlation Matrix :



```
corr_matrix = df.corr()

# Plotting the heatmap
plt.figure(figsize=(15, 10))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.title('Correlation Heatmap')
plt.show()
```

✓ 0.7s

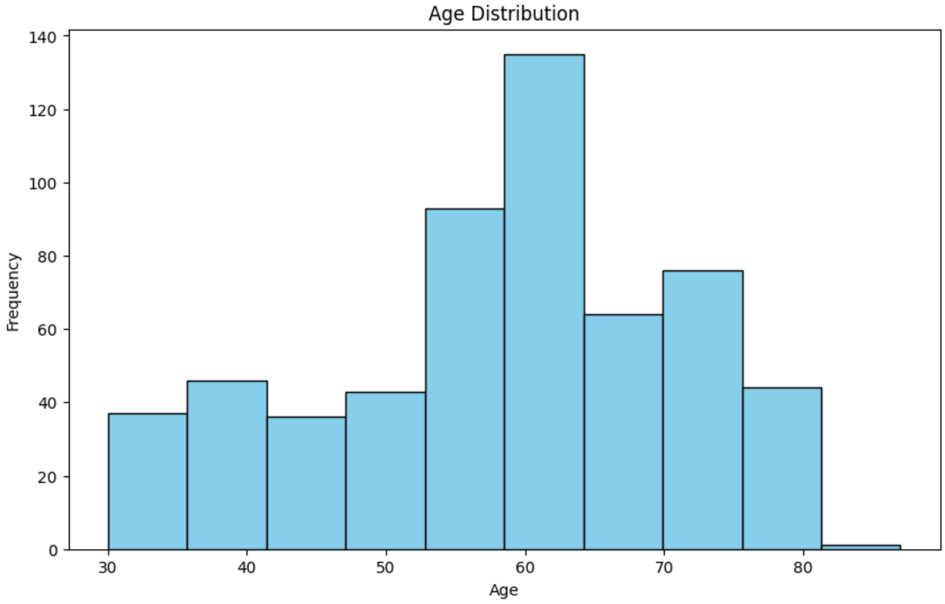


Correlation Heatmap

	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
GENDER	1.00	-0.03	-0.02	0.12	0.11	0.18	0.06	0.00	-0.07	-0.07	-0.19	-0.06	0.03	0.06	-0.17	-0.03
AGE	-0.03	1.00	0.01	0.05	0.04	-0.03	0.05	0.07	0.05	0.02	0.07	0.01	0.04	-0.07	0.03	0.33
SMOKING	-0.02	0.01	1.00	-0.04	0.01	-0.03	-0.03	0.06	-0.07	-0.06	-0.02	-0.04	0.00	0.05	0.01	0.05
YELLOW_FINGERS	0.12	0.05	-0.04	1.00	0.30	0.18	-0.04	-0.04	-0.05	-0.02	-0.15	0.05	-0.04	0.17	-0.07	0.15
ANXIETY	0.11	0.04	0.01	0.30	1.00	0.06	-0.00	-0.08	-0.10	-0.06	-0.09	-0.11	-0.09	0.17	-0.05	0.08
PEER_PRESSURE	0.18	-0.03	-0.03	0.18	0.06	1.00	-0.02	0.02	-0.04	-0.05	-0.05	0.00	-0.10	0.19	-0.05	0.05
CHRONIC_DISEASE	0.06	0.05	-0.03	-0.04	-0.00	-0.02	1.00	-0.04	0.09	0.01	-0.00	-0.09	-0.03	0.07	0.02	0.08
FATIGUE	0.00	0.07	0.06	-0.04	-0.08	0.02	-0.04	1.00	0.01	0.13	-0.08	0.09	0.24	-0.06	-0.02	0.17
ALLERGY	-0.07	0.05	-0.07	-0.05	-0.10	-0.04	0.09	0.01	1.00	0.15	0.11	0.09	-0.02	0.01	0.10	0.14
WHEEZING	-0.07	0.02	-0.06	-0.02	-0.06	-0.05	0.01	0.13	0.15	1.00	0.09	0.19	0.02	0.05	0.07	0.09
ALCOHOL_CONSUMING	-0.19	0.07	-0.02	-0.15	-0.09	-0.05	-0.00	-0.08	0.11	0.09	1.00	0.07	-0.07	-0.02	0.15	0.11
COUGHING	-0.06	0.01	-0.04	0.05	-0.11	0.00	-0.09	0.09	0.09	0.19	0.07	1.00	0.19	-0.10	-0.00	0.13
SHORTNESS_OF_BREATH	0.03	0.04	0.00	-0.04	-0.09	-0.10	-0.03	0.24	-0.02	0.02	-0.07	0.19	1.00	-0.15	-0.01	0.15
SWALLOWING_DIFFICULTY	0.06	-0.07	0.05	0.17	0.17	0.19	0.07	-0.06	0.01	0.05	-0.02	-0.10	-0.15	1.00	0.04	0.04
CHEST_PAIN	-0.17	0.03	0.01	-0.07	-0.05	-0.05	0.02	-0.02	0.10	0.07	0.15	-0.00	-0.01	0.04	1.00	0.09
LUNG_CANCER	-0.03	0.33	0.05	0.15	0.08	0.05	0.08	0.17	0.14	0.09	0.11	0.13	0.15	0.04	0.09	1.00

ue

Histogram : Age



Age Distribution

]

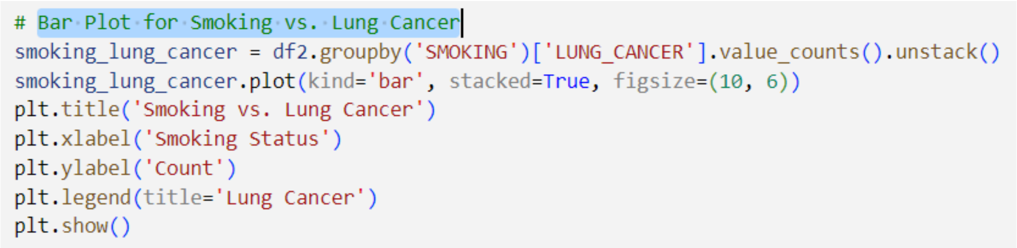
'OCT','NOV','DEC','A

- ıs.

