ONE YEAR LIFE EXPECTANCY POST THORACIC SURGERY USING MACHINE LEARNING

AN INDUSTRY ORIENTED MINI REPORT

Submitted to

JAWAHARLAL NEHRU TECNOLOGICAL UNIVERSITY, HYDERABAD

In partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING(AI&ML)

Submitted By

VENISHETTY SHRAVYA	**21UK1A6620**
ANNARAPU SRICHAITHANYA	**21UK1A6650**
ERELLI SHIVA KUMAR	**22UK5A6604**
MYDAM VAMSHI	**21UK1A6635**

Under the guidance of

Mr. T. Sanath kumar

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

VAAGDEVI ENGINEERING COLLEGE

Affiliated to JNTUH, HYDERABAD

BOLLIKUNTA, WARANGAL (T.S) – 506005

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING(AI&ML)

VAAGDEVI ENGINEERING COLLEGE(WARANGAL)

CERTIFICATE OF COMPLETION
INDUSTRY ORIENTED MINI PROJECT

This is to certify that the UG Project Phase-1 entitled "ONE YEAR LIFE EXPECTANCY POST THORACIC SURGERY USING MACHINE LEARNING" is being submitted by VENISHETTY SHRAVYA (21UK1A6620), ANNARAPU SRICHAITHANYA (21UK1A6650), ERELLI SHIVA KUMAR (22UK5A6604), MYDAM VAMSHI (21UK1A6635) in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Computer Science & Engineering to Jawaharlal Nehru Technological University Hyderabad during the academic year 2024-2025.

Project Guide	**HOD**
Mr.T.Sanath kumar	**Dr. K. Sharmila**
(Assistant Professor)	(Professor)

External

ACKNOWLEDGEMENT

We wish to take this opportunity to express our sincere gratitude and deep sense of respect to our beloved **Dr.P.PRASAD RAO,** Principal, Vaagdevi Engineering College for making us available all the required assistance and for his support and inspiration to carry out this UG Project Phase-1 in the institute.

We extend our heartfelt thanks to **Dr.K.SHARMILA**, Head of the Department of CSE, Vaagdevi Engineering College for providing us necessary infrastructure and thereby giving us freedom to carry out the UG Project Phase-1.

We express heartfelt thanks to Smart Bridge Educational Services Private Limited, for their constant supervision as well as for providing necessary information regarding the UG Project Phase-1 and for their support in completing the UG Project Phase-1.

We express heartfelt thanks to the guide, **T. SANATH KUMAR ,** Assistant professor, Department of CSE for his constant support and giving necessary guidance for completion of this UG Project Phase-1.

Finally, we express our sincere thanks and gratitude to my family members, friends for their encouragement and outpouring their knowledge and experience throughout the thesis.

VENISHETTY SHRAVYA	**(21UK1A6620)**
ANNARAPU SRICHAITHANYA	**(21UK1A6650)**
ERELLI SHIVA KUMAR	**(22UK5A6604)**
MYDAM VAMSHI	**(21UK1A6635)**

ABSTRACT

The scope of this project is to propose a life expectancy rate and examine the mortality after thoracic surgery which takes into account the different importance of various features which can have an effect in the end result. The data of the patients collected after diagnosis have been used as the dataset. Various metrics which affect the result have been analyzed with the help of random forest and decision tree algorithms to better understand the consequences of post- surgery. Particular metrics have been selected according to their weightage on the main outcome for prediction. Thus, it enables us to have better comprehension with various algorithms and also some important parameters are selected for the construction of a better model. In addition, we have several classification features such as presence of pain before surgery, haemoptysis before surgery, cough before surgery, whether the patient is a smoker, whether the patient has asthma, and a few others. This classification model predicts whether the patient will survive for a year-long period or not with better selection of the data features.

Keywords: Thoracic Surgery, Machine Learning, Random Forest, Decision Trees.

TABLE OF CONTENTS :-

1.INTRODUCTION

1.1. OVERVIEW

Predicting one-year life expectancy after thoracic surgery is crucial for patient well-being and clinical decision-making. Machine learning (ML) offers advanced capabilities to analyze complex datasets and provide accurate predictions, aiding in better management and outcomes for patients undergoing thoracic surgery. This overview explores the ML approach to predicting oneyear survival, identifying key influencing factors, and emphasizing the importance of these predictions in clinical practice.

Thoracic surgery involves procedures on the lungs, esophagus, and chest, which come with significant risks. Accurate predictions of post-surgery survival rates help in tailoring patient care and making informed surgical decisions. The ML model leverages a dataset comprising various patient attributes and health indicators to make these predictions. Key features in the dataset include diagnosis type, lung function tests (FVC and FEV1), performance status, symptoms (pain, haemoptysis, dyspnoea, cough, weakness), tumor size, and presence of comorbidities such as diabetes, recent myocardial infarction, peripheral arterial disease, smoking, and asthma. The target variable is the patient's one-year death outcome.

Machine learning algorithms can process these diverse and complex data points, identifying patterns and correlations that may not be immediately evident through traditional statistical methods. By training models on historical patient data, ML can predict the likelihood of a patient surviving one year postsurgery, providing valuable insights for clinicians.

1.2. PURPOSE

The purpose of predicting one-year life expectancy post thoracic surgery using machine learning serves several critical roles in enhancing patient care and optimizing clinical outcomes. Specifically, its purposes include:

1. **Informing Clinicians and Patients:**

- The model provides healthcare professionals and patients with easily understandable and actionable information about post-surgery survival probabilities. It aids in making informed decisions regarding surgical procedures and postoperative care plans, ultimately ensuring better preparedness and more tailored interventions.

2. **Health Protection:**
    - A primary objective is to protect patient health by identifying individuals at higher risk of poor outcomes after thoracic surgery. By understanding these risks, clinicians can take preemptive measures to mitigate potential complications, thereby improving patient survival rates and overall quality of care.
3. **Clinical Decision Support:**
    - The model assists surgeons and medical teams in making evidence-based decisions. By integrating predictive insights into the clinical workflow, it helps determine the suitability of patients for surgery and guides the customization of perioperative and postoperative management strategies.ok
4. **Optimizing Resource Allocation:**
    - The prediction model helps healthcare institutions allocate resources more effectively. By identifying patients who require intensive monitoring and specialized care, it ensures that resources such as ICU beds, nursing care, and postoperative support are optimally used, thereby improving hospital efficiency and patient outcomes.
5. **Enhancing Research and Policy:**
    - The insights derived from the model contribute to medical research by highlighting key factors influencing post-surgical survival. This knowledge can inform future studies and healthcare policies aimed at improving surgical practices and patient management protocols.
6. **Economic Impact:**
    - Understanding and predicting patient outcomes helps in evaluating the economic impact of thoracic surgeries. By potentially reducing readmissions, complications, and length of hospital stays, the model can contribute to lowering healthcare costs and improving the economic efficiency of healthcare delivery.
7. **Improving Patient Communication and Satisfaction:**
    - Providing clear, data-driven prognostic information enhances communication between healthcare providers and patients. It helps in setting realistic expectations, thereby improving patient satisfaction and adherence to postoperative care plans.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

The existing problem of predicting one-year life expectancy after thoracic surgery presents a multifaceted challenge that spans clinical, patient care, and economic domains.

- **Clinical Challenges:**
  - **High Risk and Uncertainty:** Thoracic surgeries, involving critical procedures on the lungs, esophagus, and chest, carry significant risks, including complications and mortality. Clinicians often face uncertainty in predicting patient outcomes, which can hinder optimal surgical planning and postoperative care. o **Complex Patient Profiles:** Patients undergoing thoracic surgery often present with complex medical histories, including multiple comorbidities such as diabetes, cardiovascular diseases, and respiratory conditions. This complexity makes accurate risk assessment and outcome prediction difficult.
- **Patient Care Issues:**
  - **Informed Decision-Making:** Without accurate predictions of post-surgery outcomes, patients and their families may struggle to make informed decisions about undergoing surgery and preparing for potential recovery scenarios. This can lead to increased anxiety and uncertainty. o **Tailored Treatment Plans:** The lack of precise predictive tools hampers the ability to develop personalized care plans, potentially resulting in suboptimal treatment and follow-up strategies.
- **Economic Impact:**
  - **Healthcare Costs:** Unpredictable outcomes can lead to higher healthcare costs due to prolonged hospital stays, readmissions, and the need for intensive care and additional treatments. This places a financial burden on both healthcare systems and patients. o **Resource Allocation:** Inefficient allocation of medical resources, such as ICU beds and specialized care, arises from the inability to accurately forecast patient needs. This can strain hospital capacities and impact overall healthcare delivery efficiency.
- **Technological and Data Limitations:**
  - **Inadequate Predictive Models:** Current predictive models may not fully capture the nuances of individual patient profiles and the complex interplay

of various health indicators, leading to less reliable predictions. ○ **Data Integration Issues:** Integrating diverse patient data from multiple sources, such as electronic health records (EHRs), medical imaging, and clinical assessments, remains a significant challenge. Incomplete or inconsistent data further complicates the development of robust predictive models.

- **Regulatory and Policy Constraints:**
  - **Lack of Standardization:** There is a lack of standardized protocols and guidelines for utilizing predictive models in clinical practice, which can hinder their adoption and effectiveness.
  - **Ethical and Privacy Concerns:** Ensuring patient privacy and ethical use of predictive models in healthcare requires robust regulatory frameworks, which are often lacking or insufficiently enforced.

2.2 PROPOSED SOLLUTION

To address the multifaceted challenges of predicting one-year life expectancy after thoracic surgery, the following solutions are proposed:

1. **Development of Advanced Machine Learning Models:**

   ○ **Algorithm Selection and Optimization:** Utilize state-of-the-art machine learning algorithms, such as ensemble methods (e.g., Random Forest, Gradient Boosting), neural networks, and support vector machines (SVM), to build robust predictive models. Employ techniques like hyperparameter tuning and cross-validation to optimize model performance. ○ **Feature Engineering:** Conduct thorough feature engineering to identify and extract the most relevant patient attributes and health indicators. This includes incorporating both clinical and demographic variables, such as age, gender, comorbidities, lung function tests, performance status, and tumor characteristics.

2. **Data Integration and Preprocessing:**

   - **Comprehensive Data Collection:** Integrate data from various sources, including electronic health records (EHRs), clinical assessments, and imaging data. Ensure that the dataset is comprehensive, accurate, and uptodate.

- **Data Cleaning and Normalization:** Implement rigorous data preprocessing steps to handle missing values, outliers, and inconsistencies. Normalize and standardize numerical features to ensure uniformity across the dataset.

3. **Enhancing Model Interpretability:**

   - **Explainable AI (XAI):** Incorporate explainability techniques to make the model's predictions transparent and understandable to clinicians. Methods such as SHAP (SHapley Additive exPlanations) values and LIME (Local Interpretable Model-agnostic Explanations) can provide insights into the contribution of each feature to the predicted outcome.
   - **Feature Importance Analysis:** Conduct feature importance analysis to identify the most influential factors impacting patient survival. This helps in understanding the key determinants of post-surgery outcomes and tailoring patient care accordingly.

4. **Clinical Integration and Decision Support:**
   - **Real-time Predictive Analytics:** Develop tools and dashboards that integrate the predictive model into the clinical workflow, providing real-time insights and recommendations to healthcare providers during patient consultations and surgical planning. o **Personalized Treatment Plans:** Use the model's predictions to create personalized treatment and follow-up plans for each patient, addressing their specific risks and needs to improve postoperative outcomes.

5. **Enhancing Patient Communication:**

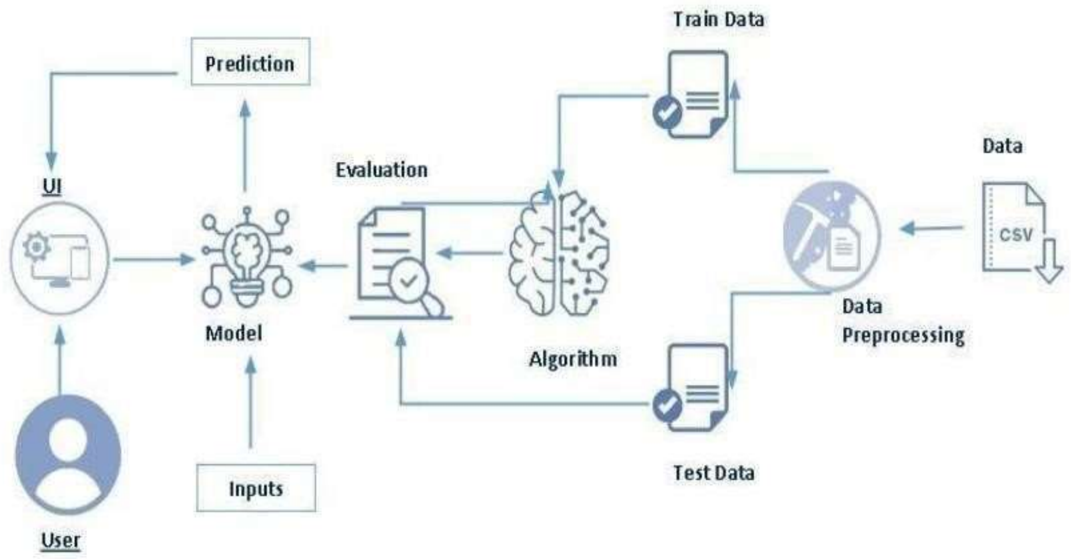
   - **Prognostic Information Sharing:** Develop user-friendly interfaces and communication strategies to share predictive insights with patients and their families. Clear, evidence-based prognostic information helps in setting realistic expectations and enhancing shared decision-making. o **Educational Resources:** Provide educational materials to patients about the factors influencing their surgery outcomes and the importance of adhering to postoperative care recommendations.

6. **Policy and Regulatory Support:**

   - **Standardization and Guidelines:** Work with regulatory bodies to develop standardized protocols for the use of predictive models in clinical practice.

3.THEORITICAL ANALYSIS

3.1. BLOCK DIAGRAM



3.2. SOFTWARE DESIGNING

The following is the software required to complete the thoracic surgery predictive modeling project:

➢ **Jupyter Notebook:**

- Jupyter Notebook will serve as the development and execution environment for predictive modeling, data preprocessing, and model training tasks. It provides an interactive environment with access to Python libraries and supports reproducible research.

➢ **Dataset (CSV File):**

- The dataset in CSV format is essential for training and testing the predictive model. It should include patient attributes and health indicators relevant to thoracic surgery outcomes, such as diagnosis, lung function tests, performance status, symptoms, tumor size, and comorbidities.

➢ **Data Preprocessing Tools:**

- Python libraries like NumPy, Pandas, and Scikit-learn will be used to preprocess the dataset. This includes handling missing data, feature scaling, data cleaning, and transforming data to make it suitable for model training.

➢ **Feature Selection/Drop:**

- Feature selection or dropping unnecessary features from the dataset can be done using Scikit-learn or custom Python code to enhance the model's efficiency. This step ensures that only relevant features are used for prediction, improving model performance.

➢ **Model Training Tools:**

- Machine learning libraries such as Scikit-learn, TensorFlow, or PyTorch will be used to develop, train, and fine-tune the predictive model. Classification models, such as logistic regression, decision trees, or neural networks, can be considered to predict one-year life expectancy after thoracic surgery.

➢ **Model Accuracy Evaluation:**

- After model training, accuracy and performance evaluation tools, such as Scikitlearn metrics or custom validation scripts, will assess the model's predictive capabilities. Metrics like accuracy, precision, recall, F1 score, and ROC-AUC will be used to measure the model's ability to predict one-year survival based on patient data.

➢ **UI Based on Flask Environment:**

- Flask, a Python web framework, will be used to develop the user interface (UI) for the system. The Flask application will provide a user-friendly platform for clinicians and patients to input patient data, view predictions, and access associated health information and recommendations.

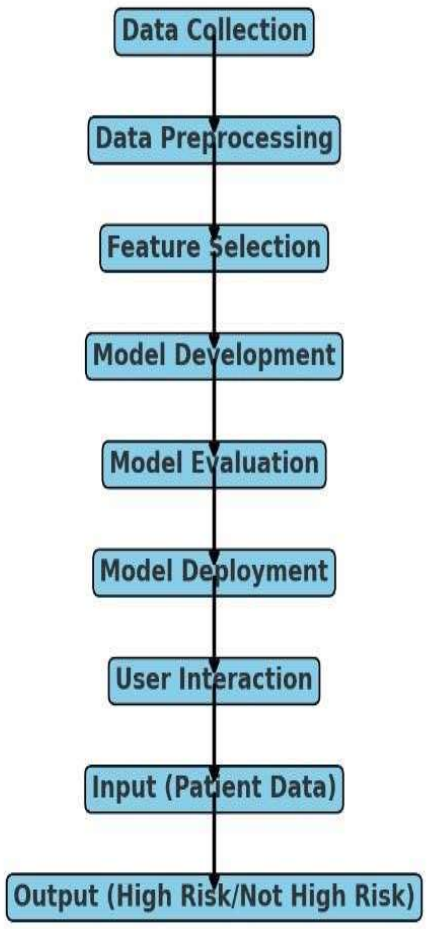
Jupyter Notebook will be the central hub for model development and training, while Flask will facilitate user interaction and data presentation. The dataset, along with data preprocessing, will ensure the quality of the training data, and feature selection will optimize the model. Finally, model accuracy evaluation will confirm the system's predictive capabilities, allowing users to rely on the predictions.

