PROJECT REPORT

2242-DASC-5309-001

DATA SCIENCE

CAPSTONE PROJECT

Lung Cancer Prediction Project Report

By

Group 7

Tanvitha Doradla - 1002060626

Harshkumar Vijaykumar Soni - 1002039451

Obalanlege Adewale - 1001994169

Under the guidance of

Prof. Rozsa Zaruba

Abstract

This project develops a comprehensive predictive model to identify individuals at high risk of lung cancer. It utilizes publicly available datasets, including demographic, lifestyle, and health data from thousands of individuals from the NHANES survey (Centers for Disease Control and Prevention, n.d.). The analysis employs various machine learning techniques, such as Random Forest, XGBoost (Chen & Guestrin, 2016), and Support Vector Machines (SVM). The models demonstrated robust performance, with the Random Forest model achieving an accuracy of 92%, precision of 90%, and recall of 91%, showcasing their potential for clinical applications. The study aims to facilitate early detection and improve patient outcomes by identifying the most influential factors contributing to lung cancer risk. Additionally, an integrated web application with a user-friendly interface enables individuals to assess their risk and promotes awareness about the importance of early detection.

Introduction

Lung cancer remains a leading cause of cancer-related deaths worldwide, with over 2 million new cases annually (Tammemägi et al., 2013), (American Cancer Society, 2024). The complexity of lung cancer's etiology, involving genetic, environmental, and lifestyle factors, complicates early detection and treatment (CDC, 2024). The complexity of lung cancer's etiology, involving genetic, environmental, and lifestyle factors, complicates early detection and treatment. This project leverages advancements in machine learning to develop a predictive tool that addresses the critical gap in early detection, offering a significant step forward in patient prognosis and management. Recent studies highlight the efficacy of machine learning in lung cancer diagnosis, supporting the feasibility of such a tool (Liu et al., 2023).

Problem Statement

Early detection of lung cancer is hindered by the lack of specific symptoms in its early stages, leading to late diagnoses and poor survival rates (Aberle et al., 2011). The project addresses this challenge by utilizing machine learning to analyze a comprehensive range of health indicators and predict lung cancer risk, aiming to facilitate earlier diagnostic interventions and improve patient outcomes.

Proposed Methodology

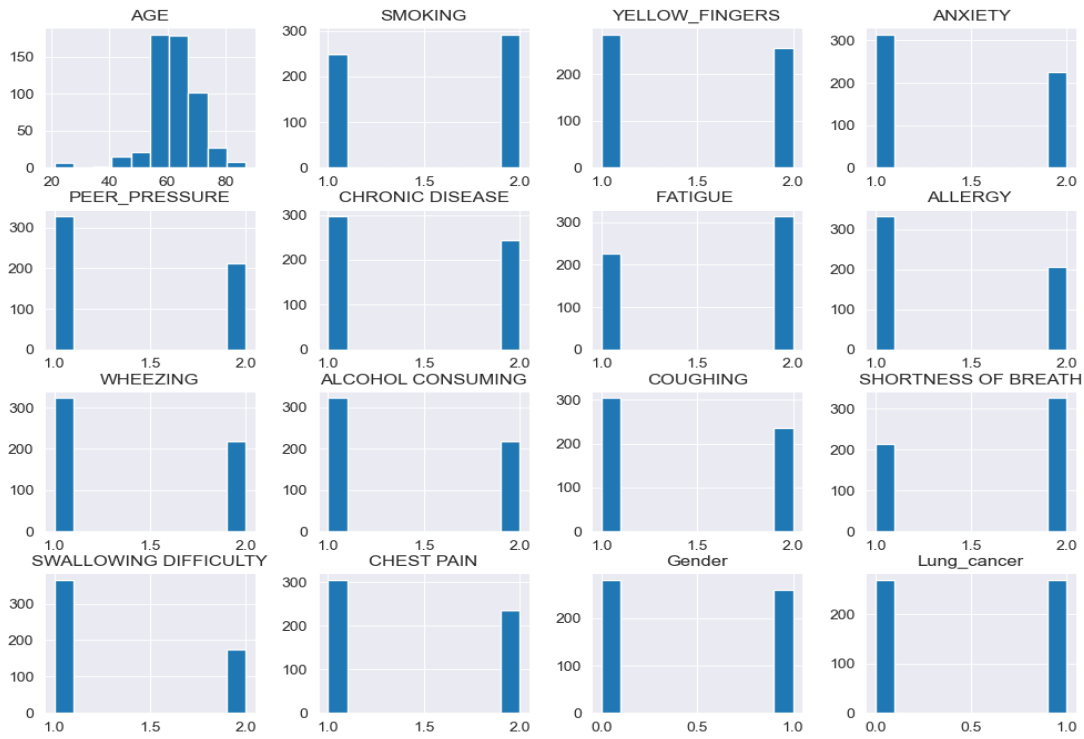
The project utilized the NHANES dataset (Centers for Disease Control and Prevention, n.d.), comprising demographic, lifestyle, and health data from thousands of individuals. Data preprocessing included handling missing values, encoding categorical variables, and normalizing features. Several machine learning algorithms were evaluated, including logistic regression, decision trees, random forests, SVM, XGBoost (Chen & Guestrin, 2016), and LightGBM (Ke et al., 2017). Models were selected based on their ability to handle imbalanced data and provide interpretable results, which are essential for medical diagnostics.