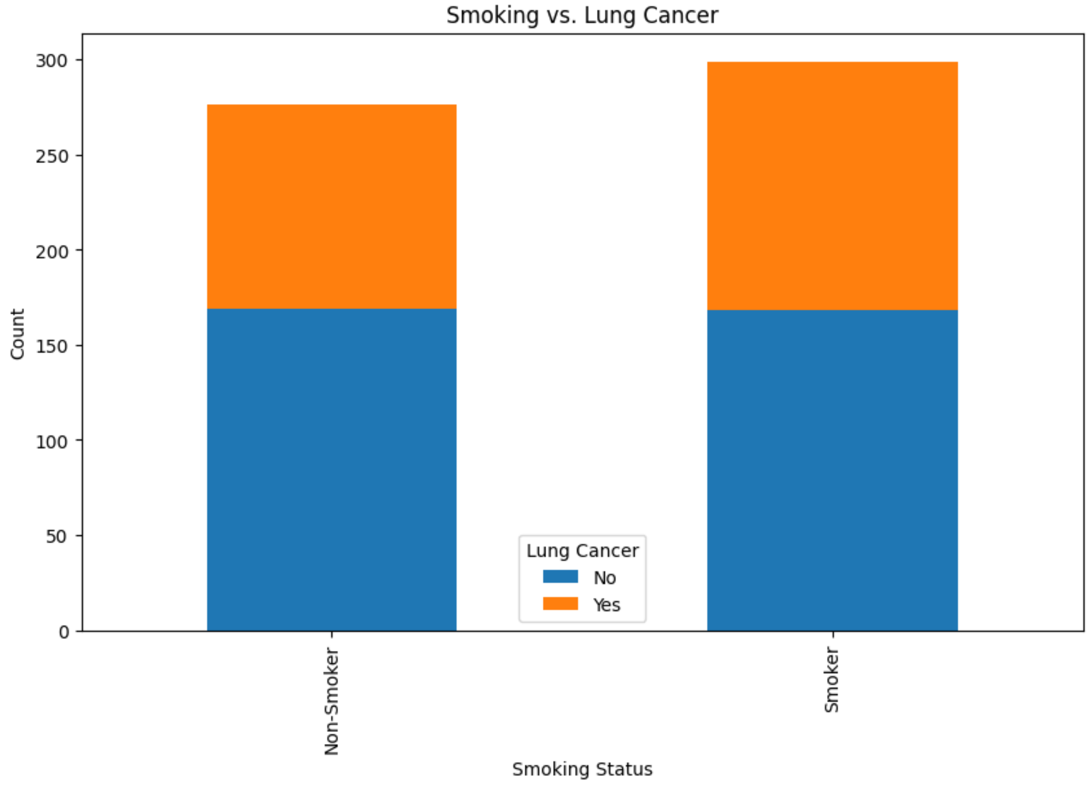
- ɛmber then the

a

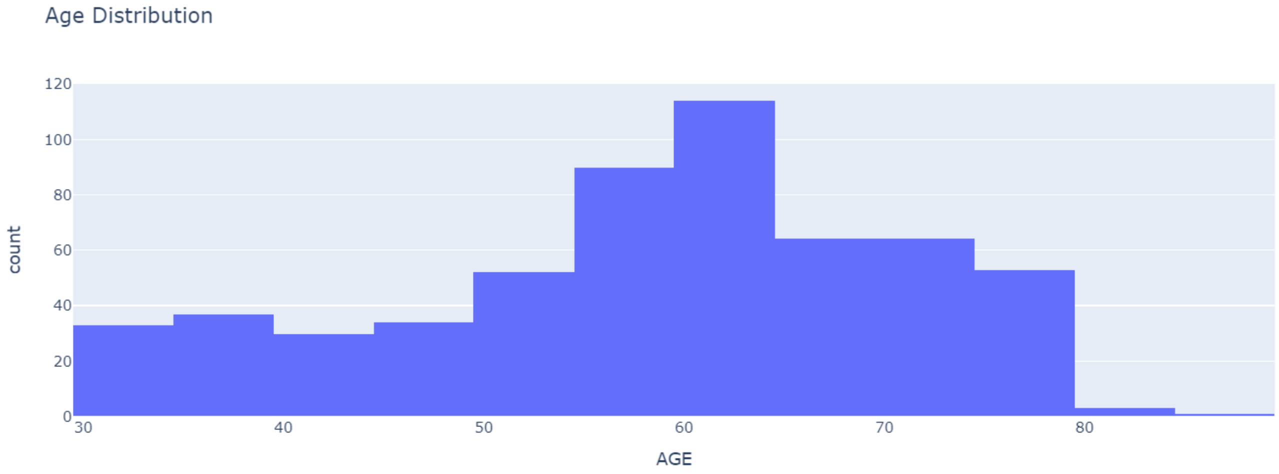
Bar Plot: Smoking vs. Lung Cancer



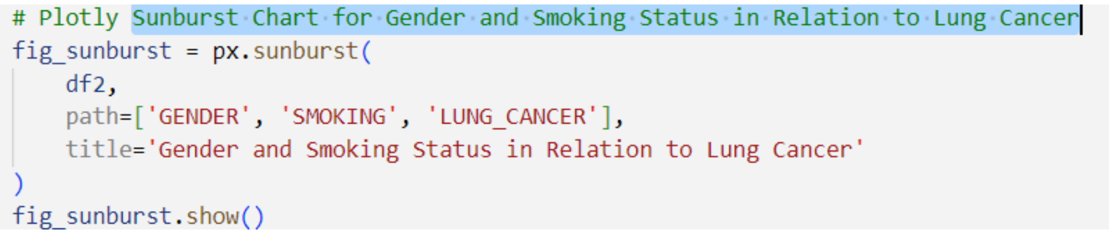
```
# Bar Plot for Smoking vs. Lung Cancer
smoking_lung_cancer = df2.groupby('SMOKING')['LUNG_CANCER'].value_counts().unstack()
smoking_lung_cancer.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.title('Smoking vs. Lung Cancer')
plt.xlabel('Smoking Status')
plt.ylabel('Count')
plt.legend(title='Lung Cancer')
plt.show()
```



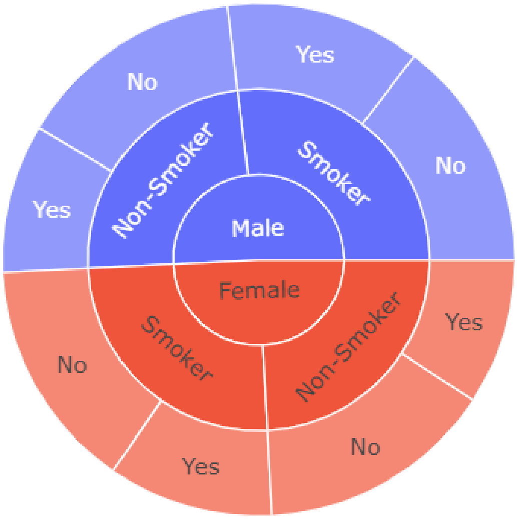
Histogram: Age Distribution



SunBurst Chart : Gender and Smoking Status w.r.t Lung Cancer



```
# Plotly Sunburst Chart for Gender and Smoking Status in Relation to Lung Cancer
fig_sunburst = px.sunburst(
    df2,
    path=['GENDER', 'SMOKING', 'LUNG_CANCER'],
    title='Gender and Smoking Status in Relation to Lung Cancer'
)
fig_sunburst.show()
```



Predictions:

• For prediction, we formatted the data based on various patient attributes such as demographic information, lifestyle factors, and medical history to predict whether a patient is likely to develop **lung cancer**.
• For all the experiments, we used an 80:20 training and testing ratio to split the dataset.