4.EXPERIMENTAL INVESTIGATION

In this project, we have used a thoracic surgery dataset. This dataset is a CSV file consisting of labelled data and has the following columns:

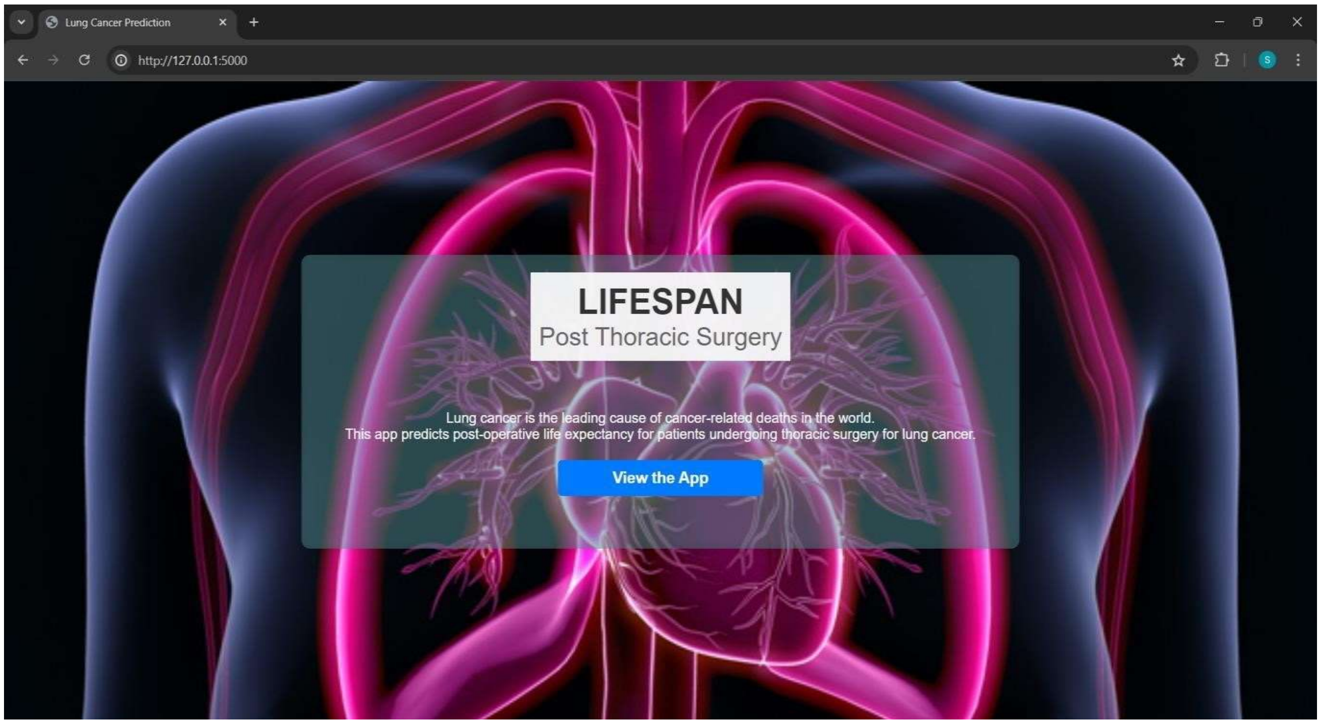
1. **Diagnosis**: The type of diagnosis related to thoracic surgery.
2. **FVC**: Forced Vital Capacity, a lung function test.
3. **FEV1**: Forced Expiratory Volume in one second, another lung function test.
4. **Performance**: The performance status of the patient.
5. **Pain**: The presence or absence of pain.
6. **Haemoptysis**: The presence or absence of coughing up blood.
7. **Dyspnoea**: The presence or absence of difficulty in breathing.
8. **Cough**: The presence or absence of coughing.
9. **Weakness**: The presence or absence of weakness.
10. **Tumor Size**: The size of the tumor.
11. **Diabetes Mellitus**: The presence or absence of diabetes mellitus.
12. **MI_6mo**: The presence or absence of a myocardial infarction within the past 6 months.
13. **PAD**: The presence or absence of peripheral arterial disease.
14. **Smoking**: The presence or absence of smoking history.
15. **Asthma**: The presence or absence of asthma.
16. **Age**: The age of the patient.
17. **Death_1yr**: The target variable indicating whether the patient died within one year post-surgery.

5. FLOW CHART


Data Collection
Data Preprocessing
Feature Selection
Model Development
Model Evaluation
Model Deployment
User Interaction
Input (Patient Data)
Output (High Risk/Not High Risk)

6.RESULT

HOME PAGE

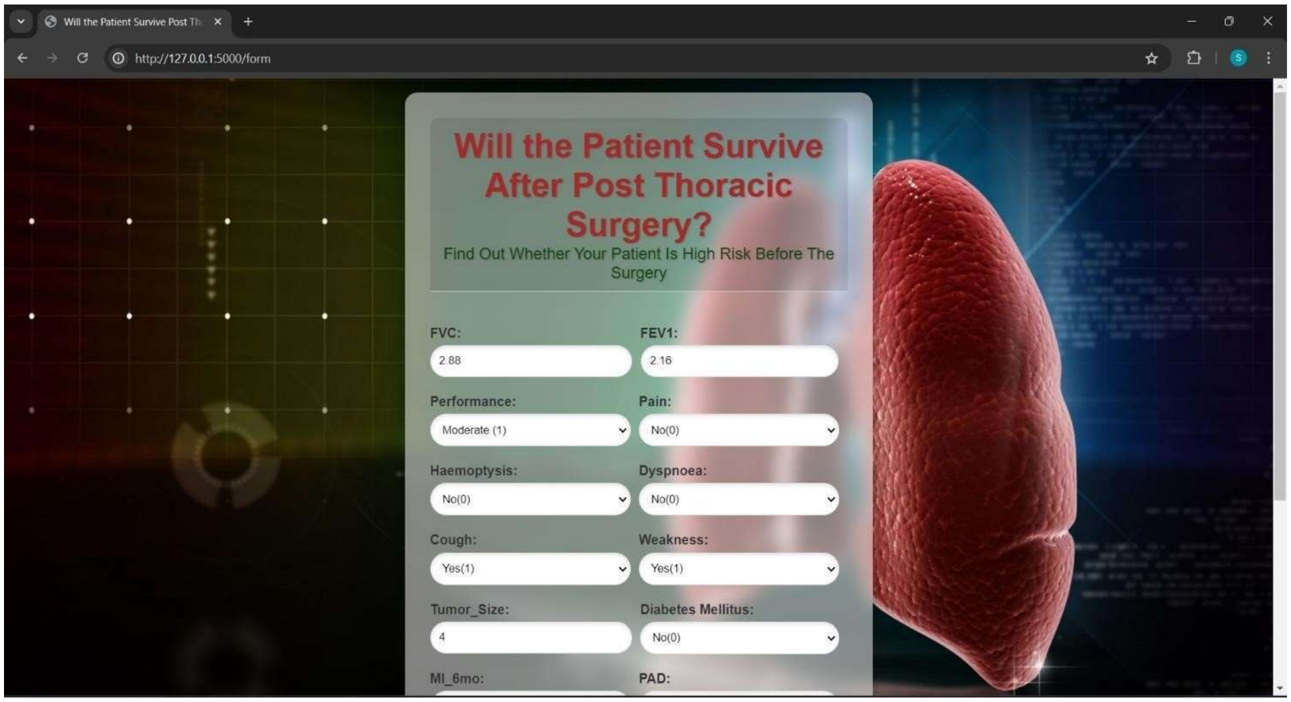


LIFESPAN

Post Thoracic Surgery

Lung cancer is the leading cause of cancer-related deaths in the world.
This app predicts post-operative life expectancy for patients undergoing thoracic surgery for lung cancer.

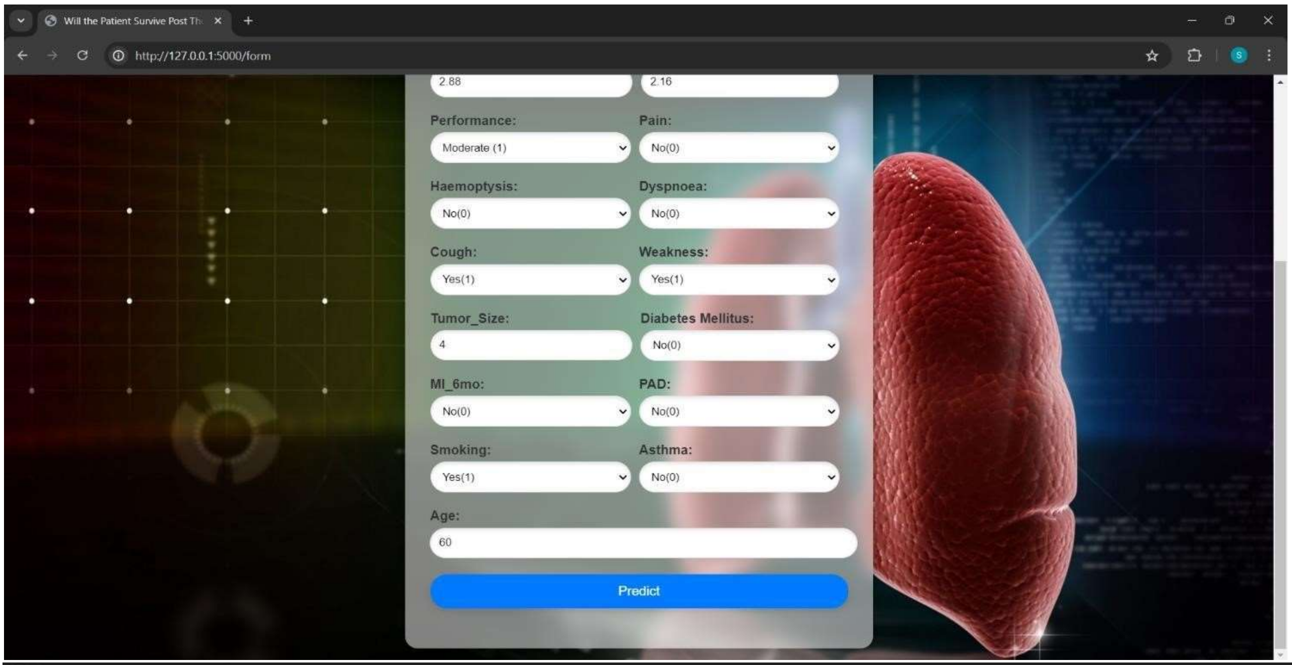
PREDICTIONS



Will the Patient Survive After Post Thoracic Surgery?

Find Out Whether Your Patient Is High Risk Before The Surgery

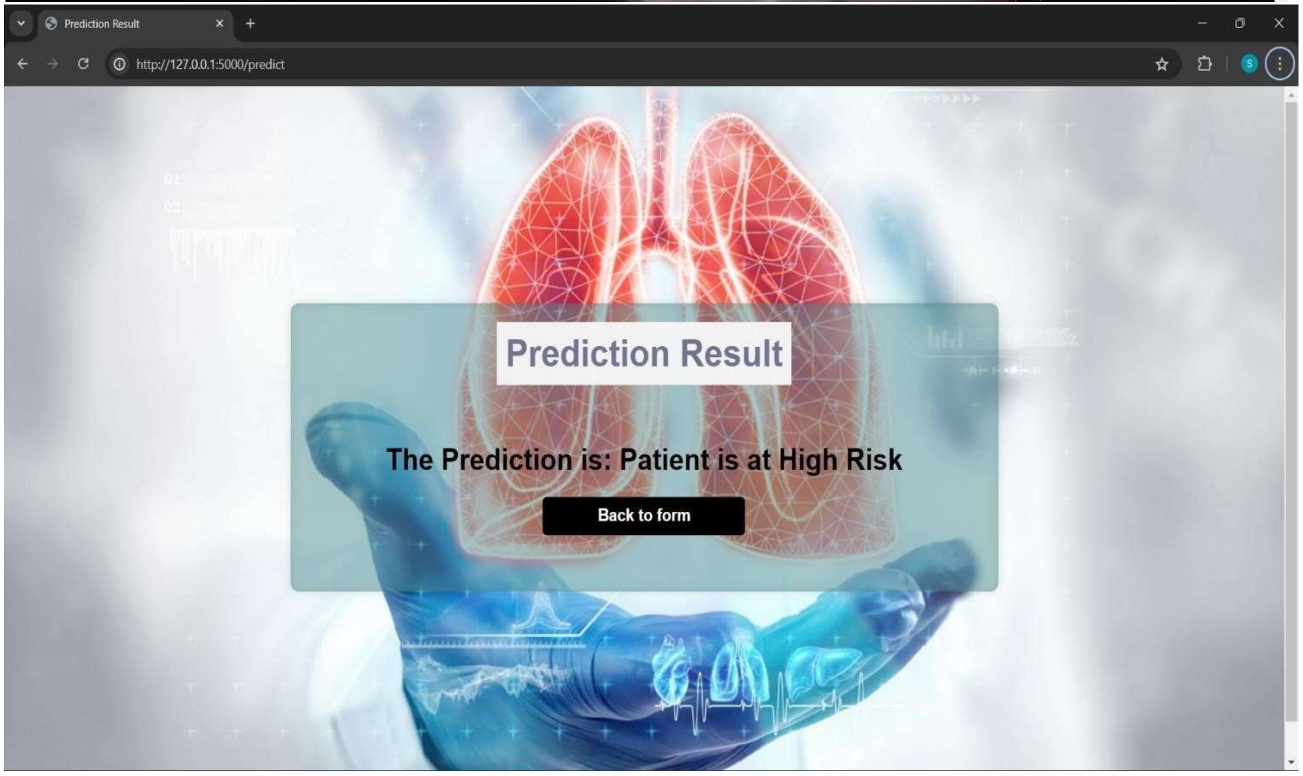
FVC: 2.88
FEV1: 2.16
Performance: Moderate (1)
Pain: No(0)
Haemoptysis: No(0)
Dyspnoea: No(0)
Cough: Yes(1)
Weakness: Yes(1)
Tumor_Size: 4
Diabetes Mellitus: No(0)
MI_6mo:
PAD:



Will the Patient Survive Post Th

http://127.0.0.1:5000/form

2.88	2.16
Performance: Moderate (1)	Pain: No(0)
Haemoptysis: No(0)	Dyspnoea: No(0)
Cough: Yes(1)	Weakness: Yes(1)
Tumor_Size: 4	Diabetes Mellitus: No(0)
MI_6mo: No(0)	PAD: No(0)
Smoking: Yes(1)	Asthma: No(0)
Age: 60	

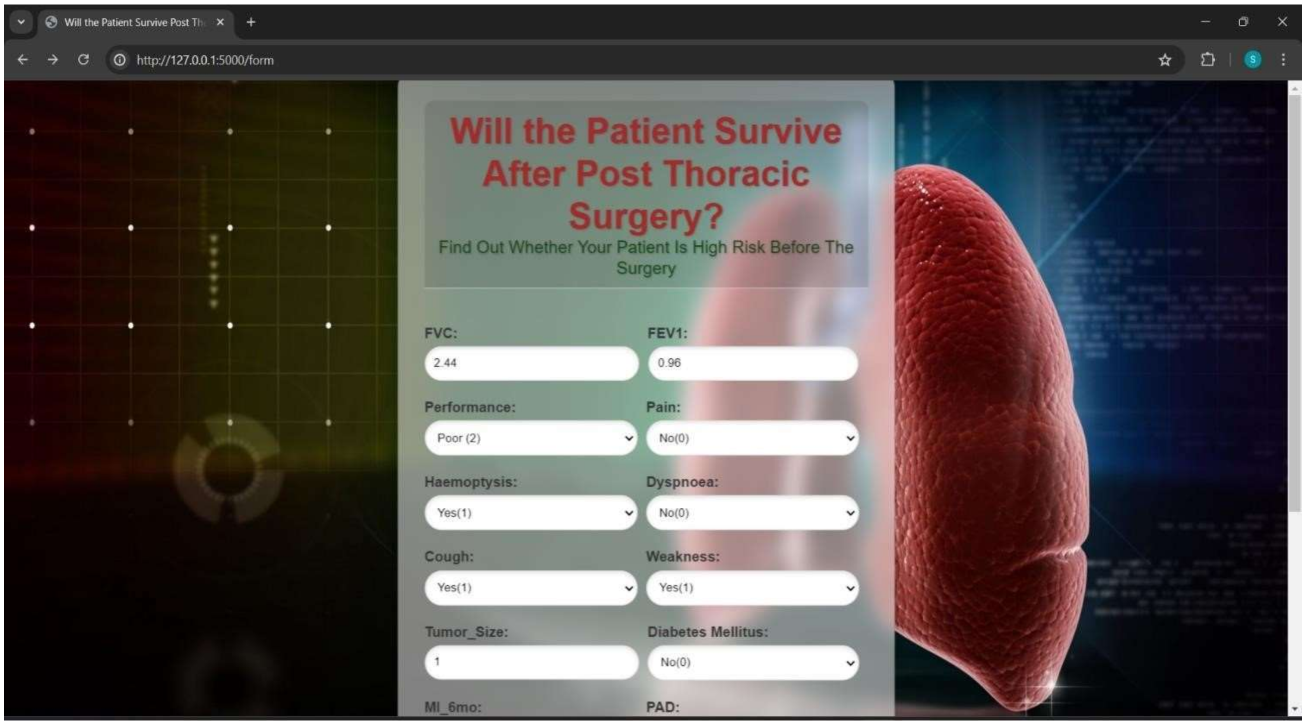


Prediction Result

http://127.0.0.1:5000/predict

Prediction Result

The Prediction is: Patient is at High Risk

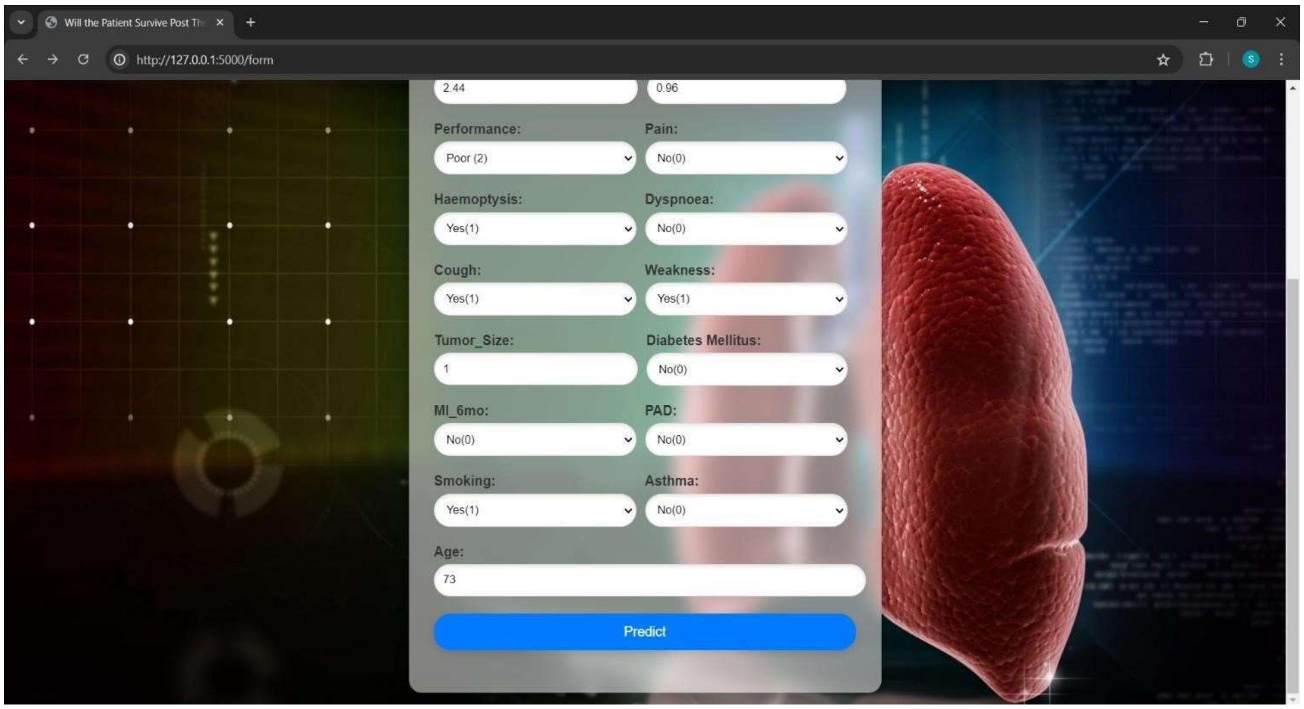

Will the Patient Survive Post Th

http://127.0.0.1:5000/form

Will the Patient Survive After Post Thoracic Surgery?