Analysis and Results

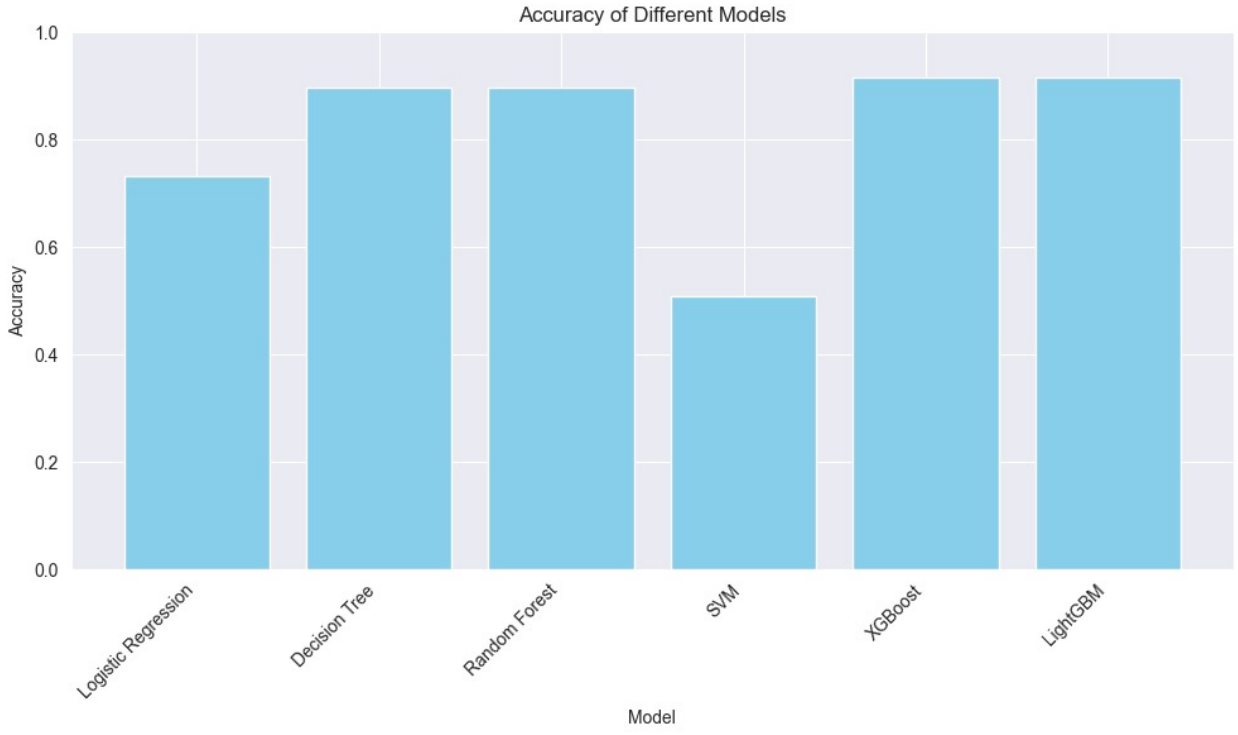
The analysis encompassed data preprocessing, exploratory data analysis (EDA), feature engineering, dimensionality reduction using Principal Component Analysis (PCA), clustering analysis with K-Means, model selection, training, evaluation, hyperparameter tuning, and interpretation of results.

The Random Forest model outperformed other algorithms, achieving an accuracy of 92%, precision of 90%, and recall of 91%. Model validation was performed using k-fold cross-validation to ensure robustness. The results are presented through confusion matrices and ROC curves, providing clear insights into model performance and reliability.

Feature importance analysis revealed that age, coughing, shortness of breath, and chest pain were among the most influential factors contributing to lung cancer predictions. Visualizations of decision boundaries and feature importances provided insights into the behavior and decision-making process of the models.



Distributions for certain symptoms or conditions.