i. XGBoost

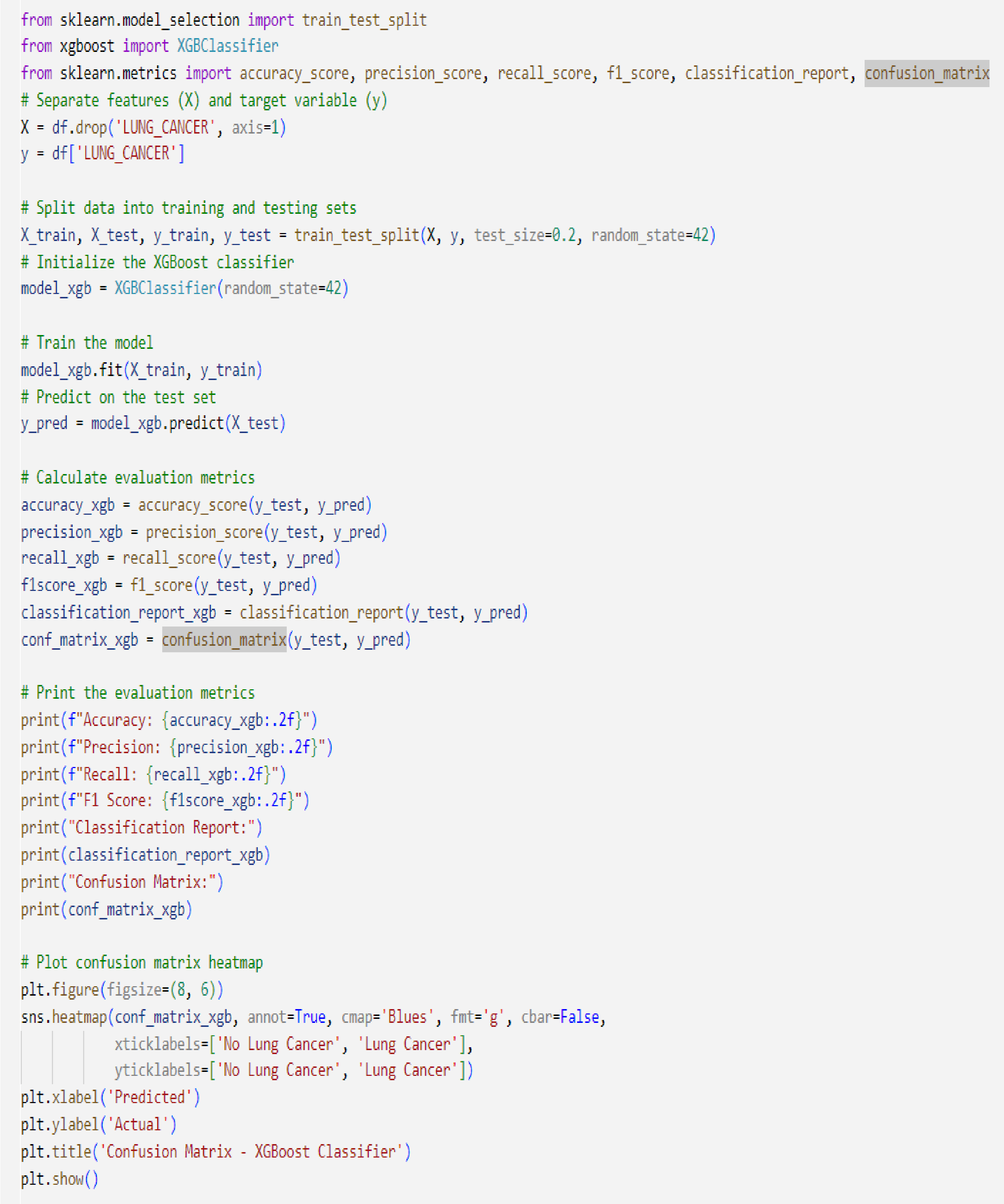
ii. Gradient Boosting Classifier

iii. Random Forest Classifier

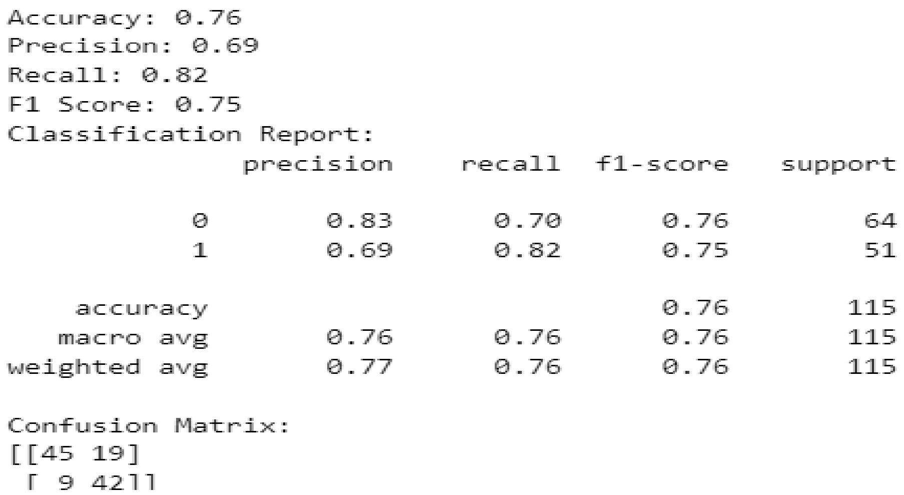
iv. Naive Bayes Classifiers

• Testing metrics: We used **accuracy**, **precision**, **recall**, **F1-score**, and **ROC-AUC** to evaluate the performance of the models.
• We also visualized the predicted and actual outcomes using **confusion matrices** and **ROC curves** to compare the model's performance.
• We conducted two types of training: one with the complete dataset and another by training the model on a specific subset of data (e.g., based on patient demographics or symptom history).
• We reported the **mean** and **standard deviation** of the evaluation metrics, with the first set representing the ground truth and the second set representing the model predictions**.**

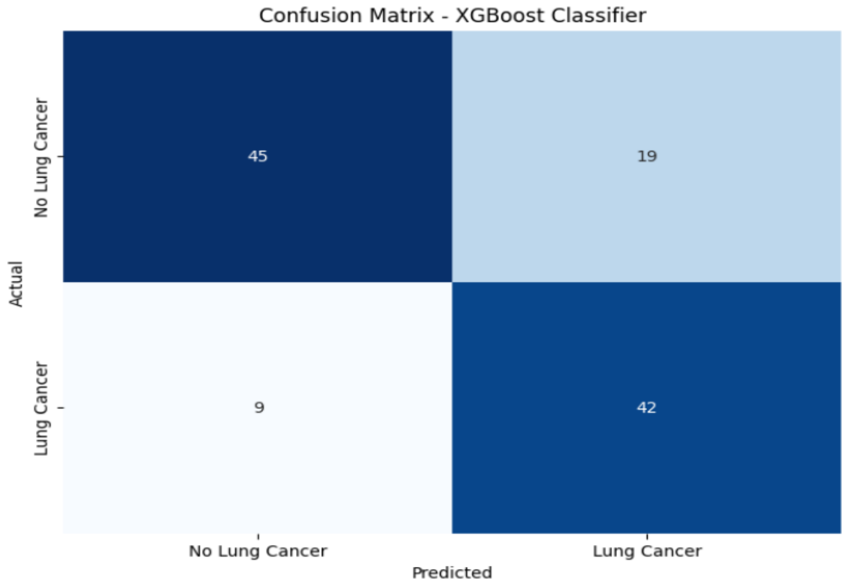
XGBoost :



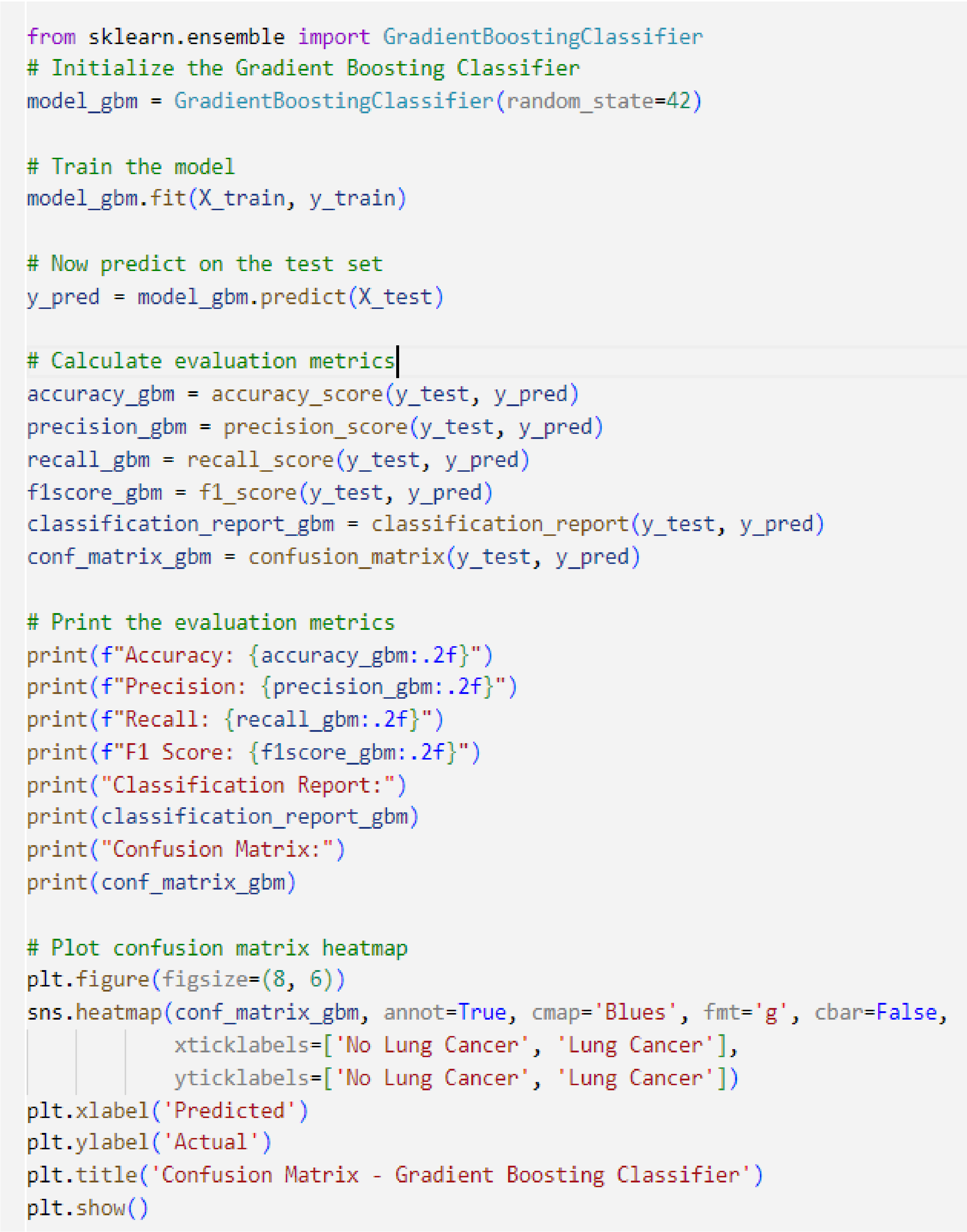
```
from sklearn.model_selection import train_test_split
from xgboost import XGBClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, classification_report, confusion_matrix
# Separate features (X) and target variable (y)
X = df.drop('LUNG_CANCER', axis=1)
y = df['LUNG_CANCER']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
# Initialize the XGBoost classifier
model_xgb = XGBClassifier(random_state=42)

# Train the model
model_xgb.fit(X_train, y_train)
# Predict on the test set
y_pred = model_xgb.predict(X_test)

# Calculate evaluation metrics
accuracy_xgb = accuracy_score(y_test, y_pred)
precision_xgb = precision_score(y_test, y_pred)
recall_xgb = recall_score(y_test, y_pred)
f1score_xgb = f1_score(y_test, y_pred)
classification_report_xgb = classification_report(y_test, y_pred)
conf_matrix_xgb = confusion_matrix(y_test, y_pred)

# Print the evaluation metrics
print(f"Accuracy: {accuracy_xgb:.2f}")
print(f"Precision: {precision_xgb:.2f}")
print(f"Recall: {recall_xgb:.2f}")
print(f"F1 Score: {f1score_xgb:.2f}")
print("Classification Report:")
print(classification_report_xgb)
print("Confusion Matrix:")
print(conf_matrix_xgb)

# Plot confusion matrix heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_xgb, annot=True, cmap='Blues', fmt='g', cbar=False,
            xticklabels=['No Lung Cancer', 'Lung Cancer'],
            yticklabels=['No Lung Cancer', 'Lung Cancer'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - XGBoost Classifier')
plt.show()
```