Find Out Whether Your Patient Is High Risk Before The Surgery

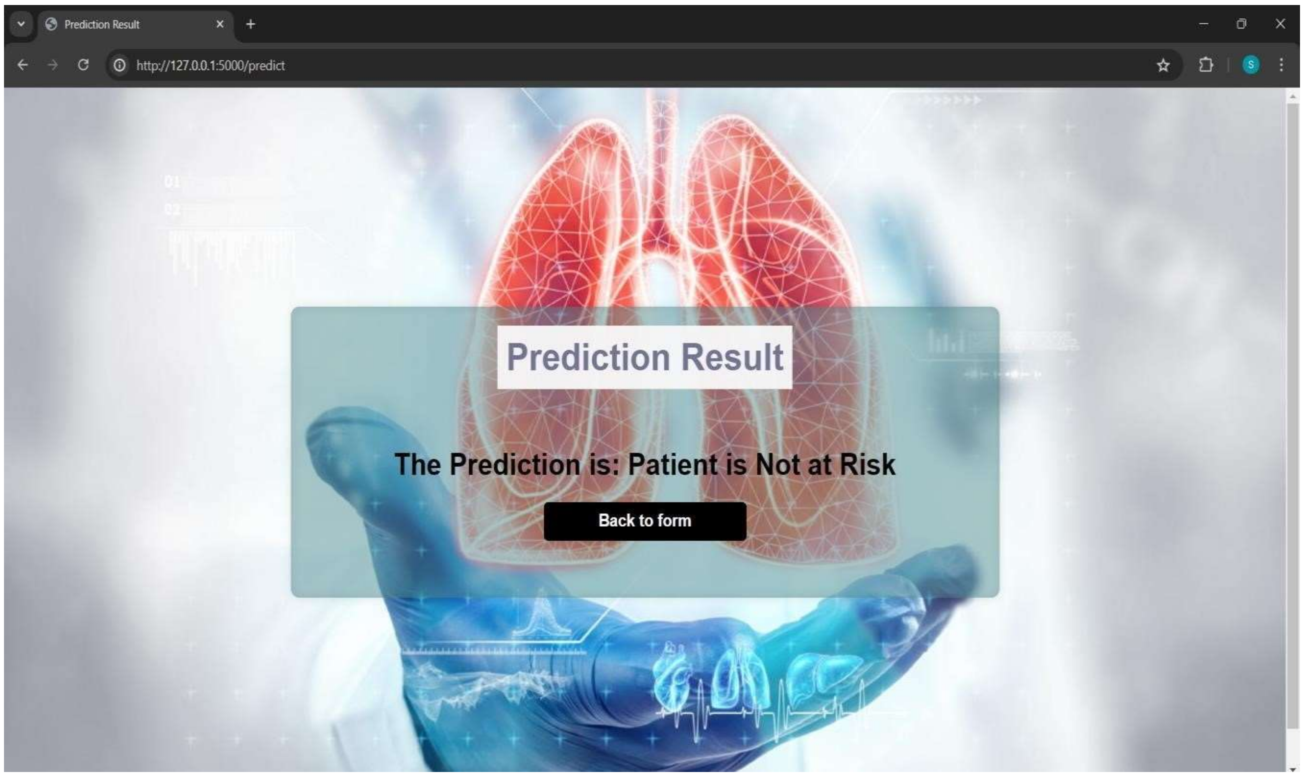
FVC: 2.44
FEV1: 0.96
Performance: Poor (2)
Pain: No(0)
Haemoptysis: Yes(1)
Dyspnoea: No(0)
Cough: Yes(1)
Weakness: Yes(1)
Tumor_Size: 1
Diabetes Mellitus: No(0)
MI_6mo:
PAD:


Will the Patient Survive Post Th

http://127.0.0.1:5000/form

2.44
0.96
Performance: Poor (2)
Pain: No(0)
Haemoptysis: Yes(1)
Dyspnoea: No(0)
Cough: Yes(1)
Weakness: Yes(1)
Tumor_Size: 1
Diabetes Mellitus: No(0)
MI_6mo: No(0)
PAD: No(0)
Smoking: Yes(1)
Asthma: No(0)
Age: 73

Predict



Prediction Result

The Prediction is: Patient is Not at Risk

7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- **Improved Patient Outcomes**: Enhances survival rates and quality of life.
- **Personalized Treatment Plans**: Tailors care based on individual risk.
- **Informed Clinical Decision-Making**: Aids in evaluating surgery appropriateness.
- **Efficient Resource Allocation**: Optimizes use of medical facilities and staff.
- **Early Intervention**: Identifies high-risk patients for proactive care.
- **Data-Driven Insights**: Uncovers new patterns and factors affecting survival

DISADVANTAGES:

- **Data Quality and Availability**: Relies on comprehensive and accurate data.
- **Complexity of Implementation**: Requires expertise in data science and surgery.
- **Interpretability Issues**: Some models lack transparency in their predictions.
- **Ethical Concerns**: Potential for bias and fairness issues.
- **Dependence on Technology**: Should complement, not replace, clinical judgment.
- **Cost and Maintenance**: Involves significant investment and upkeep.
- **Regulatory and Compliance Issues**: Must adhere to healthcare regulation

8.APPLICATIONS

1. **Patient Counseling and Education**: Informs patients and families about surgery outcomes.
2. **Personalized Follow-up Care**: Customizes follow-up schedules based on risk.
3. **Medication Management**: Tailors post-operative medication regimens.
4. **Lifestyle Adjustments**: Guides necessary lifestyle changes for better outcomes.
5. **Home Care Planning**: Plans home healthcare services based on recovery needs.
6. **Financial Planning**: Helps plan for future healthcare expenses.
7. **Mental Health Support**: Identifies need for psychological counseling.
8. **Community and Support Networks**: Encourages participation in support groups.
9. **Work and Lifestyle Planning**: Assists in planning return to work and daily activities.
10. **Diet and Nutrition Guidance**: Provides dietary recommendations for recovery.

9.CONCLUSION

- Predicting one-year life expectancy post thoracic surgery using machine learning is a significant advancement in patient care and clinical decision-making. By leveraging complex datasets and sophisticated algorithms, healthcare providers can better assess the risks and potential outcomes for individual patients. This approach enhances personalized care, guiding post-operative management, medication plans, and follow-up schedules tailored to each patient's needs.
- Accurate predictions enable patients and their families to make informed decisions about treatment options and lifestyle adjustments, improving overall quality of life. Additionally, this predictive capability supports healthcare systems in allocating resources efficiently and planning long-term care strategies.
- In summary, the integration of machine learning in predicting post-surgery life expectancy represents a crucial step towards more precise, data-driven medical practices, ultimately aiming to improve patient outcomes and optimize healthcare delivery.

10.FUTURE SCOPE

- **Enhanced Predictive Models**: Develop more advanced algorithms incorporating diverse patient data for improved accuracy.
- **Integration with Wearable Technology**: Utilize wearable devices for continuous health monitoring and real-time data updates.
- **Personalized Treatment Plans**: Create individualized treatment and rehabilitation plans based on predictive insights.
- **Telemedicine and Remote Monitoring**: Expand telemedicine for regular followups and remote patient monitoring.
- **AI-Driven Decision Support Systems**: Implement AI systems to aid in presurgical planning and post-operative care.
- **Predictive Analytics in Public Health**: Use aggregated data to inform public health policies and resource allocation.
- **Interdisciplinary Research**: Foster research combining machine learning, bioinformatics, and clinical medicine.
- **Patient Education and Engagement**: Develop tools to help patients understand and engage in their care plans.
- **Regulatory Approvals and Standards**: Work towards regulatory standards and approvals for clinical use.
- **Ethical and Privacy Considerations**: Address ethical concerns and ensure patient data privacy.

11.BIBILOGRAPHY

1. **Falcoz, P. E., Conti, M., Brouchet, L., et al. (2011)**. "The Thoracic Surgery Scoring System (Thoracoscore): Risk Model for In-Hospital Death After Thoracic Surgery." *Annals of Thoracic Surgery*, 91(4), 1063-1068.
2. **Brunelli, A., Charloux, A., Bolliger, C. T., et al. (2009)**. "ERS/ESTS clinical guidelines on fitness for radical therapy in lung cancer patients (surgery and chemo-radiotherapy)." *European Respiratory Journal*, 34(1), 17-41.
3. **Varela, G., Ballesteros, E., Jiménez, M. F., et al. (2003)**. "Cost-effectiveness analysis of prophylactic respiratory physiotherapy in pulmonary lobectomy." *European Journal of Cardio-Thoracic Surgery*, 23(5), 612-619.
4. **Bolliger, C. T., Perruchoud, A. P. (1998)**. "Functional evaluation of the lung resection candidate." *European Respiratory Journal*, 11(1), 198-212.
5. **Miller, D. L., Deschamps, C., Jenkins, G. D., et al. (2001)**. "Comparison of Cardiac and Pulmonary Risk Factors in Patients Having Lung Resection." *Annals of Thoracic Surgery*, 71(2), 435-439.
6. **Cerfolio, R. J., Bryant, A. S., et al. (2009)**. "The accuracy of the assessment of paratracheal lymph nodes in patients with non-small cell lung cancer: a prospective study." *Journal of Thoracic Oncology*, 4(5), 563-568.
7. **American Society of Anesthesiologists (ASA). (2020)**. "ASA Physical Status Classification System." Retrieved from ASA Website.
8. **European Society of Thoracic Surgeons (ESTS). (2014)**. "Guidelines on the radical management of patients with lung cancer." *European Journal of CardioThoracic Surgery*, 45(6), 937-951.
9. **Eisenhauer, E. A., Therasse, P., et al. (2009)**. "New response evaluation criteria in solid tumours: revised RECIST guideline (version 1.1)." *European Journal of Cancer*, 45(2), 228-247.
10. **Dindo, D., Demartines, N., Clavien, P. A. (2004)**. "Classification of surgical complications: a new proposal with evaluation in a cohort of 6336 patients and results of a survey." *Annals of Surgery*, 240(2), 205-213.

12.APPENDIX

Model building :

1)Dataset
2)Jupyter Notebook and Spyder
   1. HTML file (index.html, form.html, result.html)
   1. CSS file
   2. Models in pickle format

SOURCE CODE:

INDEX.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Lung Cancer Prediction</title>
  <link rel="stylesheet" href="{{ url_for('static', filename='styles.css') }}">
    <style>
  body { background: url("{{ url_for('static', filename='welcome.jpg') }}") no-repeat center center fixed;
  background-size: cover;
  color:black;
  }

  .button {
backgroundcolor: #007BFF;
color: white;      padding: 10px
20px;      border: none;
borderradius: 5px;      cursor:
pointer;      font-size: 18px;
font-weight: bold;      margin:
20px auto;      display: block;
width: 200px;      text-align:
center;      text-decoration:
none;
  }
```

```
    .button:hover {
      background-color: #00567F;
    }
  </style>

</head>
<body>

</body>

    <div class="container">
        <header>
            <h1>LIFESPAN</h1>
            <h2>Post Thoracic Surgery</h2>
        </header>
        <main>
            <p style="color:white;">Lung cancer is the leading cause of cancer-related deaths in the world.<br> This
app predicts post-operative life expectancy for patients undergoing thoracic surgery for lung cancer.</p>            <a
href="{{ url_for('form') }}" class="button">View the App</a>
        </main>
    </div>
</body> </html>
```

FORM.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Will the Patient Survive Post Thoracic Surgery?</title>
    <style>        body {
fontfamily: Arial, sans-serif;
margin: 0;            padding: 0;
display: flex;            justify-content:
center;            align-items: center;            min-height:
100vh;            color:
#333;                                                    background-image:
url("{{url_for('static',filename='form.jpg')}}");                    background-
size: cover;            padding:17px;  color:black;
        }
        .container {            width: 90%;
maxwidth: 500px;            background: rgba(255,
255, 255, 0.5);            padding: 30px;
border-radius: 15px;            box-shadow: 0 8px
16px rgba(0, 0, 0, 0.3);            display: flex;
flex-direction: column;            align-items: center;
text-align: center;            backdrop-filter:
blur(10px);            transition:
transform 0.3s ease-in-out;
        }
        .container:hover {            transform:
scale(1.05);
        }
        header {            margin-bottom: 20px;
padding: 10px;            background-color: rgba(0,
0, 0, 0.1);            border-bottom:
1px solid #ddd;            border-radius: 10px
10px 0 0;
        }        header h2 {
margin: 0;
fontsize: 2.5em;
fontweight: bold;
color: #007BFF;
            text-shadow: 1px 1px 2px rgba(0,0,0,0.2);
        }
header p {
margin: 0;            font-size:
1.2em;            color: #666;
font-weight: normal;
            text-shadow: 1px 1px 2px rgba(0, 0, 0, 0.2);
        }
main {
padding: 20px;
width: 100%;
        }        form
{            display:
flex;            flex-
wrap: wrap;
            justify-content: space-between;
        }
        .form-container{
display:flex;            flexwrap:wrap;
gap:20px;
```

```
        }
        .form-group {                 flex: 1 1 45%; /* Flexbox layout for
twocolumn  structure  */                        margin-bottom:  20px;
marginright:2%;
        }
        .form-group label {
display: block;
fontweight: bold;
marginbottom: 5px;
color: #333;            text-align:
left;
        }
        .form-group input[type="number"],
        .form-group select {            width: 100%;            padding:
10px;            border: 1px solid #ddd;            border-radius:
20px; /* More rounded corners */            box-shadow: inset
0 2px 4px rgba(0, 0, 0, 0.1);
transition: border-color 0.3s ease-in-out;
        }
        .form-group input[type="number"]:focus,
        .form-group select:focus {            border-color:
#007BFF;
        }
        form input[type="submit"] {
width: 100%;            padding: 12px 24px;
background-color:
#007BFF;
           color:  white;   border:  none;                    cursor:  pointer;
border-radius:  20px;  /*  More  rounded  corners  */                box-
shadow:  0  4px  8px  rgba(0,  0,  0,  0.1);                  font-size:  1em;
transition: background-color 0.3s ease, transform 0.3s ease;
        }
        form input[type="submit"]:hover {
background-color: #0056b3;            transform:
translateY(-2px);
        }body {            background: url("{{ url_for('static', filename='form.jpg') }}") no-repeat center center
fixed;            background-size: cover;            color:black;
        }

    </style>
</head>
<body>
    <div class="container">
    <header>
      <h2 style="color:brown;">Will the Patient Survive After Post Thoracic Surgery?</h2>
      <p style="color:#164215">Find Out Whether Your Patient Is High Risk Before The Surgery</p>
    </header>
    <main>
      <form action="{{url_for('predict')}}" method="post">
        <div class="form-group">
          <label for="FVC">FVC:</label>
          <input type="number" step="0.01" id="FVC" name="FVC" required style="width:220px;">
        </div>

        <div class="form-group">
          <label for="FEV1">FEV1:</label>
          <input type="number" step="0.01" id="FEV1" name="FEV1" required style="width:215px;">
        </div>
        <div class="form-group">
          <label for="Performance">Performance:</label>
          <select id="Performance" name="Performance" required>
            <option value="1">Good (0)</option>
            <option value="2">Moderate (1)</option>
            <option value="3">Poor (2)</option>
</select>
        </div>
        <div class="form-group">
          <label for="Pain">Pain:</label>
          <select id="Pain" name="Pain" required>
            <option value="1">Yes(1)</option>
            <option value="0">No(0)</option>
          </select>
        </div>
        <div class="form-group">
          <label for="Haemoptysis">Haemoptysis:</label>
          <select id="Haemoptysis" name="Haemoptysis" required>
            <option value="1">Yes(1)</option>
            <option value="0">No(0)</option>
          </select>
        </div>
        <div class="form-group">
          <label for="Dyspnoea">Dyspnoea:</label>
          <select id="Dyspnoea" name="Dyspnoea" required>
            <option value="1">Yes(1)</option>
            <option value="0">No(0)</option>
          </select>
```

```
</div>
<div class="form-group">
  <label for="Cough">Cough:</label>
  <select id="Cough" name="Cough" required>
    <option value="1">Yes(1)</option>
    <option value="0">No(0)</option>
  </select>
</div>
<div class="form-group">
  <label for="Weakness">Weakness:</label>
  <select id="Weakness" name="Weakness" required>
    <option value="1">Yes(1)</option>
    <option value="0">No(0)</option>
  </select>
</div>
<div class="form-group">
  <label for="Tumor_Size">Tumor_Size:</label>
  <input type="number" id="Tumor_Size" name="Tumor_Size" required style="width:220px;">
</div>

<div class="form-group">
  <label for="Diabetes_Mellitus">Diabetes Mellitus:</label>
  <select id="Diabetes_Mellitus" name="Diabetes_Mellitus" required>
    <option value="1">Yes(1)</option>
    <option value="0">No(0)</option>
  </select>
</div>
<div class="form-group">
  <label for="MI_6mo">MI_6mo:</label>
  <select id="MI_6mo" name="MI_6mo" required>
    <option value="1">Yes(1)</option>
    <option value="0">No(0)</option>
  </select>
</div>
<div class="form-group">
  <label for="PAD">PAD:</label>
  <select id="PAD" name="PAD" required>
    <option value="1">Yes(1)</option>
    <option value="0">No(0)</option>
  </select>
</div>
<div class="form-group">
  <label for="Smoking">Smoking:</label>
```

```
          <select id="Smoking" name="Smoking" required>
            <option value="1">Yes(1)</option>
            <option value="0">No(0)</option>
          </select>
        </div>
        <div class="form-group">
          <label for="Asthma">Asthma:</label>
          <select id="Asthma" name="Asthma" required>
            <option value="1">Yes(1)</option>
            <option value="0">No(0)</option>
          </select>
        </div>
        <div class="form-group">
          <label for="Age">Age:</label>
          <input type="number" id="Age" name="Age" required>
        </div>
        <input type="submit" value="Predict">
      </form>
    </main>
  </div>
</body> </html>
```

RESULT.HTML

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>Prediction Result</title>
  <link rel="stylesheet" href="{{url_for('static',filename='styles.css')}}">
<style>    body {      background-image:
url("{{url_for('static',filename='result.jpg')}}");      background-size:
cover;      padding:20px;      color:black;
  }
  .button {
backgroundcolor: #000000;
color: white;      padding: 10px
20px;      border: none;
borderradius: 5px;      cursor:
pointer;      font-size: 18px;
font-weight: bold;      margin:
20px auto;      display: block;
width: 200px;      text-align:
center;
 text-decoration: none;
    }
    .button:hover {
      background-color: #00567F;
    }
    body {       background: url("{{ url_for('static', filename='result.jpg') }}") no-repeat center
center fixed;
   background-size: cover;
  color:black;
    }
   </style>
</head>
<body>
    <div class="container">
        <header>
            <h1 style="color:#747392;">Prediction Result</h1>
        </header>
        <main>
            <h1> {{ prediction_text}}</h1>
            <a href="{{url_for('form')}}" class="button">Back to form</a>
        </main>
    </div>
</body> </html>
```

STYLES.CSS

```
body {   font-family: Arial,
sans-serif;
backgroundcolor: #f0f0f0;
margin:  0;     padding:  0;
display:   flex;        justify-
content:  center;      align-
items: center;  minheight:
100vh;    textalign:center;
color:#333;
}