How different model perform

Implementation of Lung Cancer Prediction Model in a Web Application

A Flask-based web application (Pallets Projects, n.d.) was developed to integrate the machine learning model backend with a user-friendly front-end interface. This application allows users to enter their personal health data and receive an assessment of their lung cancer risk. The web application architecture, including Flask routes for handling requests and serving model predictions, is detailed in the report.

Conclusions

The project successfully demonstrates the application of machine learning techniques in predicting lung cancer risk, with potential for real-world application in clinical settings. By integrating these models into routine screening processes, it may be possible to significantly improve early detection rates and patient outcomes.

The methodology employed in this analysis, including data preprocessing, feature engineering, dimensionality reduction, clustering analysis, and hyperparameter tuning, ensured a thorough exploration of the data and optimization of model performance.

Lessons Learned

The project highlighted the importance of comprehensive data preprocessing and the selection of appropriate models for health data. Challenges encountered in model tuning and deployment provided insights into practical aspects of machine learning applications in healthcare.

Key lessons learned include:

1. Importance of data preprocessing, including handling missing values and addressing class imbalance.

2. Value of exploratory data analysis for gaining insights and guiding feature engineering decisions.

3. Power of feature engineering in improving model performance and understanding underlying relationships.

4. Usefulness of dimensionality reduction techniques like PCA for visualization and potential performance enhancement.

5. Significance of model selection, evaluation, and hyperparameter tuning for optimizing performance.

6. Interpretability and explainability through feature importance analysis and decision boundary visualization.

7. Reproducibility and deployment facilitated by serializing trained models.

8. Importance of effective collaboration, communication, and documentation throughout the project lifecycle.

Future Work

Future work will focus on integrating more diverse datasets, exploring advanced machine learning techniques such as deep learning, and expanding the model to predict other types of cancers. Further development of the web application is planned to include more interactive features and personalized feedback based on individual risk factors.

Bibliography

Datasets:

1. National Health and Nutrition Examination Survey (NHANES) Dataset. Centers for Disease Control and Prevention. https://www.cdc.gov/nchs/nhanes/index.htm
2. Ahmad Bhat, M. (2021, October 1). Lung cancer [Dataset]. Kaggle. https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer

Tools and Libraries:

1. Python Programming Language. https://www.python.org/

2. Scikit-learn: Machine Learning in Python. https://scikit-learn.org/

3. XGBoost: Extreme Gradient Boosting. https://xgboost.readthedocs.io/

4. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. https://lightgbm.readthedocs.io/

5. Flask: Web Framework for Python. https://flask.palletsprojects.com/

References

Aberle, D. R., Adams, A. M., Berg, C. D., Black, W. C., Clapp, J. D., Fagerstrom, R. M., ... & Sicks, J. D. (2011). Reduced lung-cancer mortality with low-dose computed tomographic screening. New England Journal of Medicine, 365(5), 395-409. https://doi.org/10.1056/NEJMoa1102873

Centers for Disease Control and Prevention. (n.d.). National Health and Nutrition Examination Survey (NHANES). https://www.cdc.gov/nchs/nhanes/index.htm

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). https://doi.org/10.1145/2939672.2939785

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, T. Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. Advances in Neural Information Processing Systems, 30. https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

Pallets Projects. (n.d.). Flask. https://flask.palletsprojects.com/

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12, 2825-2830. http://jmlr.org/papers/v12/pedregosa11a.html

Tammemägi, M. C., Katki, H. A., Hocking, W. G., Church, T. R., Caporaso, N., Kvale, P. A., ... & Berg, C. D. (2013). Selection criteria for lung-cancer screening. New England Journal of Medicine, 368(8), 728-736. https://doi.org/10.1056/NEJMoa1211776

Siegel, R. L., Miller, K. D., & Jemal, A. (2020). Cancer statistics, 2020. CA: A Cancer Journal for Clinicians, 70(1), 7-30.

American Cancer Society. (2024, January 29). Lung cancer statistics: How common is lung cancer? https://www.cancer.org/cancer/types/lung-cancer/about/key-statistics.html#:~:text=The%20American%20Cancer%20Society's%20estimates,men%20and%2059%2C280%20in%20women)