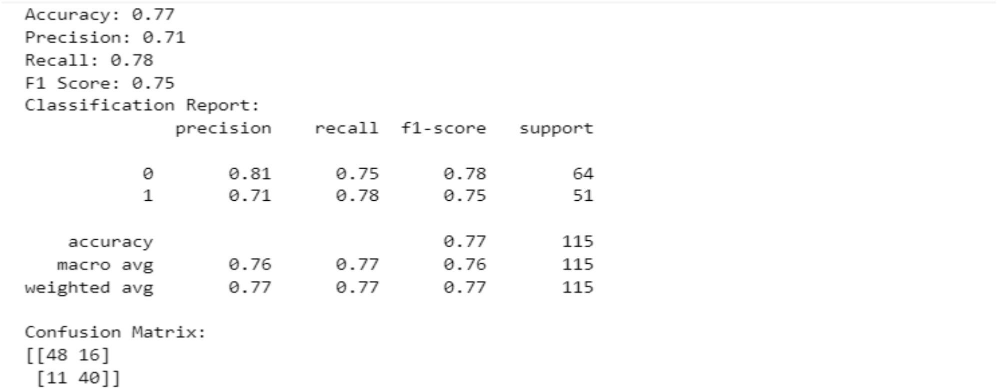
```
Accuracy: 0.76
Precision: 0.69
Recall: 0.82
F1 Score: 0.75
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.70      0.76        64
           1       0.69      0.82      0.75        51

    accuracy                           0.76       115
   macro avg       0.76      0.76      0.76       115
weighted avg       0.77      0.76      0.76       115

Confusion Matrix:
[[45 19]
 [ 9 42]]
```



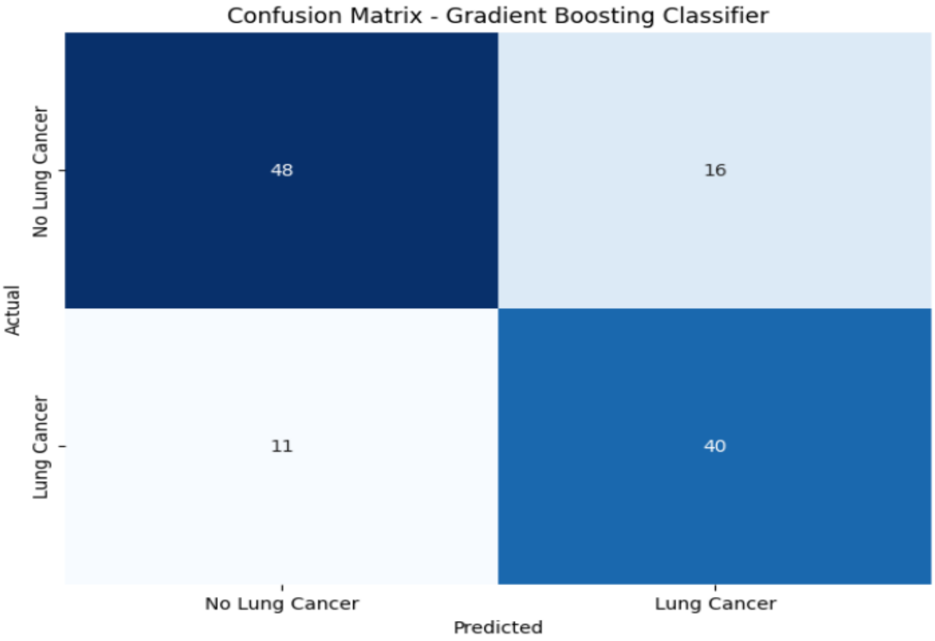
Gradient Boosting Classifier :



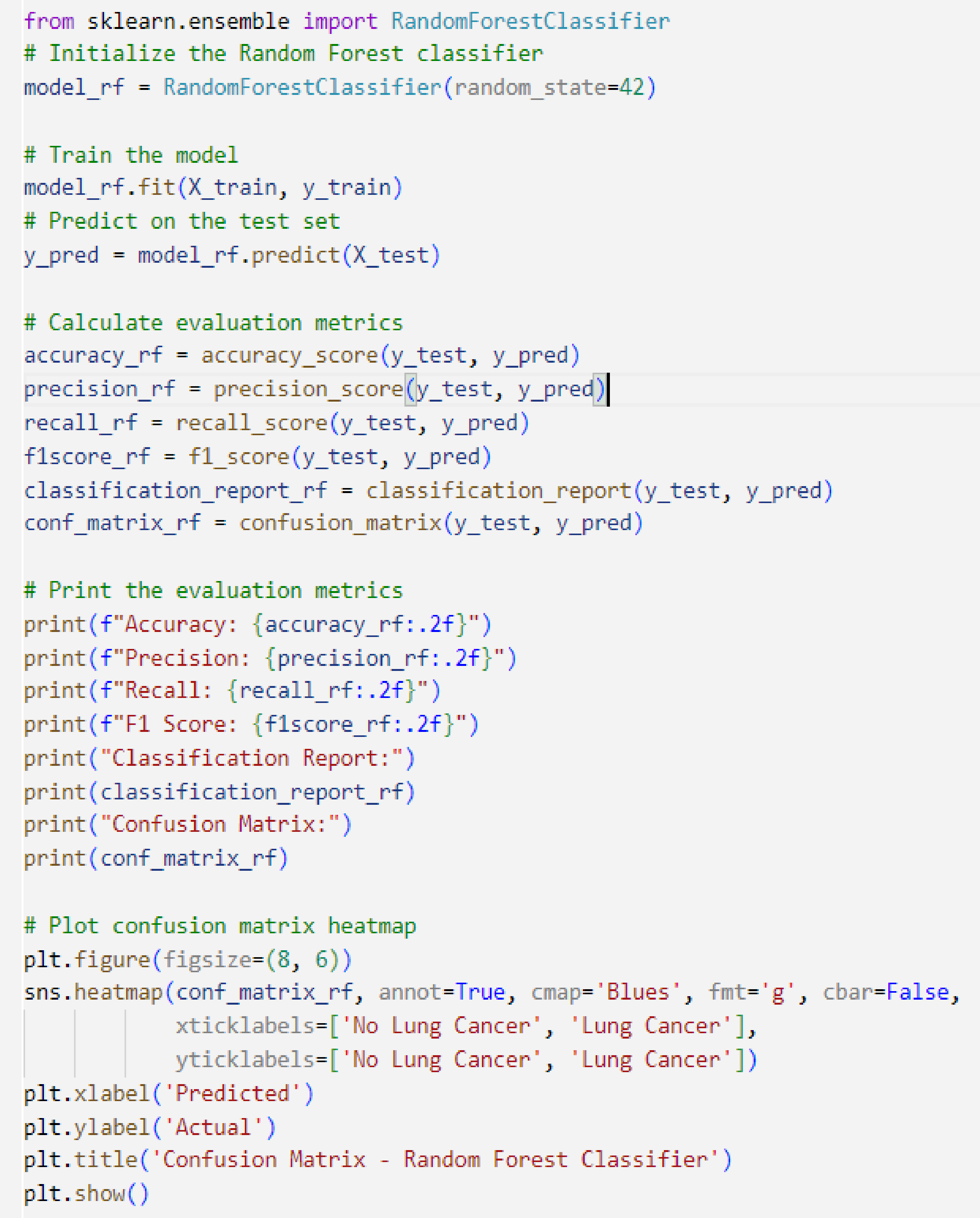
```python
from sklearn.ensemble import GradientBoostingClassifier
# Initialize the Gradient Boosting Classifier
model_gbm = GradientBoostingClassifier(random_state=42)

# Train the model
model_gbm.fit(X_train, y_train)

# Now predict on the test set
y_pred = model_gbm.predict(X_test)

# Calculate evaluation metrics
accuracy_gbm = accuracy_score(y_test, y_pred)
precision_gbm = precision_score(y_test, y_pred)
recall_gbm = recall_score(y_test, y_pred)
f1score_gbm = f1_score(y_test, y_pred)
classification_report_gbm = classification_report(y_test, y_pred)
conf_matrix_gbm = confusion_matrix(y_test, y_pred)

# Print the evaluation metrics
print(f"Accuracy: {accuracy_gbm:.2f}")
print(f"Precision: {precision_gbm:.2f}")
print(f"Recall: {recall_gbm:.2f}")
print(f"F1 Score: {f1score_gbm:.2f}")
print("Classification Report:")
print(classification_report_gbm)
print("Confusion Matrix:")
print(conf_matrix_gbm)

# Plot confusion matrix heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_gbm, annot=True, cmap='Blues', fmt='g', cbar=False,
            xticklabels=['No Lung Cancer', 'Lung Cancer'],
            yticklabels=['No Lung Cancer', 'Lung Cancer'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Gradient Boosting Classifier')
plt.show()
```