.container          {
width:          80%;
maxwidth:800px;
```

```
  background-color: rgba(84, 141, 145, 0.5);
padding:  20px;      box-shadow:  0  0  10px
rgba(0,  0,  0,  0.1);      border-radius:  10px;
overflow:    hidden;            display:    flex;
flexdirection: column;   align-items: center;
margin:0 auto;
  color:black;
}

header       {                    text-align:center;
marginbottom:  20px;       padding:  10px;
background-color: rgba(247,247,247,0.8);
border-bottom: 1px solid #ddd;
}

header        h1        {
margin:0;   font-size:
2.5em;   fontweight:
bold;   color: #333;
}

header h2 {   margin:0;
font-size:
1.8em;   color: #666;   font-weight:
normal;
}

main {   text-align: center;
padding:
20px;
}

form  {      margin-top:
20px;      display:  flex;
flex-wrap:          wrap;
justify-content: center;
gap:10px;
}

label  {      display:  block;
margin:    10px    0    5px;
width:  calc(100%  -  20px);
padding:                  10px;
```

```
background-color:
rgba(247,247,247,0.8);
border-radius:            5px;
boxshadow:    0    0    5px
rgba(0, 0,
0, 0.1);
}

input[type="number"], select {   width: calc(100%
- 20px);   padding: 10px;   margin-bottom: 10px;
border: none;   border-radius: 5px;   box-shadow:
0 0
5px rgba(0, 0, 0, 0.1);
}

input[type="submit"] {   padding: 10px
20px;      background-color:  #007BFF;
color: white;    border: none;    cursor:
pointer;             border-radius:     5px;
boxshadow: 0 0 5px rgba(0, 0, 0, 0.1);
transition:background-color 0.3s ease;
}

input[type="submit"]:hover {   background-
color: #00567F;
}

input[type="number"]:focus, select:focus {
outline: none;   box-shadow: 0 0 5px
rgba(0, 0, 0, 0.2);
}

.container {   box-shadow: 0 0 10px rgba(0,0,0, 0.1), 0 0 10px rgba(0,0,0,
0.1) inset;
}

header {   background-image: linear-gradient(to bottom, rgba(247,247,247,0.8),
rgba(247,247,247,0.8));
}

label {   background-image: linear-gradient(to bottom, rgba(247,247,247,0.8),
rgba(247,247,247,0.8));
}
```

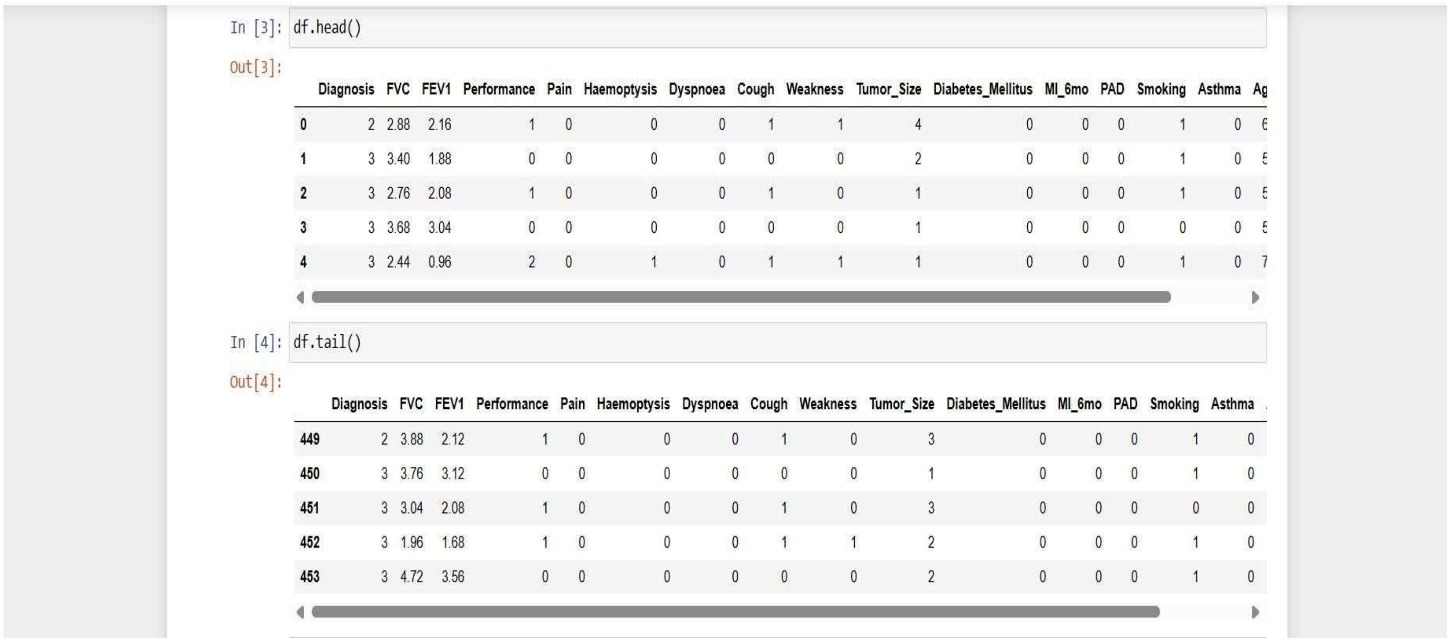
```
input[type="submit"] {   background-image: linear-gradient(to
bottom, #007BFF, #00567F);
}
```

APP.PY

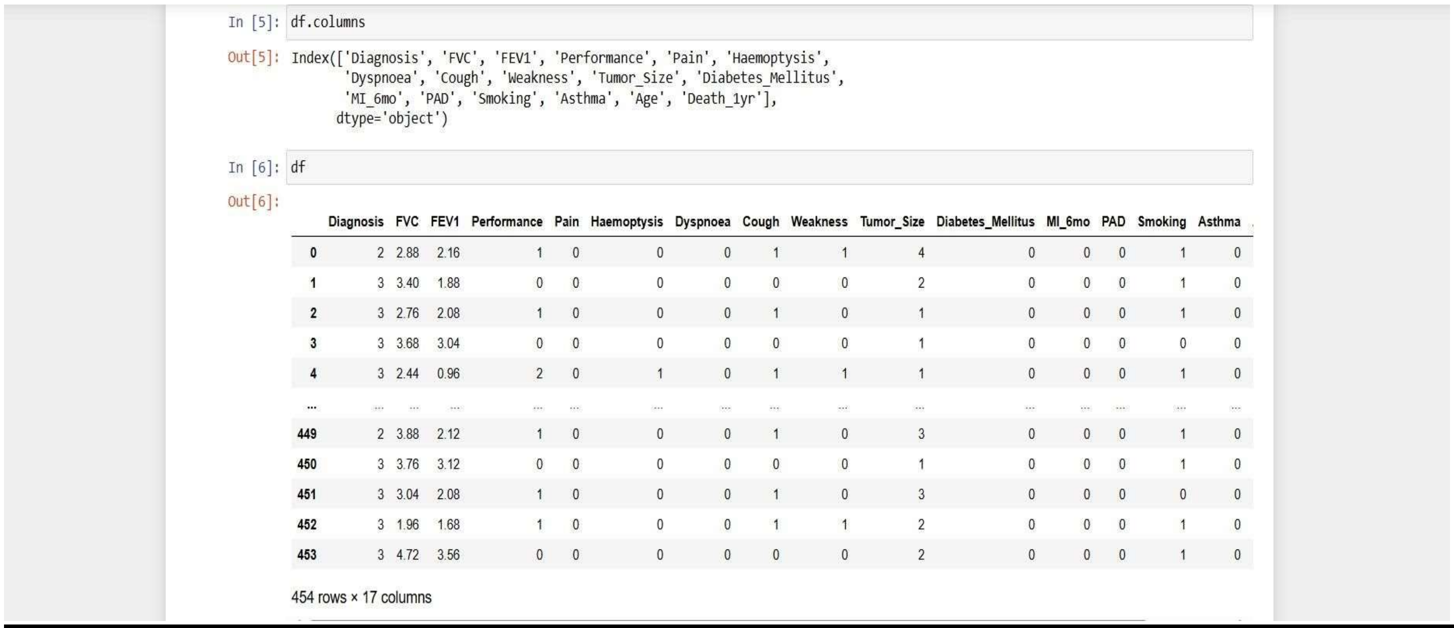
```
from flask import Flask, render_template, request
import numpy as np import pickle


app = Flask(__name__)


# Load your model
model = pickle.load(open('best_random_forest_model.pkl', 'rb'))


with open('scaler.pkl', 'rb') as scaler_file:
   scaler = pickle.load(scaler_file)


@app.route('/') def home():
   return render_template('index.html')


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


@app.route('/predict', methods=['POST']) def
predict():     try:
      # Get the form data
form_data = request.form.to_dict()
# Convert form data to numpy array
      input_features = np.array([[float(form_data['FVC']),
float(form_data['FEV1']),
int(form_data['Performance']),
int(form_data['Pain']),
int(form_data['Haemoptysis']),
int(form_data['Dyspnoea']),
 int(form_data['Cough']),
int(form_data['Weakness']),
float(form_data['Tumor_Size']),
int(form_data['Diabetes_Mellitus']),
int(form_data['MI_6mo']),
 int(form_data['PAD']),
 int(form_data['Smoking']),
int(form_data['Asthma']),
int(form_data['Age'])]])

        # Normalize the input features
        input_features_normalized = scaler.transform(input_features)

        # Make prediction with the loaded model
        prediction = model.predict(input_features_normalized)


        # Determine the prediction result
        prediction_text = 'Patient is at High Risk' if prediction[0] == 0 else 'Patient is Not at Risk'


        return render_template('result.html', prediction_text=f'The Prediction is: {"Patient is at High Risk" if
prediction[0] == 0 else "Patient is Not at Risk"}')     except Exception as e:
return render_template('index.html', prediction_text=f'Error: {str(e)}')

if __name__ == "__main__":
    app.run(debug=True)
```

CODE SNIPPETS

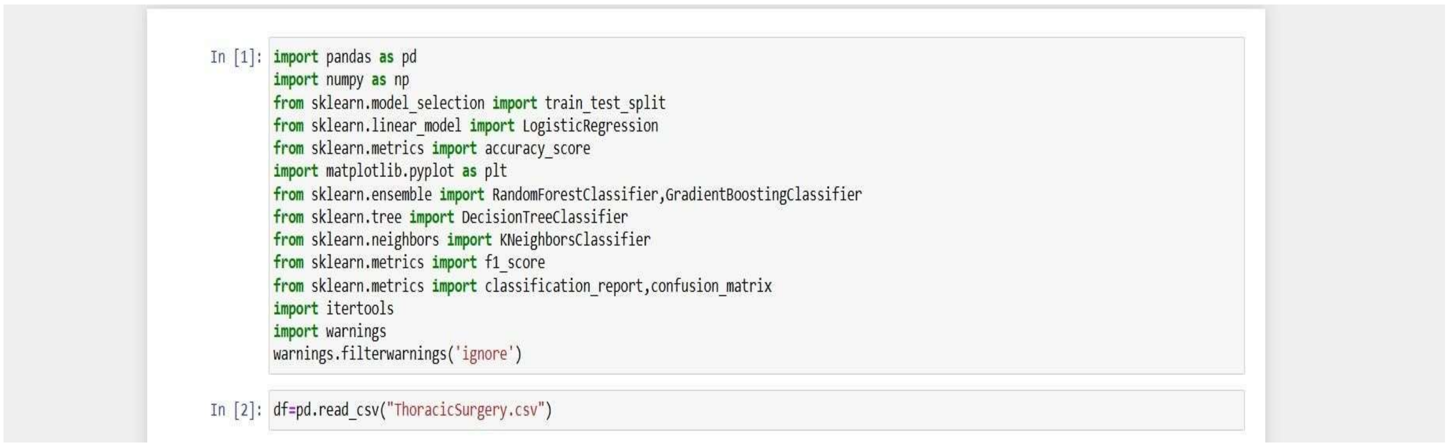
MODEL BUILDING



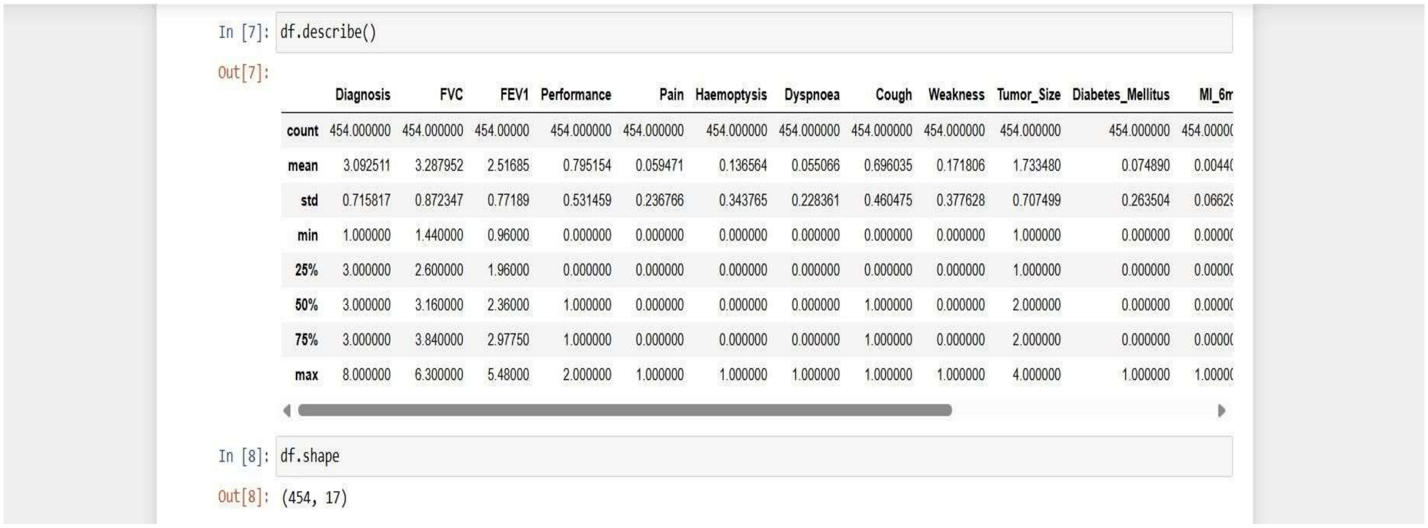
```
In [3]: df.head()

Out[3]:
     Diagnosis  FVC  FEV1  Performance  Pain  Haemoptysis  Dyspnoea  Cough  Weakness  Tumor_Size  Diabetes_Mellitus  MI_6mo  PAD  Smoking  Asthma  Ag
  0          2  2.88  2.16            1     0            0         0      1         1           4                  0       0    0        1       0  6
  1          3  3.40  1.88            0     0            0         0      0         0           2                  0       0    0        1       0  5
  2          3  2.76  2.08            1     0            0         0      1         0           1                  0       0    0        1       0  5
  3          3  3.68  3.04            0     0            0         0      0         0           1                  0       0    0        0       0  5
  4          3  2.44  0.96            2     0            1         0      1         1           1                  0       0    0        1       0  7

In [4]: df.tail()

Out[4]:
       Diagnosis  FVC  FEV1  Performance  Pain  Haemoptysis  Dyspnoea  Cough  Weakness  Tumor_Size  Diabetes_Mellitus  MI_6mo  PAD  Smoking  Asthma
  449          2  3.88  2.12            1     0            0         0      1         0           3                  0       0    0        1       0
  450          3  3.76  3.12            0     0            0         0      0         0           1                  0       0    0        1       0
  451          3  3.04  2.08            1     0            0         0      1         0           3                  0       0    0        0       0
  452          3  1.96  1.68            1     0            0         0      1         1           2                  0       0    0        1       0
  453          3  4.72  3.56            0     0            0         0      0         0           2                  0       0    0        1       0
```



```
In [5]: df.columns

Out[5]: Index(['Diagnosis', 'FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis',
               'Dyspnoea', 'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus',
               'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age', 'Death_1yr'],
              dtype='object')

In [6]: df

Out[6]:
       Diagnosis  FVC  FEV1  Performance  Pain  Haemoptysis  Dyspnoea  Cough  Weakness  Tumor_Size  Diabetes_Mellitus  MI_6mo  PAD  Smoking  Asthma
    0          2  2.88  2.16            1     0            0         0      1         1           4                  0       0    0        1       0
    1          3  3.40  1.88            0     0            0         0      0         0           2                  0       0    0        1       0
    2          3  2.76  2.08            1     0            0         0      1         0           1                  0       0    0        1       0
    3          3  3.68  3.04            0     0            0         0      0         0           1                  0       0    0        0       0
    4          3  2.44  0.96            2     0            1         0      1         1           1                  0       0    0        1       0
  ...        ...   ...   ...          ...   ...          ...       ...    ...       ...         ...                ...     ...  ...      ...     ...
  449          2  3.88  2.12            1     0            0         0      1         0           3                  0       0    0        1       0
  450          3  3.76  3.12            0     0            0         0      0         0           1                  0       0    0        1       0
  451          3  3.04  2.08            1     0            0         0      1         0           3                  0       0    0        0       0
  452          3  1.96  1.68            1     0            0         0      1         1           2                  0       0    0        1       0
  453          3  4.72  3.56            0     0            0         0      0         0           2                  0       0    0        1       0

454 rows × 17 columns
```



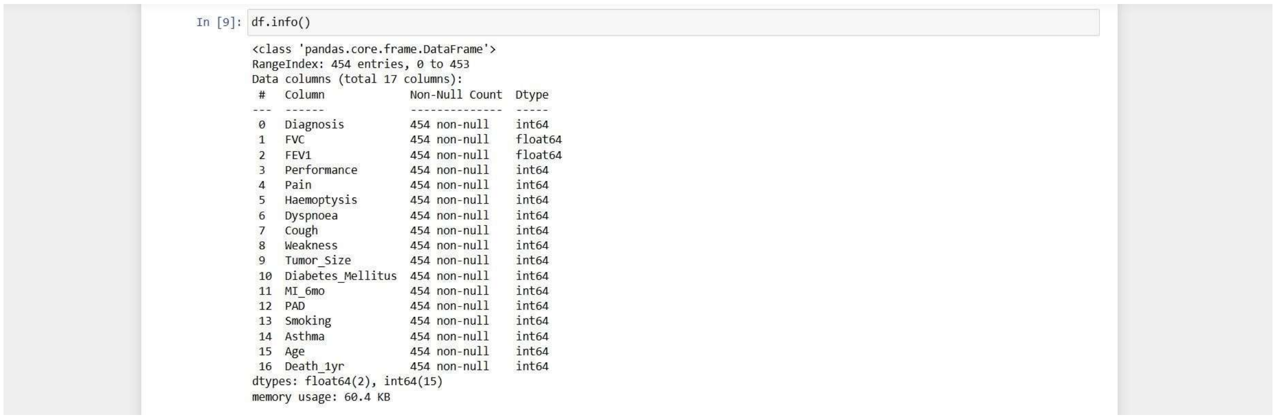
```
In [1]: import pandas as pd
        import numpy as np
        from sklearn.model_selection import train_test_split
        from sklearn.linear_model import LogisticRegression
        from sklearn.metrics import accuracy_score
        import matplotlib.pyplot as plt
        from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier
        from sklearn.tree import DecisionTreeClassifier
        from sklearn.neighbors import KNeighborsClassifier
        from sklearn.metrics import f1_score
        from sklearn.metrics import classification_report,confusion_matrix
        import itertools
        import warnings
        warnings.filterwarnings('ignore')