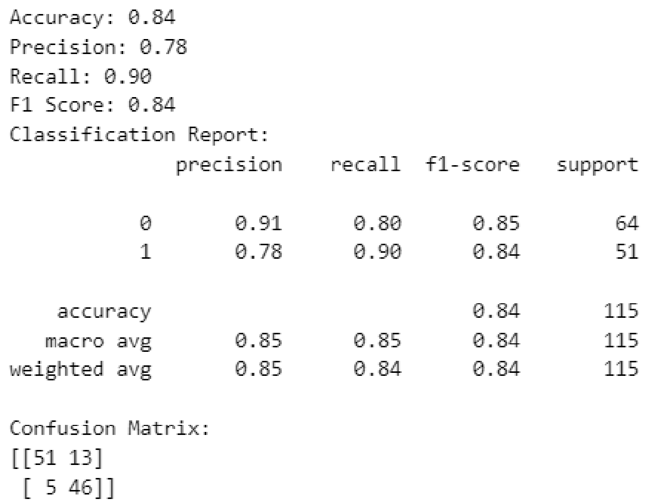
```
Accuracy: 0.77
Precision: 0.71
Recall: 0.78
F1 Score: 0.75
Classification Report:
              precision    recall  f1-score   support

           0       0.81      0.75      0.78        64
           1       0.71      0.78      0.75        51

    accuracy                           0.77       115
   macro avg       0.76      0.77      0.76       115
weighted avg       0.77      0.77      0.77       115

Confusion Matrix:
[[48 16]
 [11 40]]
```



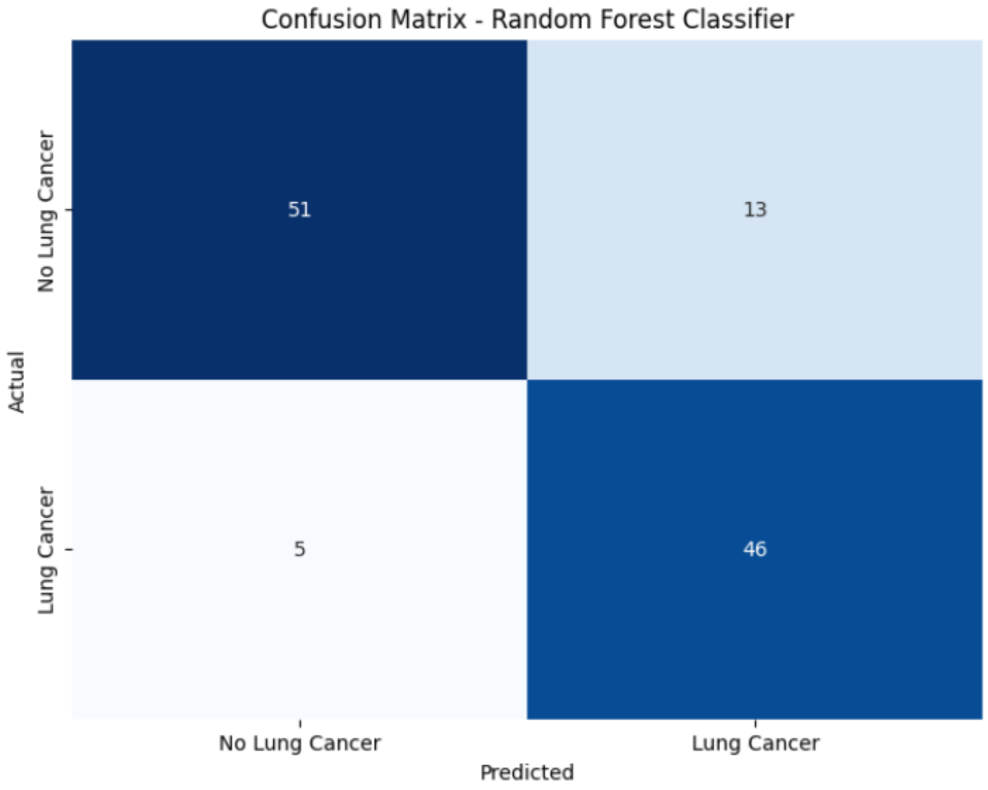
Random Forest Classifier :



```
from sklearn.ensemble import RandomForestClassifier
# Initialize the Random Forest classifier
model_rf = RandomForestClassifier(random_state=42)

# Train the model
model_rf.fit(X_train, y_train)
# Predict on the test set
y_pred = model_rf.predict(X_test)

# Calculate evaluation metrics
accuracy_rf = accuracy_score(y_test, y_pred)
precision_rf = precision_score(y_test, y_pred)
recall_rf = recall_score(y_test, y_pred)
f1score_rf = f1_score(y_test, y_pred)
classification_report_rf = classification_report(y_test, y_pred)
conf_matrix_rf = confusion_matrix(y_test, y_pred)

# Print the evaluation metrics
print(f"Accuracy: {accuracy_rf:.2f}")
print(f"Precision: {precision_rf:.2f}")
print(f"Recall: {recall_rf:.2f}")
print(f"F1 Score: {f1score_rf:.2f}")
print("Classification Report:")
print(classification_report_rf)
print("Confusion Matrix:")
print(conf_matrix_rf)

# Plot confusion matrix heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(conf_matrix_rf, annot=True, cmap='Blues', fmt='g', cbar=False,
            xticklabels=['No Lung Cancer', 'Lung Cancer'],
            yticklabels=['No Lung Cancer', 'Lung Cancer'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix - Random Forest Classifier')
plt.show()
```



```
Accuracy: 0.84
Precision: 0.78
Recall: 0.90
F1 Score: 0.84
Classification Report:
              precision    recall  f1-score   support

           0       0.91      0.80      0.85        64
           1       0.78      0.90      0.84        51

    accuracy                           0.84       115
   macro avg       0.85      0.85      0.84       115
weighted avg       0.85      0.84      0.84       115

Confusion Matrix:
[[51 13]
 [ 5 46]]
```

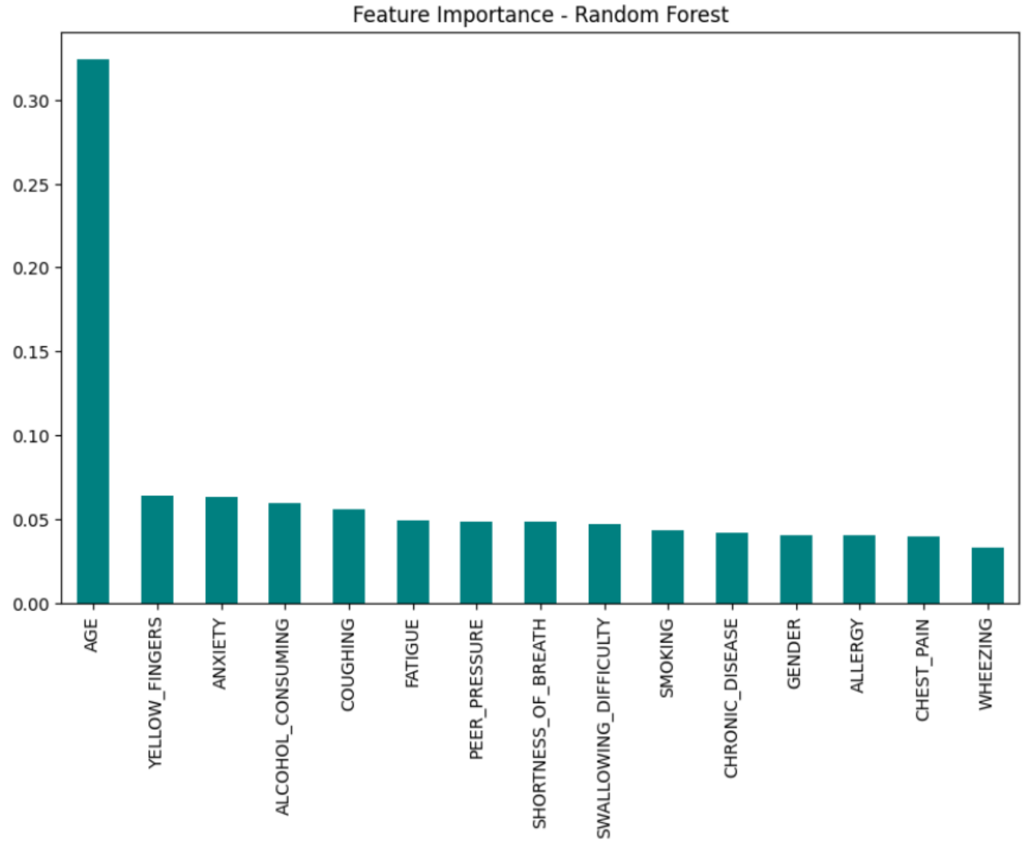


Prediction Observations:

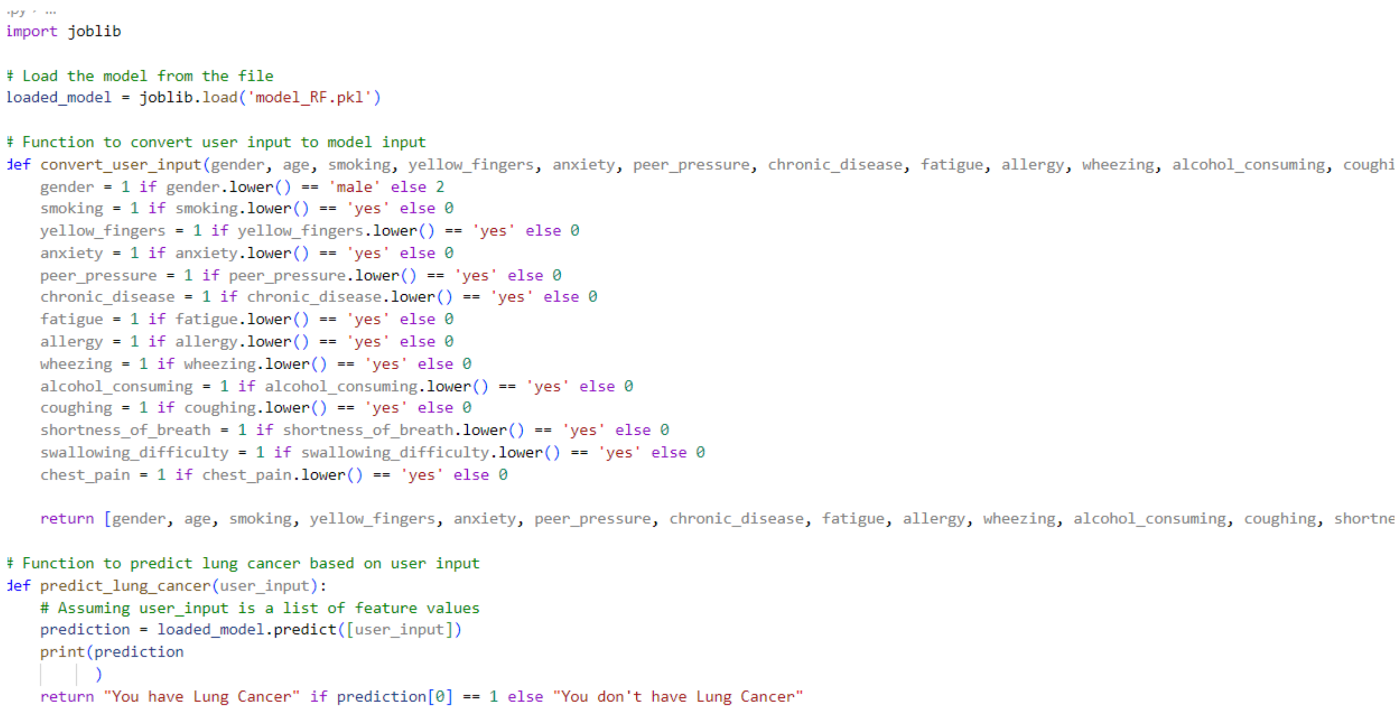
Training on complete dataset-

Algorithm	ACCURACY
XGBoost	76
Gradient Boosting	77
Random Forest	84
Gaussian NB	70

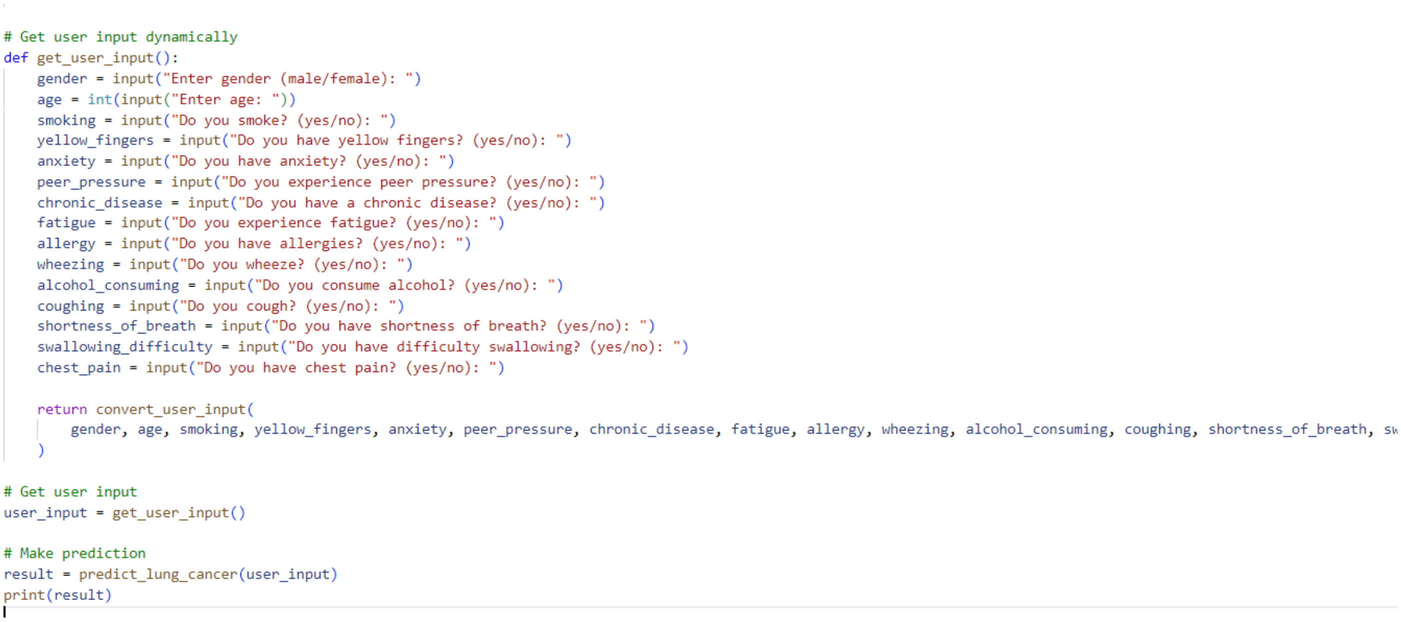
Feature Importance of RANDOM FOREST :