In [2]: df=pd.read_csv("ThoracicSurgery.csv")
```



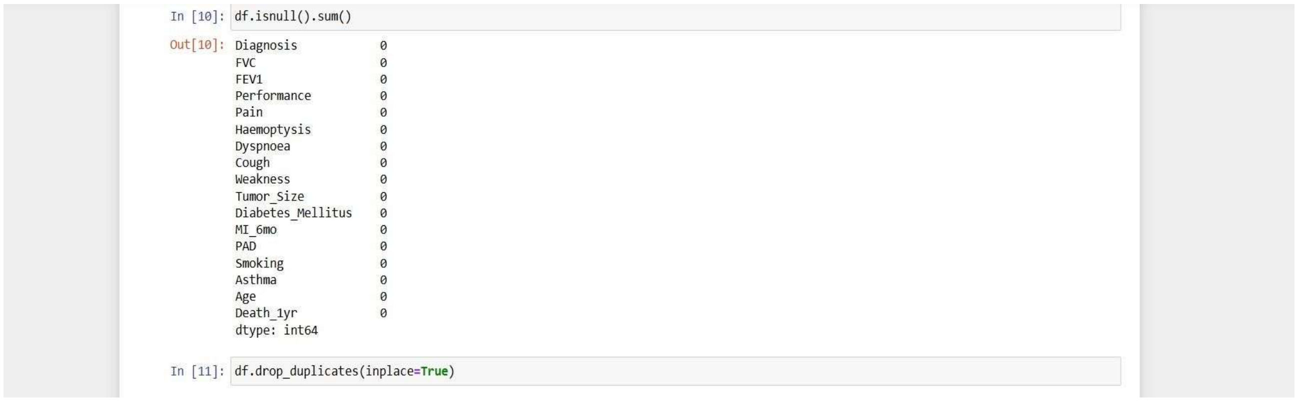
```
In [7]: df.describe()

Out[7]:
```

	Diagnosis	FVC	FEV1	Performance	Pain	Haemoptysis	Dyspnoea	Cough	Weakness	Tumor_Size	Diabetes_Mellitus	MI_6m
count	454.000000	454.000000	454.00000	454.000000	454.000000	454.000000	454.000000	454.000000	454.000000	454.000000	454.000000	454.0000
mean	3.092511	3.287952	2.51685	0.795154	0.059471	0.136564	0.055066	0.696035	0.171806	1.733480	0.074890	0.0044
std	0.715817	0.872347	0.77189	0.531459	0.236766	0.343765	0.228361	0.460475	0.377628	0.707499	0.263504	0.0662
min	1.000000	1.440000	0.96000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.0000
25%	3.000000	2.600000	1.96000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	0.000000	0.0000
50%	3.000000	3.160000	2.36000	1.000000	0.000000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000	0.0000
75%	3.000000	3.840000	2.97750	1.000000	0.000000	0.000000	0.000000	1.000000	0.000000	2.000000	0.000000	0.0000
max	8.000000	6.300000	5.48000	2.000000	1.000000	1.000000	1.000000	1.000000	1.000000	4.000000	1.000000	1.0000

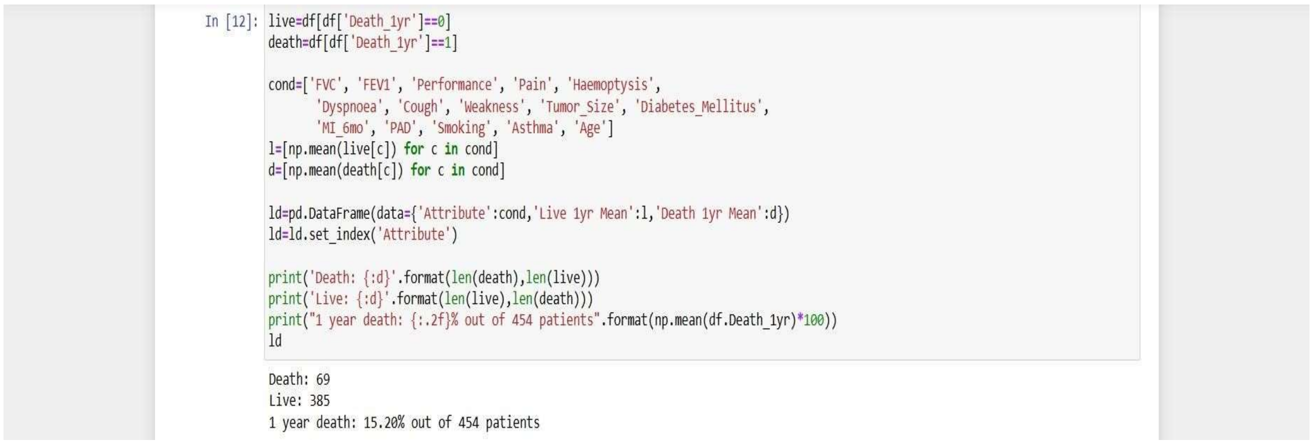
```
In [8]: df.shape

Out[8]: (454, 17)
```



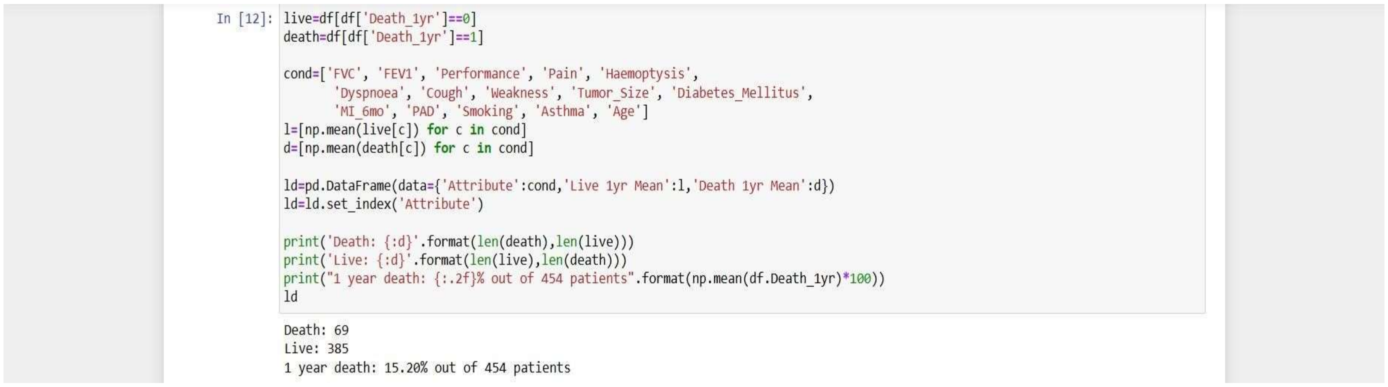
```
In [9]: df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 454 entries, 0 to 453
Data columns (total 17 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Diagnosis          454 non-null    int64
 1   FVC                454 non-null    float64
 2   FEV1               454 non-null    float64
 3   Performance        454 non-null    int64
 4   Pain               454 non-null    int64
 5   Haemoptysis        454 non-null    int64
 6   Dyspnoea           454 non-null    int64
 7   Cough              454 non-null    int64
 8   Weakness           454 non-null    int64
 9   Tumor_Size         454 non-null    int64
 10  Diabetes_Mellitus  454 non-null    int64
 11  MI_6mo             454 non-null    int64
 12  PAD                454 non-null    int64
 13  Smoking            454 non-null    int64
 14  Asthma             454 non-null    int64
 15  Age                454 non-null    int64
 16  Death_1yr          454 non-null    int64
dtypes: float64(2), int64(15)
memory usage: 60.4 KB
```



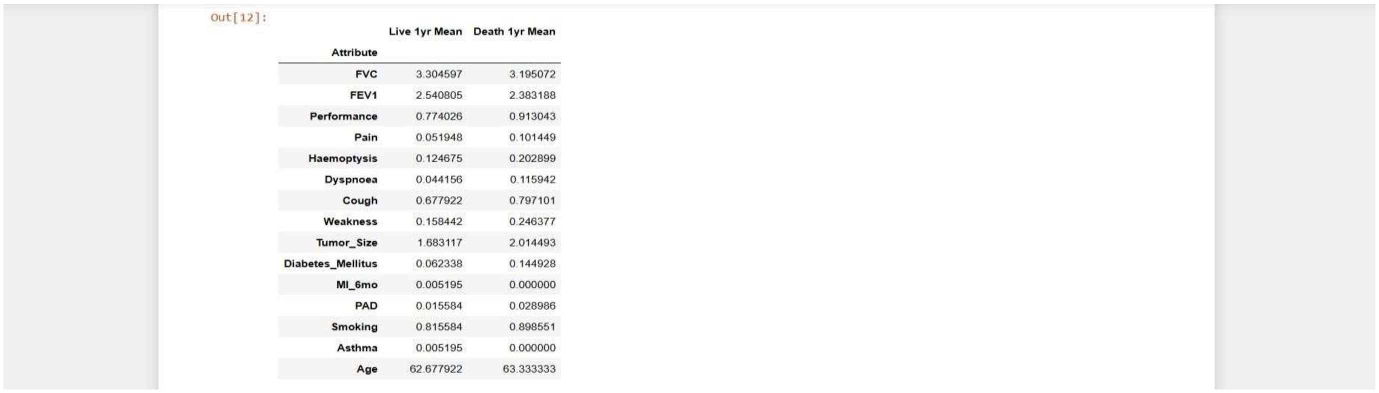
```
In [10]: df.isnull().sum()

Out[10]: Diagnosis            0
         FVC                  0
         FEV1                 0
         Performance          0
         Pain                 0
         Haemoptysis          0
         Dyspnoea             0
         Cough                0
         Weakness             0
         Tumor_Size           0
         Diabetes_Mellitus    0
         MI_6mo               0
         PAD                  0
         Smoking              0
         Asthma               0
         Age                  0
         Death_1yr            0
         dtype: int64

In [11]: df.drop_duplicates(inplace=True)
```



```
In [12]: live=df[df['Death_1yr']==0]
         death=df[df['Death_1yr']==1]

         cond=['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis',
               'Dyspnoea', 'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus',
               'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age']
         l=[np.mean(live[c]) for c in cond]
         d=[np.mean(death[c]) for c in cond]

         ld=pd.DataFrame(data={'Attribute':cond,'Live 1yr Mean':l,'Death 1yr Mean':d})
         ld=ld.set_index('Attribute')

         print('Death: {:d}'.format(len(death),len(live)))
         print('Live: {:d}'.format(len(live),len(death)))
         print("1 year death: {:.2f}% out of 454 patients".format(np.mean(df.Death_1yr)*100))
         ld

         Death: 69
         Live: 385
         1 year death: 15.20% out of 454 patients
```

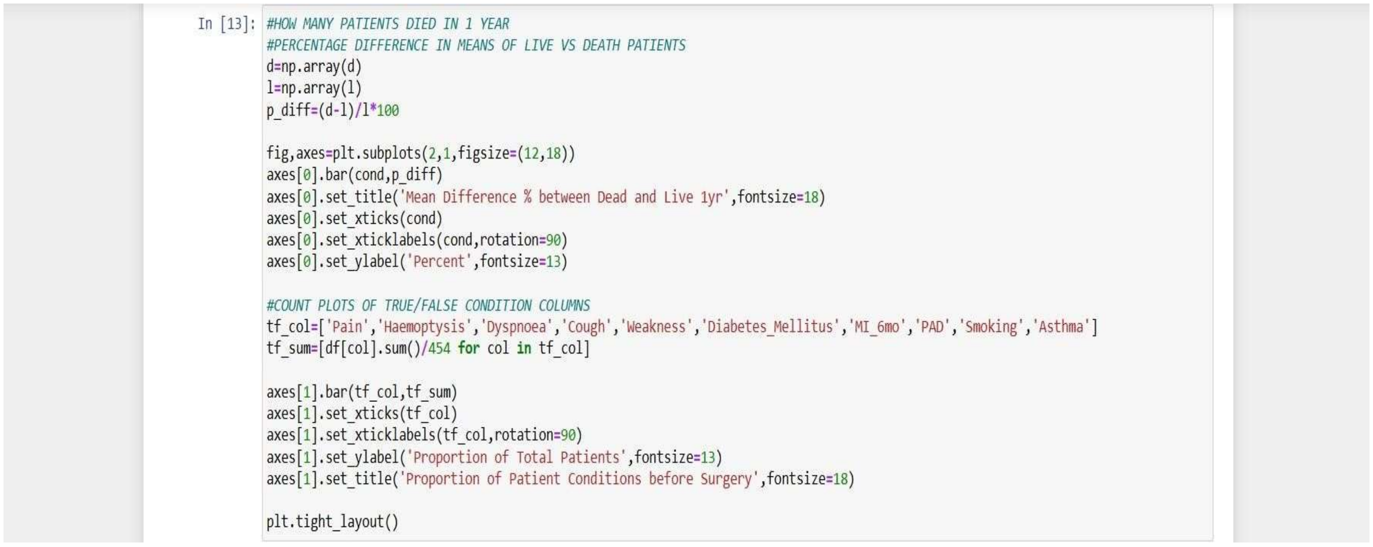


```
In [12]: live=df[df['Death_1yr']==0]
         death=df[df['Death_1yr']==1]

         cond=['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis',
                'Dyspnoea', 'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus',
                'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age']
         l=[np.mean(live[c]) for c in cond]
         d=[np.mean(death[c]) for c in cond]

         ld=pd.DataFrame(data={'Attribute':cond,'Live 1yr Mean':l,'Death 1yr Mean':d})
         ld=ld.set_index('Attribute')

         print('Death: {:d}'.format(len(death),len(live)))
         print('Live: {:d}'.format(len(live),len(death)))
         print("1 year death: {:.2f}% out of 454 patients".format(np.mean(df.Death_1yr)*100))
         ld