Model Implementation :

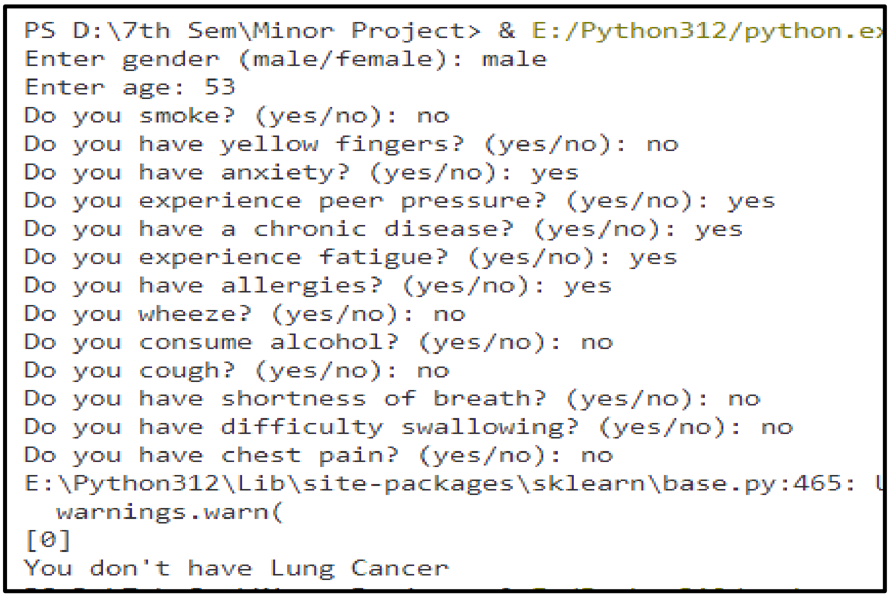


```
.py · ...
import joblib

# Load the model from the file
loaded_model = joblib.load('model_RF.pkl')

# Function to convert user input to model input
def convert_user_input(gender, age, smoking, yellow_fingers, anxiety, peer_pressure, chronic_disease, fatigue, allergy, wheezing, alcohol_consuming, coughi
    gender = 1 if gender.lower() == 'male' else 2
    smoking = 1 if smoking.lower() == 'yes' else 0
    yellow_fingers = 1 if yellow_fingers.lower() == 'yes' else 0
    anxiety = 1 if anxiety.lower() == 'yes' else 0
    peer_pressure = 1 if peer_pressure.lower() == 'yes' else 0
    chronic_disease = 1 if chronic_disease.lower() == 'yes' else 0
    fatigue = 1 if fatigue.lower() == 'yes' else 0
    allergy = 1 if allergy.lower() == 'yes' else 0
    wheezing = 1 if wheezing.lower() == 'yes' else 0
    alcohol_consuming = 1 if alcohol_consuming.lower() == 'yes' else 0
    coughing = 1 if coughing.lower() == 'yes' else 0
    shortness_of_breath = 1 if shortness_of_breath.lower() == 'yes' else 0
    swallowing_difficulty = 1 if swallowing_difficulty.lower() == 'yes' else 0
    chest_pain = 1 if chest_pain.lower() == 'yes' else 0

    return [gender, age, smoking, yellow_fingers, anxiety, peer_pressure, chronic_disease, fatigue, allergy, wheezing, alcohol_consuming, coughing, shortne

# Function to predict lung cancer based on user input
def predict_lung_cancer(user_input):
    # Assuming user_input is a list of feature values
    prediction = loaded_model.predict([user_input])
    print(prediction
        )
    return "You have Lung Cancer" if prediction[0] == 1 else "You don't have Lung Cancer"
```

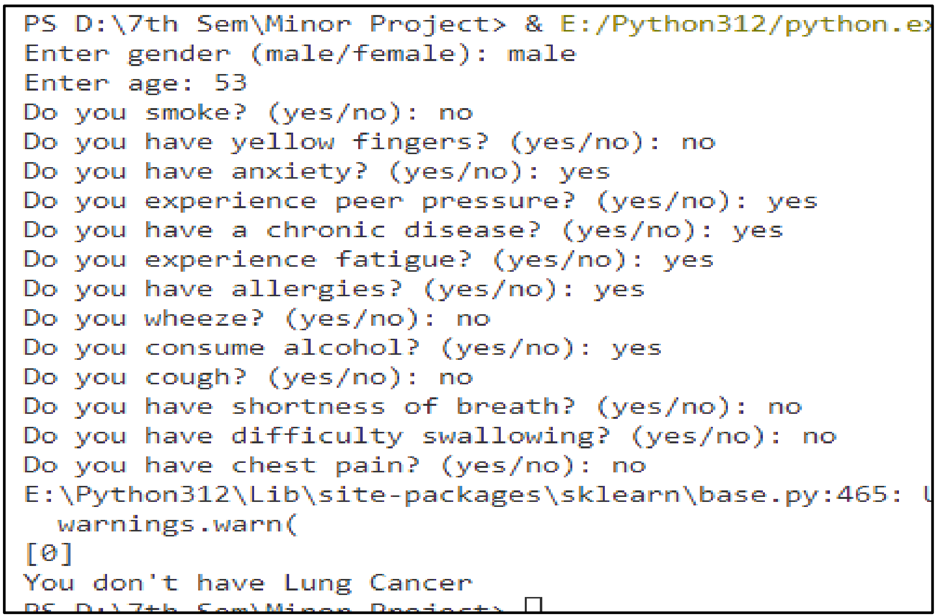


```
# Get user input dynamically
def get_user_input():
    gender = input("Enter gender (male/female): ")
    age = int(input("Enter age: "))
    smoking = input("Do you smoke? (yes/no): ")
    yellow_fingers = input("Do you have yellow fingers? (yes/no): ")
    anxiety = input("Do you have anxiety? (yes/no): ")
    peer_pressure = input("Do you experience peer pressure? (yes/no): ")
    chronic_disease = input("Do you have a chronic disease? (yes/no): ")
    fatigue = input("Do you experience fatigue? (yes/no): ")
    allergy = input("Do you have allergies? (yes/no): ")
    wheezing = input("Do you wheeze? (yes/no): ")
    alcohol_consuming = input("Do you consume alcohol? (yes/no): ")
    coughing = input("Do you cough? (yes/no): ")
    shortness_of_breath = input("Do you have shortness of breath? (yes/no): ")
    swallowing_difficulty = input("Do you have difficulty swallowing? (yes/no): ")
    chest_pain = input("Do you have chest pain? (yes/no): ")

    return convert_user_input(
        gender, age, smoking, yellow_fingers, anxiety, peer_pressure, chronic_disease, fatigue, allergy, wheezing, alcohol_consuming, coughing, shortness_of_breath, sw
    )

# Get user input
user_input = get_user_input()

# Make prediction
result = predict_lung_cancer(user_input)
print(result)
```

OUTPUT :



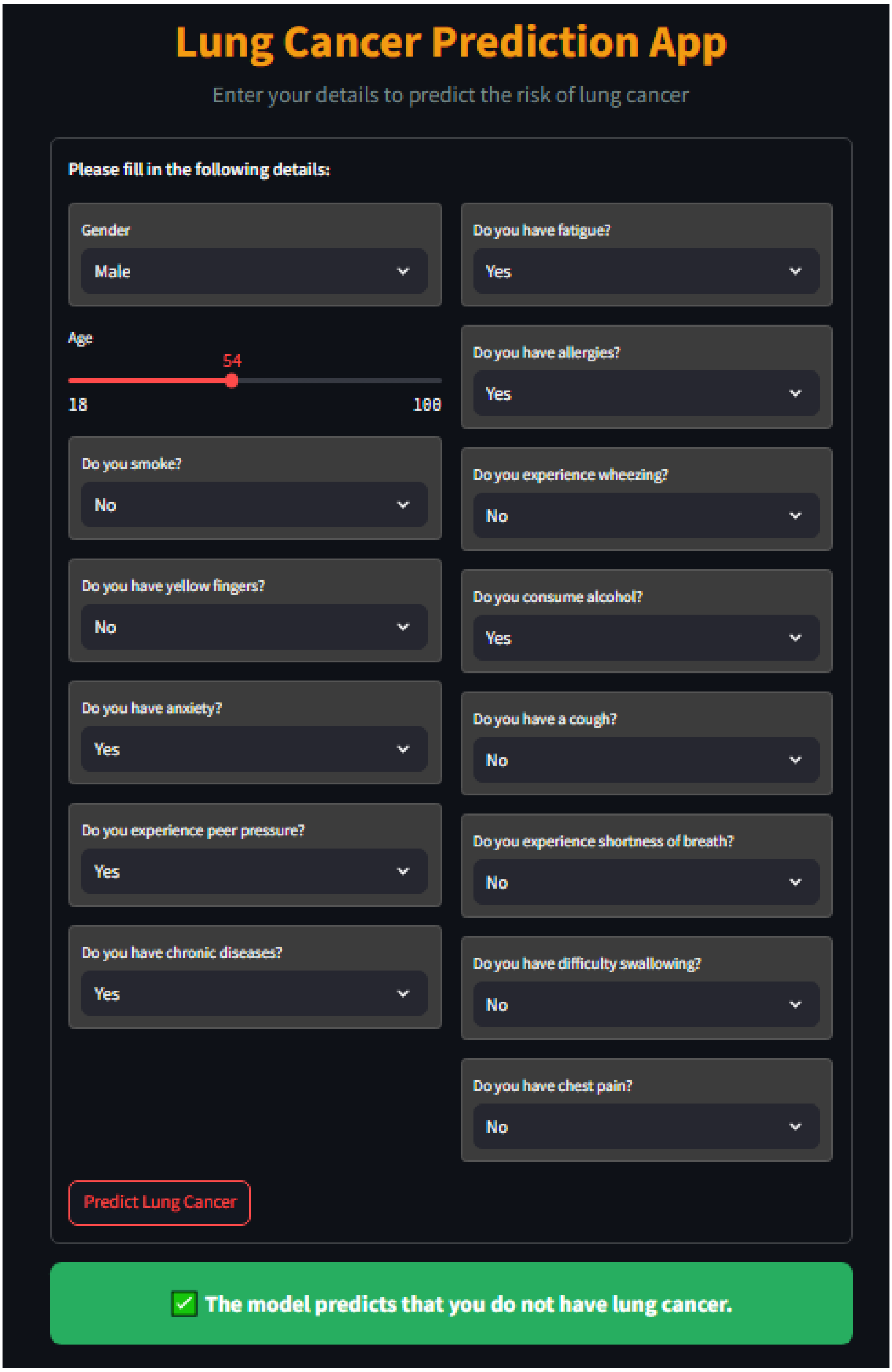
```
PS D:\7th Sem\Minor Project> & E:/Python312/python.ex
Enter gender (male/female): male
Enter age: 53
Do you smoke? (yes/no): no
Do you have yellow fingers? (yes/no): no
Do you have anxiety? (yes/no): yes
Do you experience peer pressure? (yes/no): yes
Do you have a chronic disease? (yes/no): yes
Do you experience fatigue? (yes/no): yes
Do you have allergies? (yes/no): yes
Do you wheeze? (yes/no): no
Do you consume alcohol? (yes/no): no
Do you cough? (yes/no): no
Do you have shortness of breath? (yes/no): no
Do you have difficulty swallowing? (yes/no): no
Do you have chest pain? (yes/no): no
E:\Python312\Lib\site-packages\sklearn\base.py:465: U
  warnings.warn(
[0]
You don't have Lung Cancer
```



```
PS D:\7th Sem\Minor Project> & E:/Python312/python.ex
Enter gender (male/female): male
Enter age: 53
Do you smoke? (yes/no): no
Do you have yellow fingers? (yes/no): no
Do you have anxiety? (yes/no): yes
Do you experience peer pressure? (yes/no): yes
Do you have a chronic disease? (yes/no): yes
Do you experience fatigue? (yes/no): yes
Do you have allergies? (yes/no): yes
Do you wheeze? (yes/no): no
Do you consume alcohol? (yes/no): yes
Do you cough? (yes/no): no
Do you have shortness of breath? (yes/no): no
Do you have difficulty swallowing? (yes/no): no
Do you have chest pain? (yes/no): no
E:\Python312\Lib\site-packages\sklearn\base.py:465: U
  warnings.warn(
[0]
You don't have Lung Cancer
PS D:\7th Sem\Minor Project> 
```

Web App:

The web app is built with **Streamlit**, a powerful framework for developing interactive web applications, ensuring an easy-to-use interface and real-time predictions. The model is trained on the Indian Meteorological Department's (IMD) Yearly Gridded Rainfall dataset, offering a comprehensive analysis of the model's prediction accuracy and statistics for each year.



Lung Cancer Prediction App

Enter your details to predict the risk of lung cancer

Please fill in the following details:

Gender: Male

Age: 54 (18–100)

Do you smoke?: No

Do you have yellow fingers?: No

Do you have anxiety?: Yes

Do you experience peer pressure?: Yes

Do you have chronic diseases?: Yes

Do you have fatigue?: Yes

Do you have allergies?: Yes

Do you experience wheezing?: No

Do you consume alcohol?: Yes

Do you have a cough?: No

Do you experience shortness of breath?: No

Do you have difficulty swallowing?: No

Do you have chest pain?: No

✅ The model predicts that you do not have lung cancer.

Future Enhancements

To further enhance the **Lung Cancer Detection using Machine Learning** project and improve its diagnostic capabilities, several key improvements are planned for future development:

1. **Real-Time Data Integration:** Integrating real-time data from medical databases or health monitoring devices will allow the model to provide up-to-date predictions. This will enable more accurate and timely cancer detection, aiding in early diagnosis and better treatment planning.
2. **Advanced Model Tuning:** Further hyperparameter tuning and the exploration of advanced machine learning techniques, such as **ensemble learning** (e.g., **Random Forests**, **Gradient Boosting Machines**) and **deep learning** (e.g., **Convolutional Neural Networks**, **Recurrent Neural Networks**), will be employed to improve prediction accuracy and the robustness of the model.
3. **Incorporation of Additional Clinical Factors:** Including more clinical and genetic factors (e.g., **family history**, **genetic markers**, **previous medical conditions**) in the model will offer a more comprehensive and personalized prediction, potentially improving the accuracy of lung cancer detection.
4. **Multimodal Data Integration:** Developing models that combine data from various sources such as **X-rays**, **CT scans**, **biopsy reports**, and **patient history** will enhance the precision of diagnosis. This approach would address different stages and types of lung cancer, improving the model's adaptability.
5. **Visualization Enhancements:** Implementing interactive visualizations like **heatmaps**, **decision trees**, and **ROC curves** will aid healthcare professionals in interpreting model predictions. These visual enhancements can improve the decision-making process, making the predictions more interpretable and actionable.
6. **Mobile and Web Interface:** Developing a user-friendly **mobile and web interface** will make the lung cancer prediction tool accessible to a wider range of users, including healthcare providers, patients, and researchers. This will increase the model's usability and impact, enabling users to access predictions and guidance easily.

These enhancements aim to make the **Lung Cancer Detection** model more **accurate**, **robust**, and **accessible**, contributing to the advancement of **early cancer diagnosis** and **personalized healthcare**.

Conclusion

In conclusion, the **Lung Cancer Detection using Machine Learning** project represents a significant advancement in applying machine learning techniques for early cancer detection. By utilizing clinical and lifestyle data, the project establishes a strong foundation for predicting the likelihood of lung cancer. The processes implemented, including **data preprocessing**, **exploratory data analysis**, and **model selection**, have contributed to building an initial predictive model.

While the current implementation focuses on machine learning techniques such as **Logistic Regression**, **Random Forest**, and **Gradient Boosting**, the project is poised for further enhancement. Planned improvements, such as incorporating **real-time patient data**, refining the model with **additional clinical factors** (e.g., genetic information), and improving **visualization techniques**, will enhance the model's **accuracy** and **usability**.

As the project progresses, these future enhancements will ensure that the **lung cancer detection** model becomes a valuable tool for healthcare professionals, offering **early diagnosis** and **personalized treatment plans**. The ultimate goal is to provide **accurate**, **real-time insights** into lung cancer risk, contributing to better-informed decisions in **cancer prevention** and **treatment**.

References:

1. Pandas Documentation: https://pandas.pydata.org/docs/
2. NumPy Documentation: https://numpy.org/doc/
3. Scikit-learn Documentation: https://scikit-learn.org/stable/
4. Matplotlib Documentation: https://matplotlib.org/stable/contents.html
5. https://www.cancer.org/cancer/types/lung-cancer/about/key-statistics.html): https://www.americanSociety welfare.gov.in/
6. Seaborn Documentation: https://seaborn.pydata.org/