         Death: 69
         Live: 385
         1 year death: 15.20% out of 454 patients
```

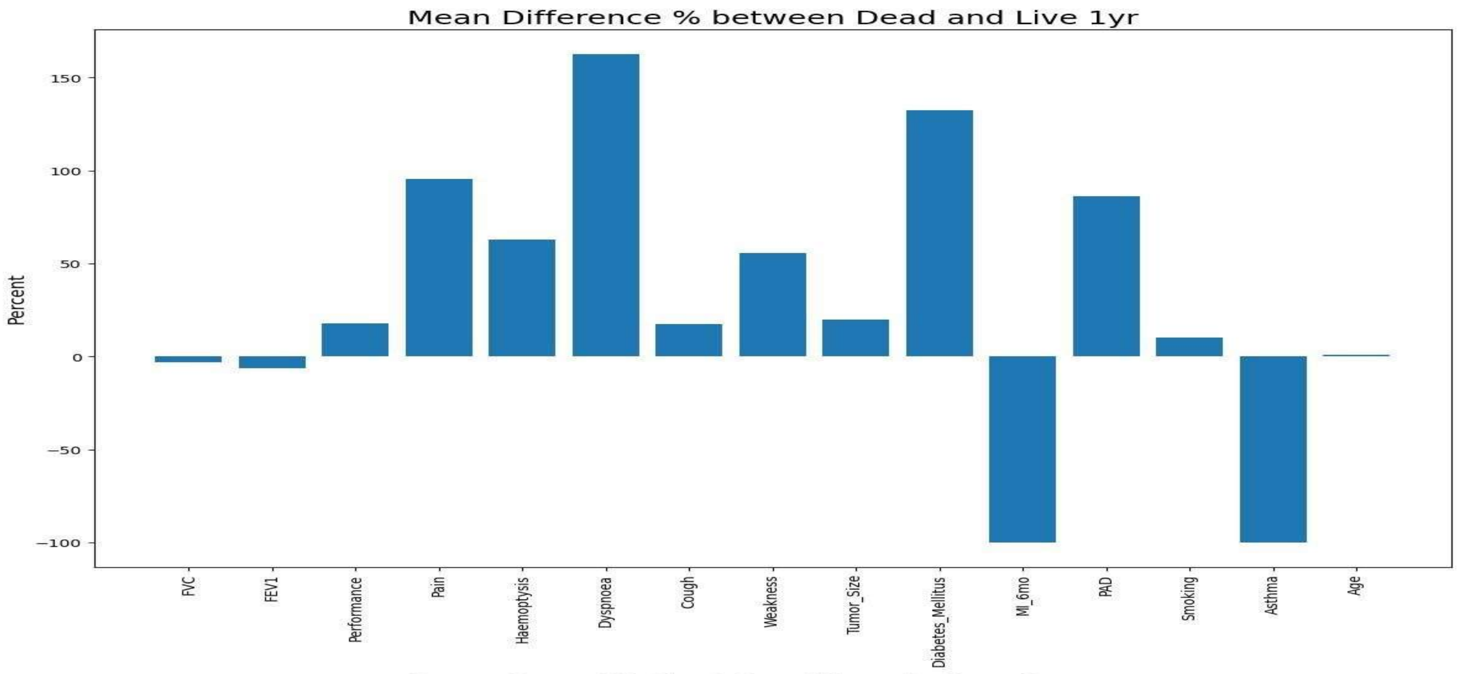


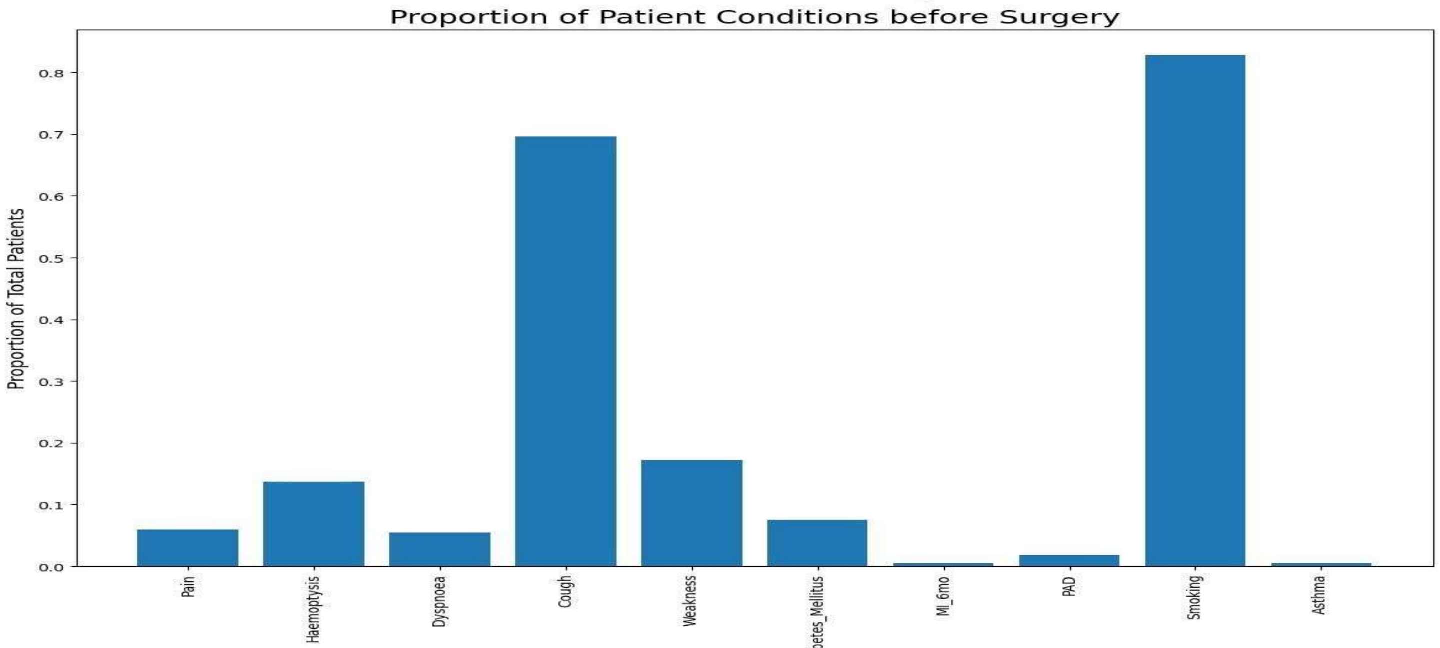
Out[12]:

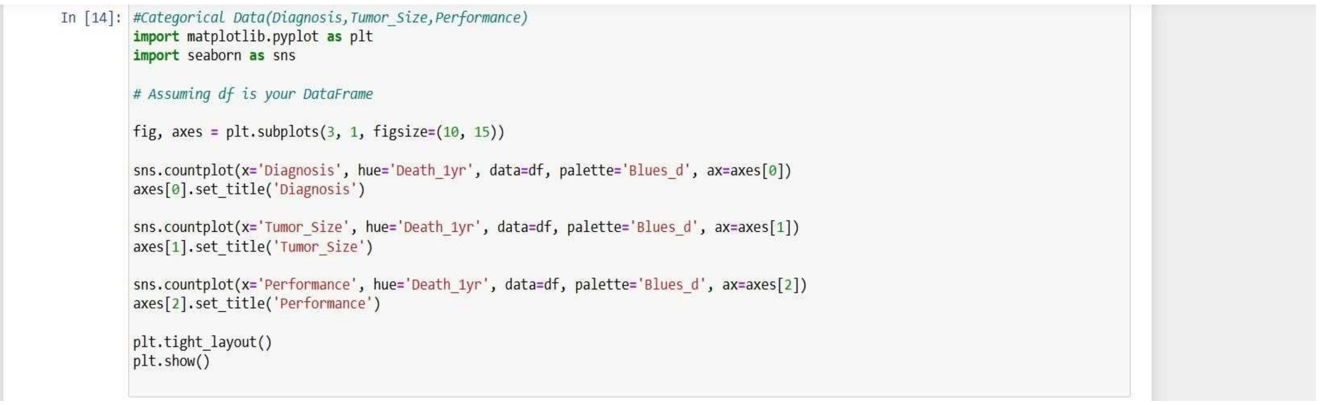
	Live 1yr Mean	Death 1yr Mean
Attribute		
FVC	3.304597	3.195072
FEV1	2.540805	2.383188
Performance	0.774026	0.913043
Pain	0.051948	0.101449
Haemoptysis	0.124675	0.202899
Dyspnoea	0.044156	0.115942
Cough	0.677922	0.797101
Weakness	0.158442	0.246377
Tumor_Size	1.683117	2.014493
Diabetes_Mellitus	0.062338	0.144928
MI_6mo	0.005195	0.000000
PAD	0.015584	0.028986
Smoking	0.815584	0.898551
Asthma	0.005195	0.000000
Age	62.677922	63.333333



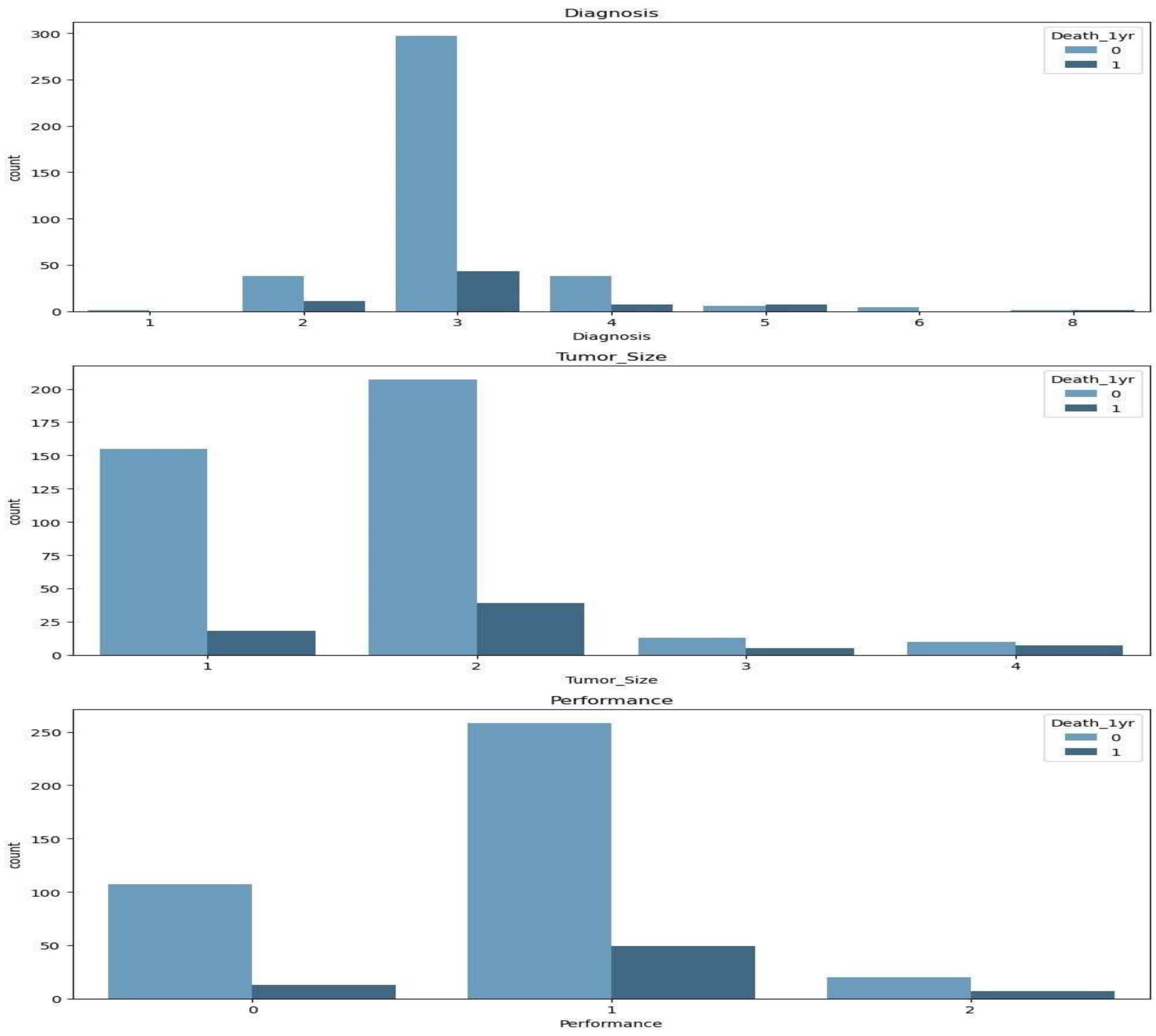
```
In [13]: #HOW MANY PATIENTS DIED IN 1 YEAR
         #PERCENTAGE DIFFERENCE IN MEANS OF LIVE VS DEATH PATIENTS
         d=np.array(d)
         l=np.array(l)
         p_diff=(d-l)/l*100

         fig,axes=plt.subplots(2,1,figsize=(12,18))
         axes[0].bar(cond,p_diff)
         axes[0].set_title('Mean Difference % between Dead and Live 1yr',fontsize=18)
         axes[0].set_xticks(cond)
         axes[0].set_xticklabels(cond,rotation=90)
         axes[0].set_ylabel('Percent',fontsize=13)

         #COUNT PLOTS OF TRUE/FALSE CONDITION COLUMNS
         tf_col=['Pain','Haemoptysis','Dyspnoea','Cough','Weakness','Diabetes_Mellitus','MI_6mo','PAD','Smoking','Asthma']
         tf_sum=[df[col].sum()/454 for col in tf_col]

         axes[1].bar(tf_col,tf_sum)
         axes[1].set_xticks(tf_col)
         axes[1].set_xticklabels(tf_col,rotation=90)
         axes[1].set_ylabel('Proportion of Total Patients',fontsize=13)
         axes[1].set_title('Proportion of Patient Conditions before Surgery',fontsize=18)

         plt.tight_layout()
```

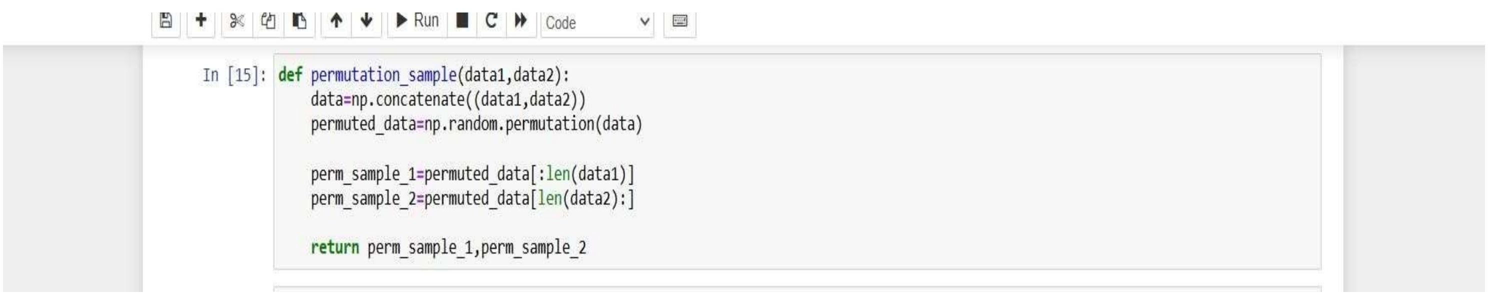

Mean Difference % between Dead and Live 1yr
Percent
150
100
50
0
−50
−100
FVC
FEV1
Performance
Pain
Haemoptysis
Dyspnoea
Cough
Weakness
Tumor_Size
Diabetes_Mellitus
MI_6mo
PAD
Smoking
Asthma
Age


Proportion of Patient Conditions before Surgery
Proportion of Total Patients
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Pain
Haemoptysis
Dyspnoea
Cough
Weakness
Diabetes_Mellitus
MI_6mo
PAD
Smoking
Asthma

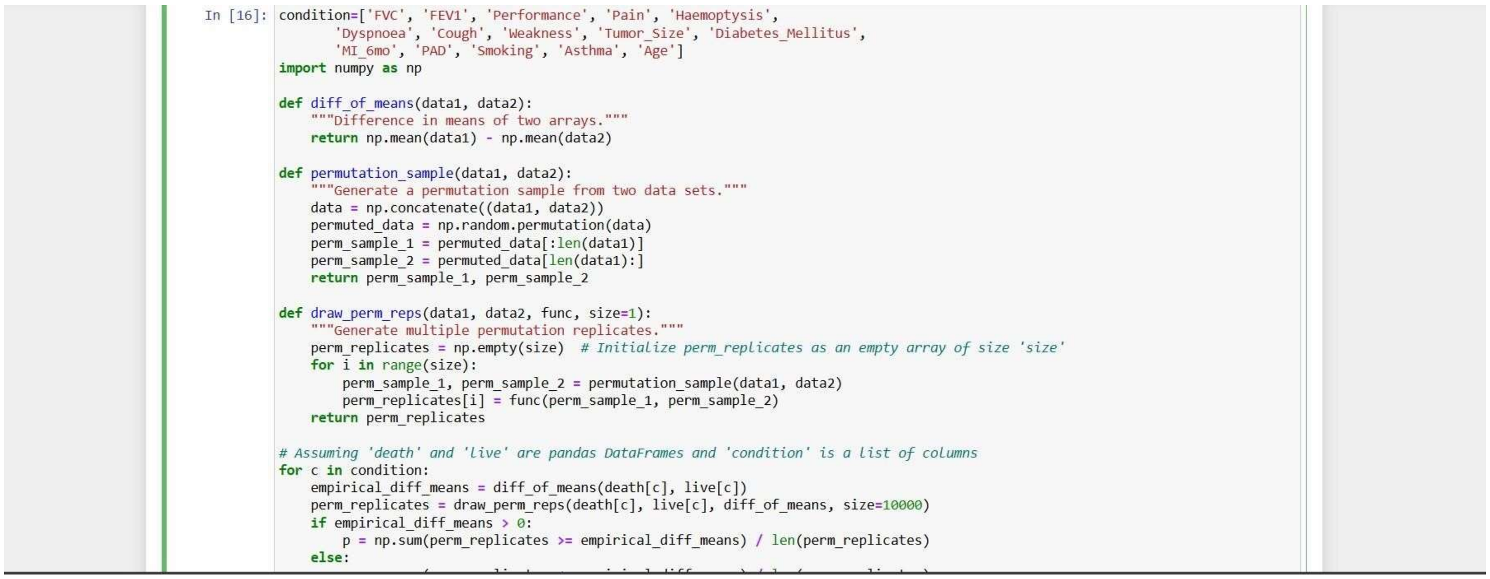


```
In [14]: #Categorical Data(Diagnosis,Tumor_Size,Performance)
         import matplotlib.pyplot as plt
         import seaborn as sns

         # Assuming df is your DataFrame

         fig, axes = plt.subplots(3, 1, figsize=(10, 15))

         sns.countplot(x='Diagnosis', hue='Death_1yr', data=df, palette='Blues_d', ax=axes[0])
         axes[0].set_title('Diagnosis')

         sns.countplot(x='Tumor_Size', hue='Death_1yr', data=df, palette='Blues_d', ax=axes[1])
         axes[1].set_title('Tumor_Size')

         sns.countplot(x='Performance', hue='Death_1yr', data=df, palette='Blues_d', ax=axes[2])
         axes[2].set_title('Performance')

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

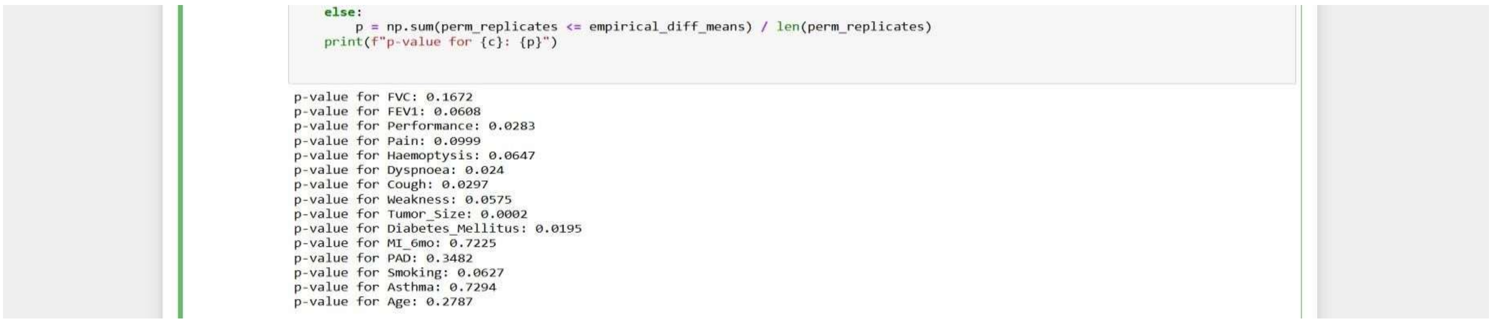




```
In [15]: def permutation_sample(data1,data2):
             data=np.concatenate((data1,data2))
             permuted_data=np.random.permutation(data)

             perm_sample_1=permuted_data[:len(data1)]
             perm_sample_2=permuted_data[len(data2):]

             return perm_sample_1,perm_sample_2
```

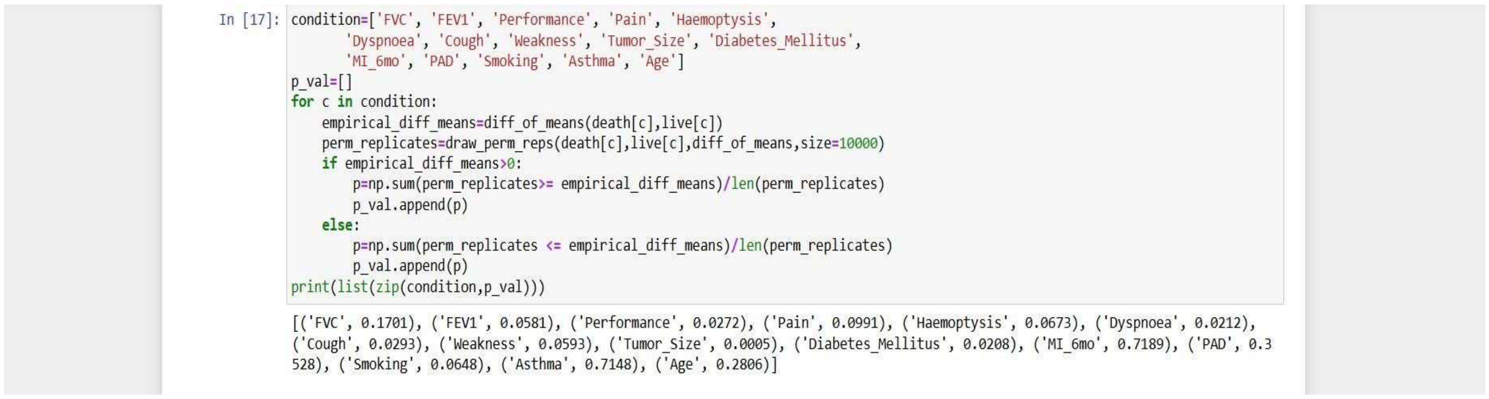


```
In [16]: condition=['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis',
                'Dyspnoea', 'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus',
                'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age']
         import numpy as np

         def diff_of_means(data1, data2):
             """Difference in means of two arrays."""
             return np.mean(data1) - np.mean(data2)

         def permutation_sample(data1, data2):
             """Generate a permutation sample from two data sets."""
             data = np.concatenate((data1, data2))
             permuted_data = np.random.permutation(data)
             perm_sample_1 = permuted_data[:len(data1)]
             perm_sample_2 = permuted_data[len(data1):]
             return perm_sample_1, perm_sample_2

         def draw_perm_reps(data1, data2, func, size=1):
             """Generate multiple permutation replicates."""
             perm_replicates = np.empty(size)  # Initialize perm_replicates as an empty array of size 'size'
             for i in range(size):
                 perm_sample_1, perm_sample_2 = permutation_sample(data1, data2)
                 perm_replicates[i] = func(perm_sample_1, perm_sample_2)
             return perm_replicates

         # Assuming 'death' and 'live' are pandas DataFrames and 'condition' is a list of columns
         for c in condition:
             empirical_diff_means = diff_of_means(death[c], live[c])
             perm_replicates = draw_perm_reps(death[c], live[c], diff_of_means, size=10000)
             if empirical_diff_means > 0:
                 p = np.sum(perm_replicates >= empirical_diff_means) / len(perm_replicates)
             else:
```

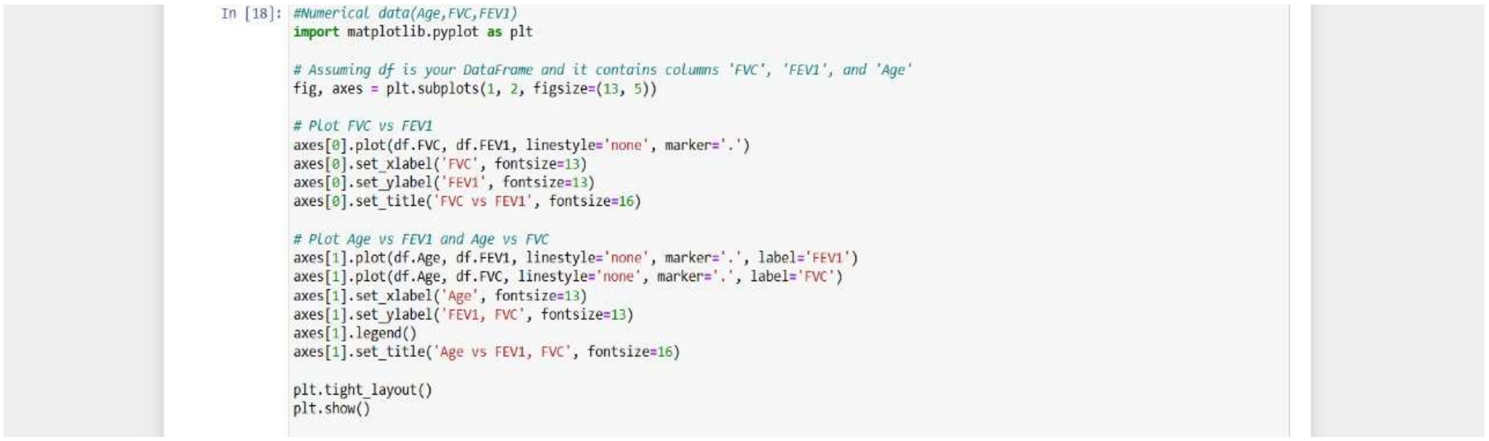


```
    else:
        p = np.sum(perm_replicates <= empirical_diff_means) / len(perm_replicates)
    print(f"p-value for {c}: {p}")
```

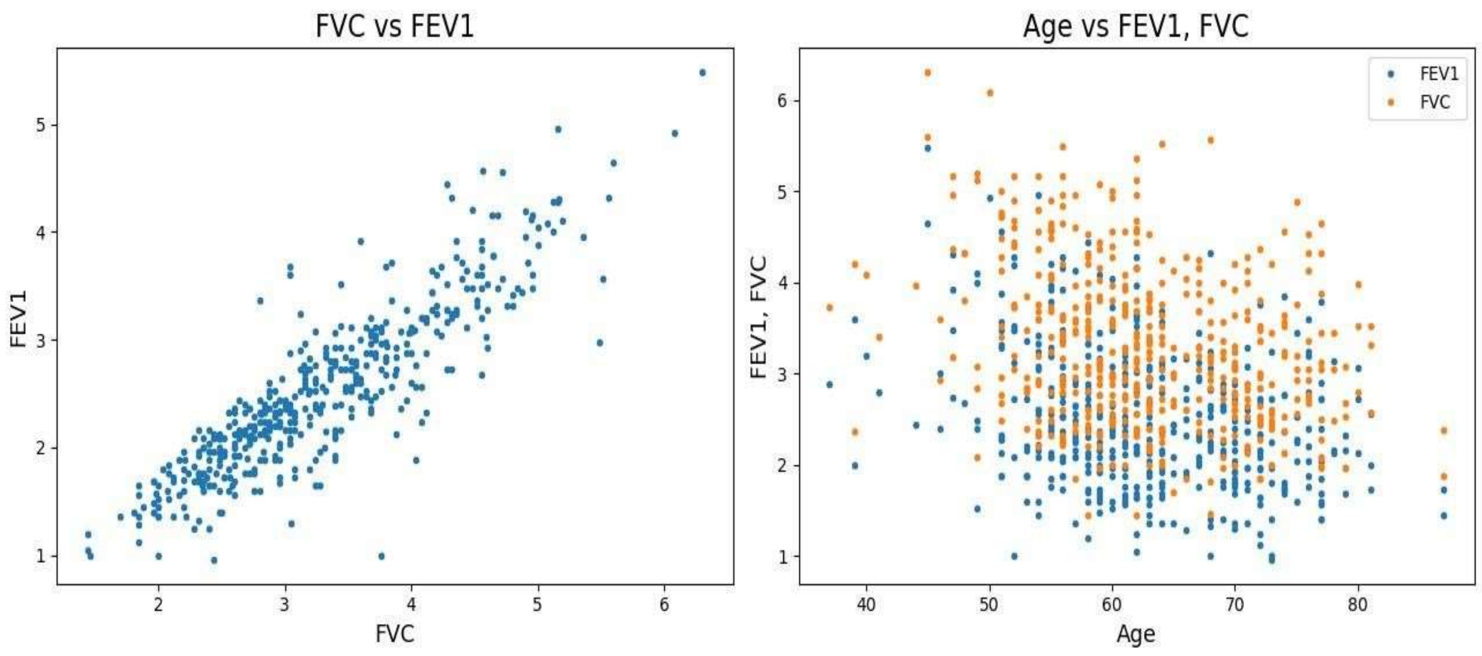
```
p-value for FVC: 0.1672
p-value for FEV1: 0.0608
p-value for Performance: 0.0283
p-value for Pain: 0.0999
p-value for Haemoptysis: 0.0647
p-value for Dyspnoea: 0.024
p-value for Cough: 0.0297
p-value for Weakness: 0.0575
p-value for Tumor_Size: 0.0002
p-value for Diabetes_Mellitus: 0.0195
p-value for MI_6mo: 0.7225
p-value for PAD: 0.3482
p-value for Smoking: 0.0627
p-value for Asthma: 0.7294
p-value for Age: 0.2787
```

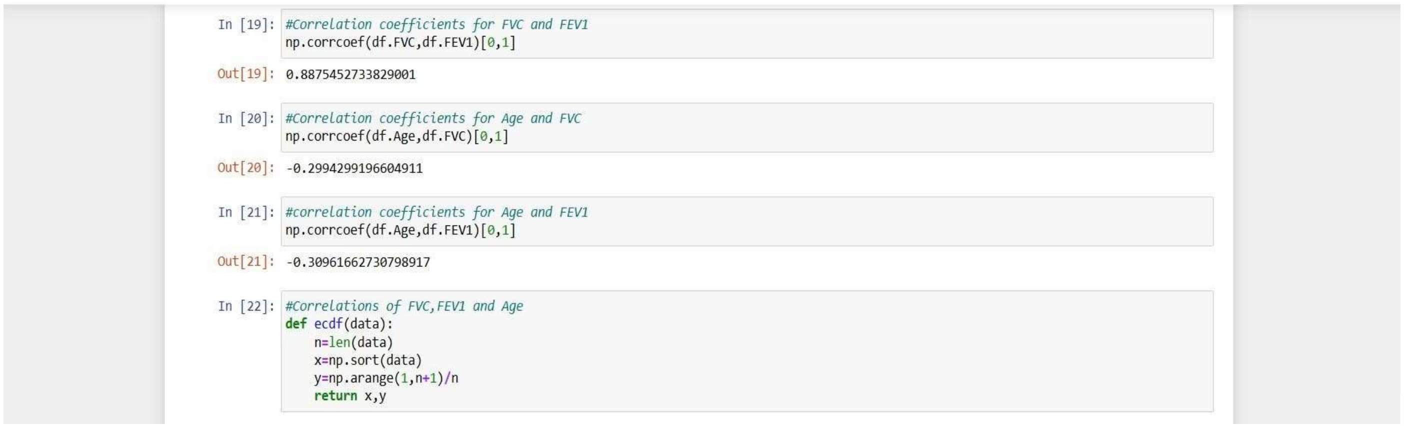


```
In [17]: condition=['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis',
                'Dyspnoea', 'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus',
                'MI_6mo', 'PAD', 'Smoking', 'Asthma', 'Age']
         p_val=[]
         for c in condition:
             empirical_diff_means=diff_of_means(death[c],live[c])
             perm_replicates=draw_perm_reps(death[c],live[c],diff_of_means,size=10000)
             if empirical_diff_means>0:
                 p=np.sum(perm_replicates>= empirical_diff_means)/len(perm_replicates)
                 p_val.append(p)
             else:
                 p=np.sum(perm_replicates <= empirical_diff_means)/len(perm_replicates)
                 p_val.append(p)
         print(list(zip(condition,p_val)))

         [('FVC', 0.1701), ('FEV1', 0.0581), ('Performance', 0.0272), ('Pain', 0.0991), ('Haemoptysis', 0.0673), ('Dyspnoea', 0.0212),
         ('Cough', 0.0293), ('Weakness', 0.0593), ('Tumor_Size', 0.0005), ('Diabetes_Mellitus', 0.0208), ('MI_6mo', 0.7189), ('PAD', 0.3
         528), ('Smoking', 0.0648), ('Asthma', 0.7148), ('Age', 0.2806)]
```

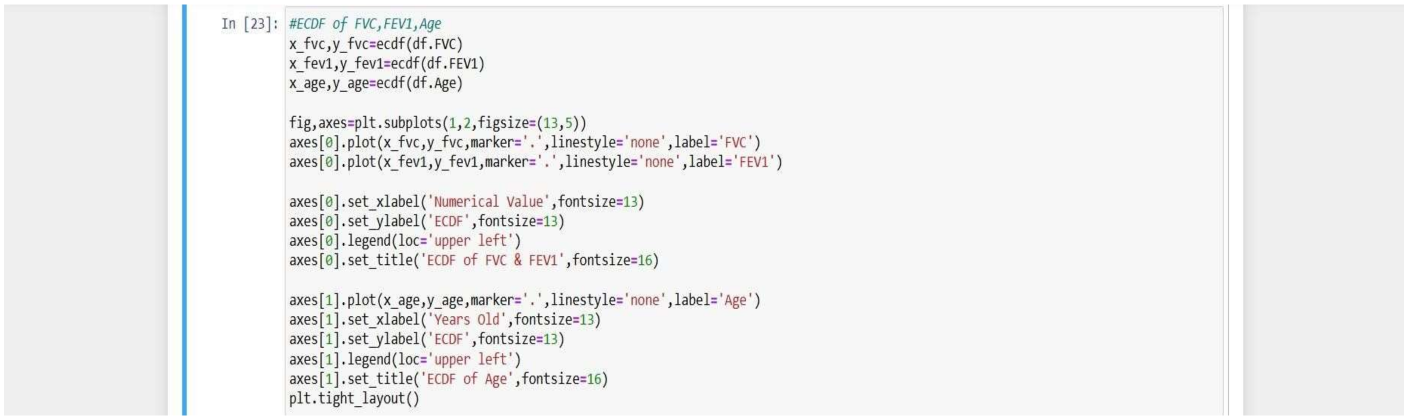


```
In [18]: #Numerical data(Age,FVC,FEV1)
         import matplotlib.pyplot as plt

         # Assuming df is your DataFrame and it contains columns 'FVC', 'FEV1', and 'Age'
         fig, axes = plt.subplots(1, 2, figsize=(13, 5))

         # Plot FVC vs FEV1
         axes[0].plot(df.FVC, df.FEV1, linestyle='none', marker='.')
         axes[0].set_xlabel('FVC', fontsize=13)
         axes[0].set_ylabel('FEV1', fontsize=13)
         axes[0].set_title('FVC vs FEV1', fontsize=16)

         # Plot Age vs FEV1 and Age vs FVC
         axes[1].plot(df.Age, df.FEV1, linestyle='none', marker='.', label='FEV1')
         axes[1].plot(df.Age, df.FVC, linestyle='none', marker='.', label='FVC')
         axes[1].set_xlabel('Age', fontsize=13)
         axes[1].set_ylabel('FEV1, FVC', fontsize=13)
         axes[1].legend()
         axes[1].set_title('Age vs FEV1, FVC', fontsize=16)

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

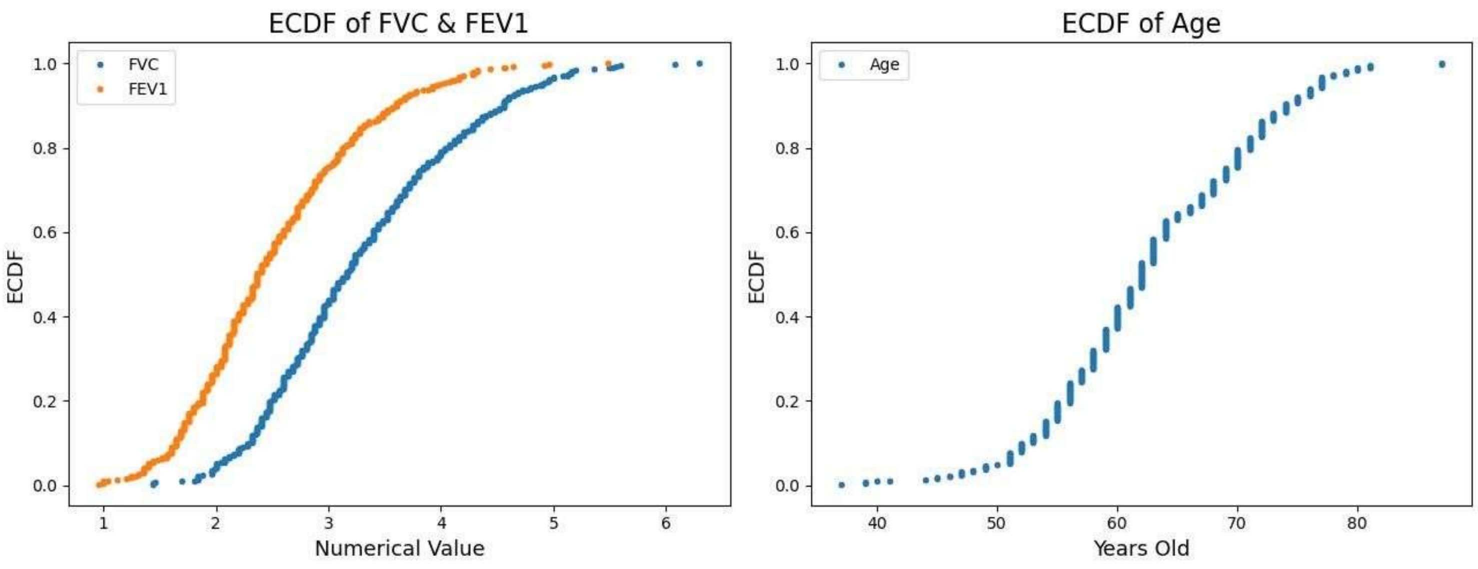


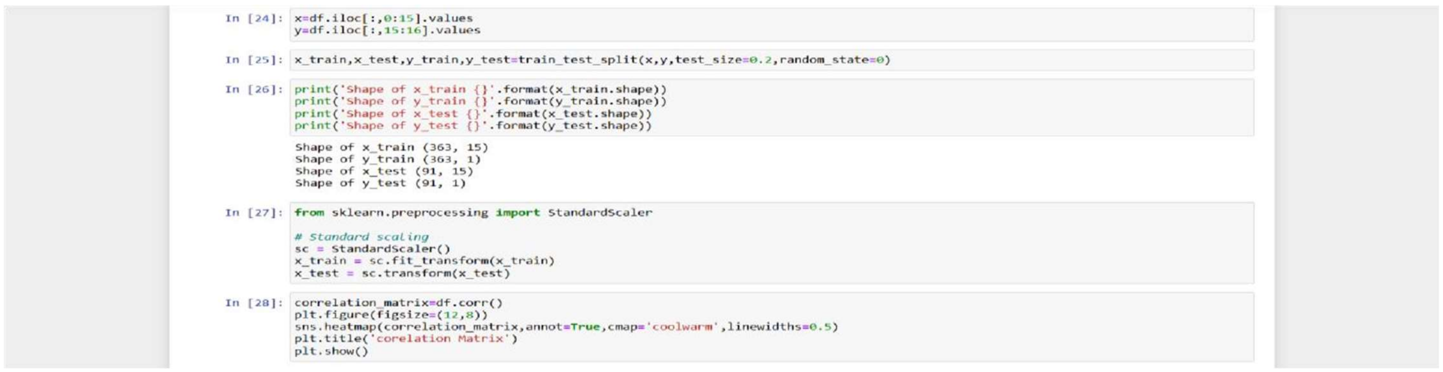


```
In [19]: #Correlation coefficients for FVC and FEV1
         np.corrcoef(df.FVC,df.FEV1)[0,1]

Out[19]: 0.8875452733829001

In [20]: #Correlation coefficients for Age and FVC
         np.corrcoef(df.Age,df.FVC)[0,1]

Out[20]: -0.2994299196604911

In [21]: #correlation coefficients for Age and FEV1
         np.corrcoef(df.Age,df.FEV1)[0,1]

Out[21]: -0.30961662730798917

In [22]: #Correlations of FVC,FEV1 and Age
         def ecdf(data):
             n=len(data)
             x=np.sort(data)
             y=np.arange(1,n+1)/n
             return x,y
```

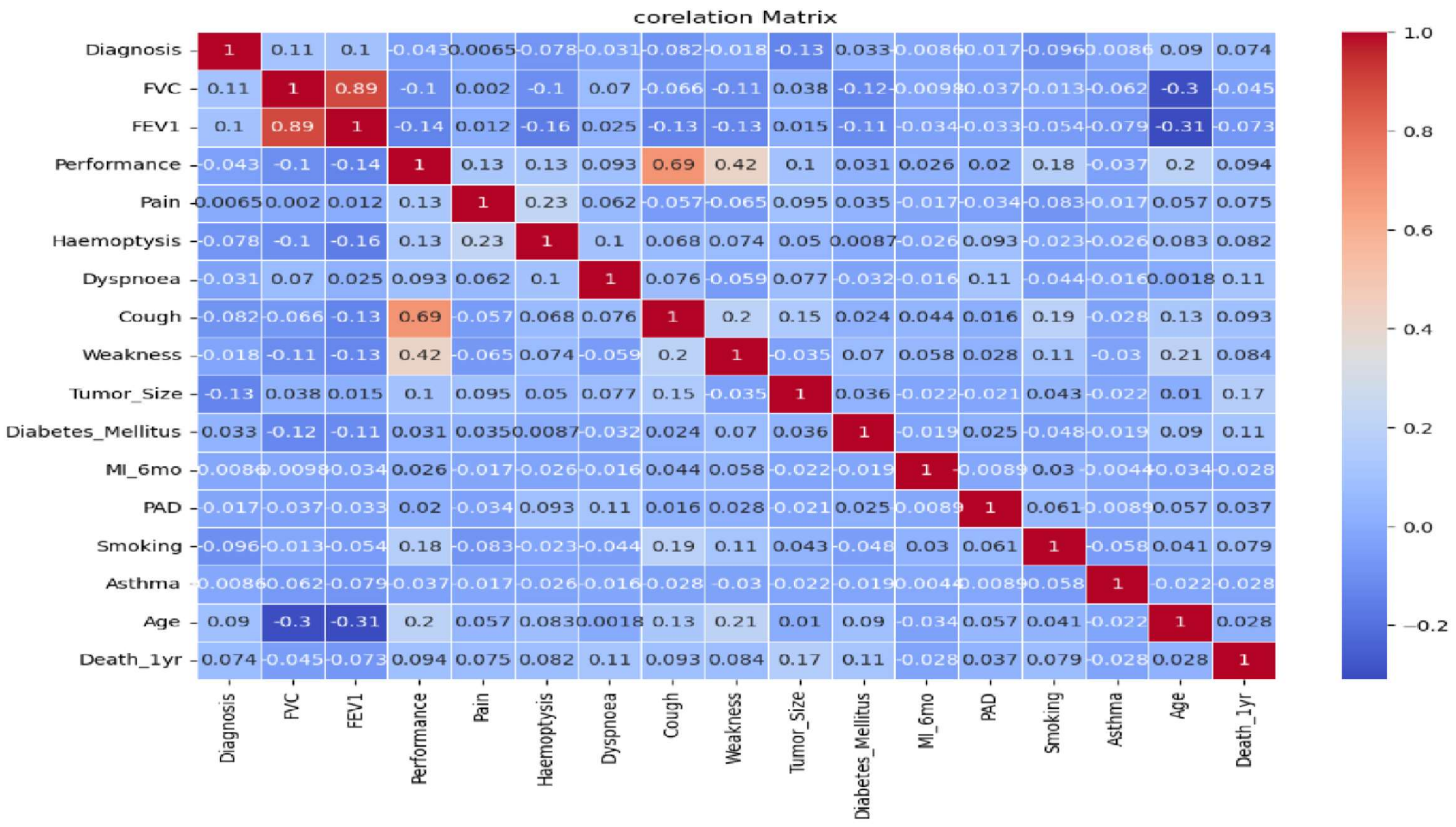


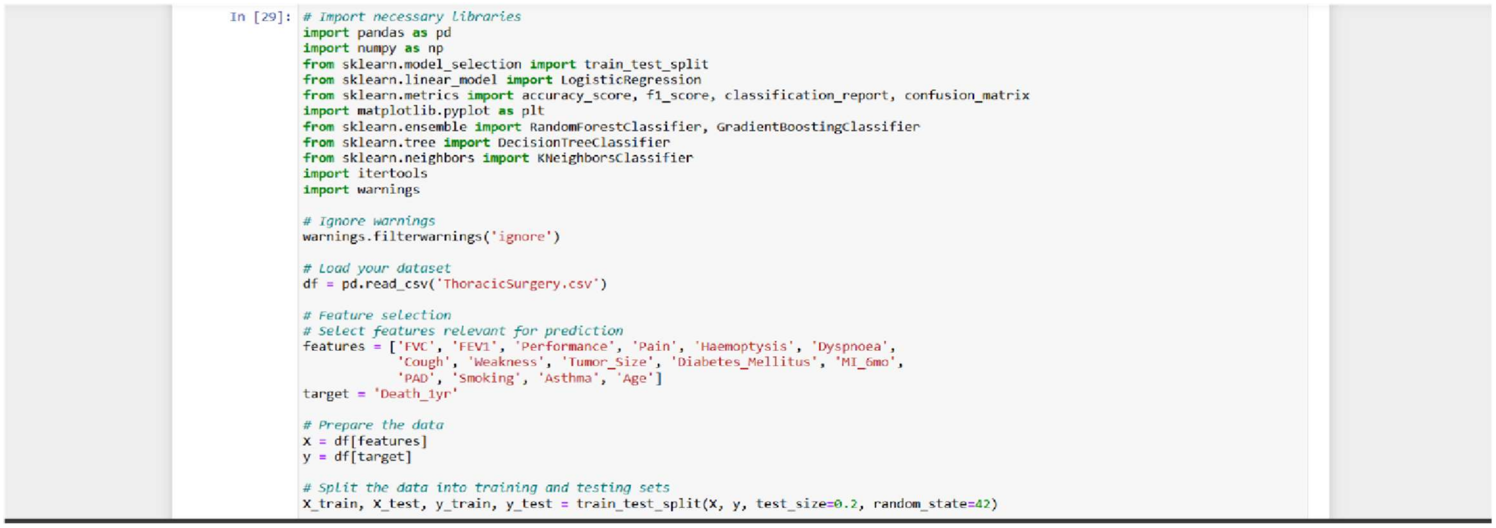
```
In [23]: #ECDF of FVC,FEV1,Age
         x_fvc,y_fvc=ecdf(df.FVC)
         x_fev1,y_fev1=ecdf(df.FEV1)
         x_age,y_age=ecdf(df.Age)

         fig,axes=plt.subplots(1,2,figsize=(13,5))
         axes[0].plot(x_fvc,y_fvc,marker='.',linestyle='none',label='FVC')
         axes[0].plot(x_fev1,y_fev1,marker='.',linestyle='none',label='FEV1')

         axes[0].set_xlabel('Numerical Value',fontsize=13)
         axes[0].set_ylabel('ECDF',fontsize=13)
         axes[0].legend(loc='upper left')
         axes[0].set_title('ECDF of FVC & FEV1',fontsize=16)

         axes[1].plot(x_age,y_age,marker='.',linestyle='none',label='Age')
         axes[1].set_xlabel('Years Old',fontsize=13)
         axes[1].set_ylabel('ECDF',fontsize=13)
         axes[1].legend(loc='upper left')
         axes[1].set_title('ECDF of Age',fontsize=16)
         plt.tight_layout()
```



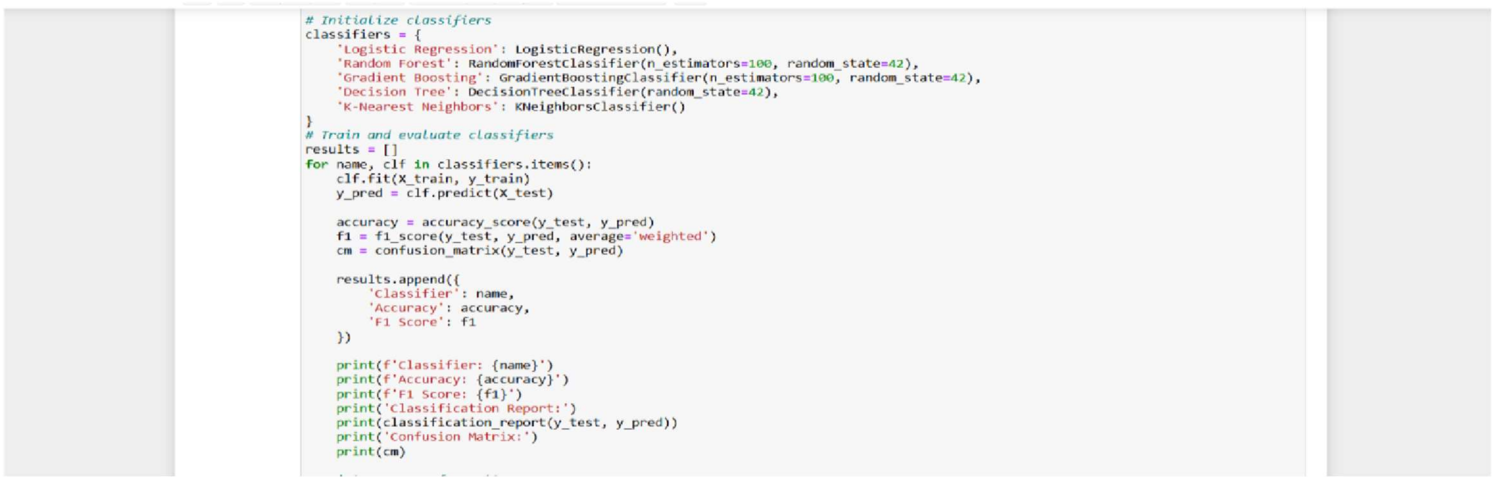


```
In [24]: x=df.iloc[:,0:15].values
         y=df.iloc[:,15:16].values

In [25]: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=0)

In [26]: print('Shape of x_train {}'.format(x_train.shape))
         print('Shape of y_train {}'.format(y_train.shape))
         print('Shape of x_test {}'.format(x_test.shape))
         print('Shape of y_test {}'.format(y_test.shape))

         Shape of x_train (363, 15)
         Shape of y_train (363, 1)
         Shape of x_test (91, 15)
         Shape of y_test (91, 1)

In [27]: from sklearn.preprocessing import StandardScaler

         # Standard scaling
         sc = StandardScaler()
         x_train = sc.fit_transform(x_train)
         x_test = sc.transform(x_test)

In [28]: correlation_matrix=df.corr()
         plt.figure(figsize=(12,8))
         sns.heatmap(correlation_matrix,annot=True,cmap='coolwarm',linewidths=0.5)
         plt.title('corelation Matrix')
         plt.show()
```

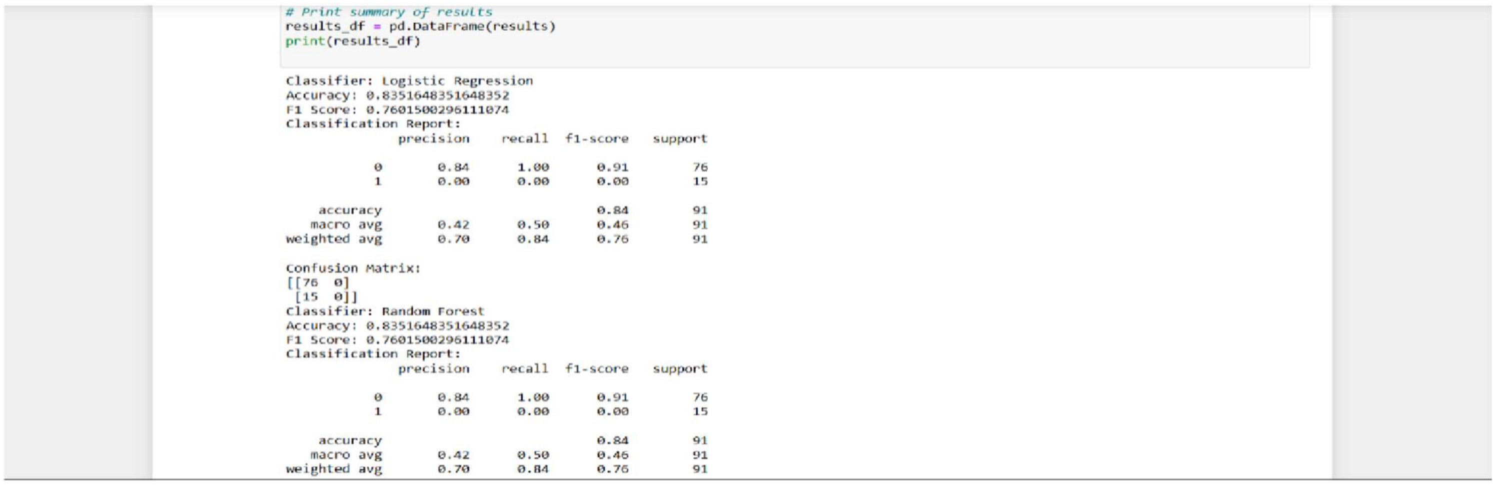




```
In [29]: # Import necessary libraries
         import pandas as pd
         import numpy as np
         from sklearn.model_selection import train_test_split
         from sklearn.linear_model import LogisticRegression
         from sklearn.metrics import accuracy_score, f1_score, classification_report, confusion_matrix
         import matplotlib.pyplot as plt
         from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
         from sklearn.tree import DecisionTreeClassifier
         from sklearn.neighbors import KNeighborsClassifier
         import itertools
         import warnings

         # Ignore warnings
         warnings.filterwarnings('ignore')

         # Load your dataset
         df = pd.read_csv('ThoracicSurgery.csv')

         # Feature selection
         # Select features relevant for prediction
         features = ['FVC', 'FEV1', 'Performance', 'Pain', 'Haemoptysis', 'Dyspnoea',
                     'Cough', 'Weakness', 'Tumor_Size', 'Diabetes_Mellitus', 'MI_6mo',
                     'PAD', 'Smoking', 'Asthma', 'Age']
         target = 'Death_1yr'

         # Prepare the data
         X = df[features]
         y = df[target]

         # Split the 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)
```

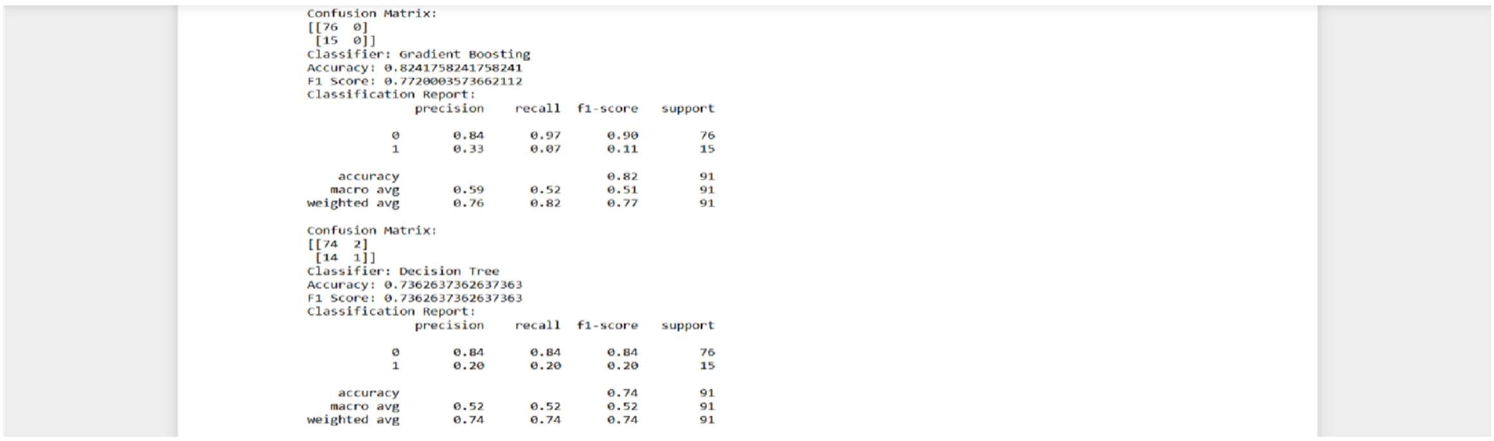


```python
# Initialize classifiers
classifiers = {
    'Logistic Regression': LogisticRegression(),
    'Random Forest': RandomForestClassifier(n_estimators=100, random_state=42),
    'Gradient Boosting': GradientBoostingClassifier(n_estimators=100, random_state=42),
    'Decision Tree': DecisionTreeClassifier(random_state=42),
    'K-Nearest Neighbors': KNeighborsClassifier()
}
# Train and evaluate classifiers
results = []
for name, clf in classifiers.items():
    clf.fit(X_train, y_train)
    y_pred = clf.predict(X_test)

    accuracy = accuracy_score(y_test, y_pred)
    f1 = f1_score(y_test, y_pred, average='weighted')
    cm = confusion_matrix(y_test, y_pred)

    results.append({
        'Classifier': name,
        'Accuracy': accuracy,
        'F1 Score': f1
    })

    print(f'Classifier: {name}')
    print(f'Accuracy: {accuracy}')
    print(f'F1 Score: {f1}')
    print('Classification Report:')
    print(classification_report(y_test, y_pred))
    print('Confusion Matrix:')
    print(cm)
```

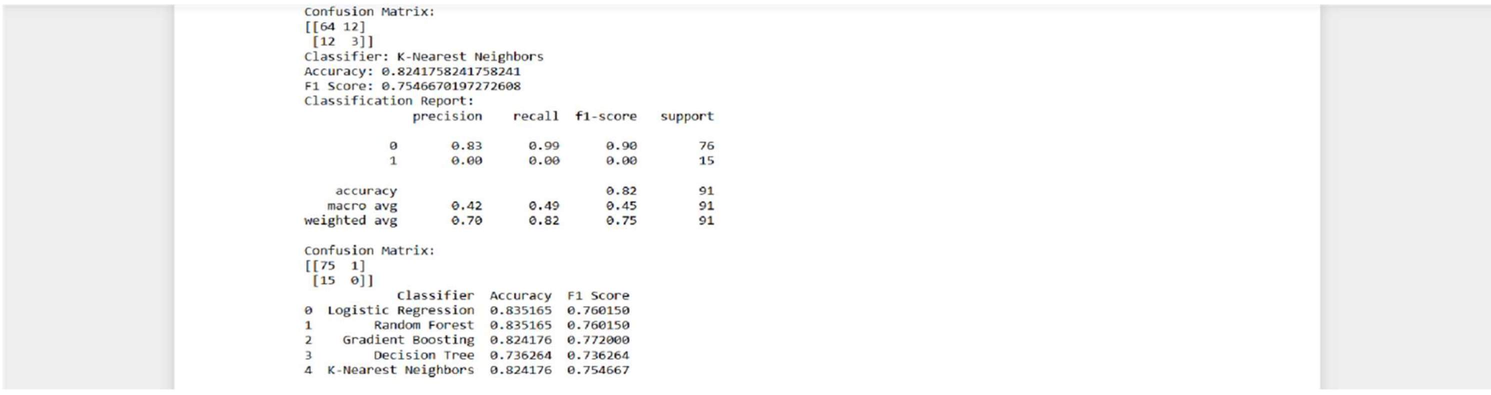


```python
# Print summary of results
results_df = pd.DataFrame(results)
print(results_df)
```

```
Classifier: Logistic Regression
Accuracy: 0.8351648351648352
F1 Score: 0.7601500296111074
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        76
           1       0.00      0.00      0.00        15

    accuracy                           0.84        91
   macro avg       0.42      0.50      0.46        91
weighted avg       0.70      0.84      0.76        91

Confusion Matrix:
[[76  0]
 [15  0]]
Classifier: Random Forest
Accuracy: 0.8351648351648352
F1 Score: 0.7601500296111074
Classification Report:
              precision    recall  f1-score   support

           0       0.84      1.00      0.91        76
           1       0.00      0.00      0.00        15

    accuracy                           0.84        91
   macro avg       0.42      0.50      0.46        91
weighted avg       0.70      0.84      0.76        91
```



```
Confusion Matrix:
[[76  0]
 [15  0]]
Classifier: Gradient Boosting
Accuracy: 0.8241758241758241
F1 Score: 0.7720003573662112
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.97      0.90        76
           1       0.33      0.07      0.11        15

    accuracy                           0.82        91
   macro avg       0.59      0.52      0.51        91
weighted avg       0.76      0.82      0.77        91

Confusion Matrix:
[[74  2]
 [14  1]]
Classifier: Decision Tree
Accuracy: 0.7362637362637363
F1 Score: 0.7362637362637363
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.84      0.84        76
           1       0.20      0.20      0.20        15

    accuracy                           0.74        91
   macro avg       0.52      0.52      0.52        91
weighted avg       0.74      0.74      0.74        91
```



```
Confusion Matrix:
[[64 12]
 [12  3]]
Classifier: K-Nearest Neighbors
Accuracy: 0.8241758241758241
F1 Score: 0.7546670197272608
Classification Report:
              precision    recall  f1-score   support

           0       0.83      0.99      0.90        76
           1       0.00      0.00      0.00        15

    accuracy                           0.82        91
   macro avg       0.42      0.49      0.45        91
weighted avg       0.70      0.82      0.75        91

Confusion Matrix:
[[75  1]
 [15  0]]
            Classifier  Accuracy  F1 Score
0  Logistic Regression  0.835165  0.760150
1        Random Forest  0.835165  0.760150
2    Gradient Boosting  0.824176  0.772000
3        Decision Tree  0.736264  0.736264
4  K-Nearest Neighbors  0.824176  0.754667
```