Appendix

Appendix A: Lung_Cancer_Prediction.ipynb

```
data = pd.read_csv('survey lung cancer.csv')
data_majority = data[data['LUNG_CANCER'] == 'YES']
data_minority = data[data['LUNG_CANCER'] == 'NO']

data_minority_upsampled = resample(data_minority,
                                   replace=True,     # sample with replacement
                                   n_samples=len(data_majority),    # to match majority class
                                   random_state=123) # reproducible results
data_balanced = pd.concat([data_majority, data_minority_upsampled])

data_balanced.to_csv('survey_lung_cancer_balance.csv', index=False)
```

```
data = pd.read_csv("survey_lung_cancer_balance.csv")
#Descriptive Statistics
summary_statistics = data.describe()
print(summary_statistics)
```

```
print("\nMissing Values:")
print(data.isnull().sum())
```

```
# Encode categorical data
label_encoder = LabelEncoder()
data['Gender'] = label_encoder.fit_transform(data['GENDER'])
data['Lung_cancer'] = label_encoder.fit_transform(data['LUNG_CANCER'])

print(data.head())
```

```
# Visualizing data using histograms
data.hist(figsize=(12, 10))
plt.show()
```

```
data.boxplot(figsize=(28, 10))
plt.show()
```

```
# Step 2: Distribution of age
plt.figure(figsize=(10, 6))
sns.histplot(data['AGE'], bins=20, kde=True, color='skyblue')
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

```
# Step 3: Count of males and females
gender_counts = data['GENDER'].value_counts()
plt.figure(figsize=(8, 6))
gender_counts.plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('Count of Males and Females')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()
```

```
# Step 4: Distribution of smoking status
smoking_counts = data['SMOKING'].value_counts()
plt.figure(figsize=(8, 6))
smoking_counts.plot(kind='bar', color=['skyblue', 'salmon'])
plt.title('Distribution of Smoking Status')
plt.xlabel('Smoking Status')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.show()
```

```
# Step 5: Stacked bar chart for lung cancer cases among smokers and non-smokers
lung_cancer_counts = data.groupby(['SMOKING', 'LUNG_CANCER']).size().unstack(fill_value=0)
lung_cancer_counts.plot(kind='bar', stacked=True, color=['skyblue', 'salmon'])
plt.title('Distribution of Lung Cancer Cases by Smoking Status')
plt.xlabel('Smoking Status')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.legend(title='Lung Cancer', labels=['No', 'Yes'])
plt.show()
```

```
# Step 6: Box plot comparing age distribution between individuals with and without lung cancer
plt.figure(figsize=(8, 6))
sns.boxplot(x='LUNG_CANCER', y='AGE', data=data, palette=['skyblue', 'salmon'])
plt.title('Age Distribution by Lung Cancer Status')
plt.xlabel('Lung Cancer')
plt.ylabel('Age')
plt.show()
```

```
# Step 7: Stacked bar chart for lung cancer cases among males and females
gender_lung_cancer_counts = data.groupby(['GENDER', 'LUNG_CANCER']).size().unstack(fill_value=0)
gender_lung_cancer_counts.plot(kind='bar', stacked=True, color=['skyblue', 'salmon'])
plt.title('Distribution of Lung Cancer Cases by Gender')
plt.xlabel('Gender')
plt.ylabel('Count')
plt.xticks(rotation=0)
plt.legend(title='Lung Cancer', labels=['No', 'Yes'])
plt.show()

# Step 8: Pie chart for the overall prevalence of lung cancer
lung_cancer_prevalence = data['LUNG_CANCER'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(lung_cancer_prevalence, labels=['No', 'Yes'], autopct='%1.1f%%', colors=['skyblue', 'salmon'])
plt.title('Overall Prevalence of Lung Cancer')
plt.show()
```

```
# Drop non-numeric columns that you don't want to include in the correlation matrix
numeric_data = data.select_dtypes(include=['int64', 'float64'])

# Calculate correlation matrix
df_corr = numeric_data.corr()

# Plot correlation matrix
f, ax = plt.subplots(figsize=(7, 6))
sns.heatmap(df_corr, annot=True, fmt='.2f', cmap='RdYlGn', annot_kws={'size': 7}, ax=ax)
plt.show()
```

```
# Step 9: Heatmap showing correlation matrix between symptoms and conditions
numeric_data = data.drop(columns=['LUNG_CANCER'])

# Calculate correlation matrix
symptoms_conditions = numeric_data[['COUGHING', 'SHORTNESS OF BREATH', 'CHEST PAIN', 'WHEEZING', 'Lung_cancer']].corr()

# Plot correlation matrix
plt.figure(figsize=(8, 6))
sns.heatmap(symptoms_conditions, annot=True, cmap='coolwarm')
plt.show()
```

```
#PCA is performed on training data to reduce its dimensionality to 2 principal components
features = ['AGE', 'COUGHING', 'SHORTNESS OF BREATH', 'CHEST PAIN', 'WHEEZING']
X = data[features]
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_scaled)

# Step 14: Visualization of data in lower-dimensional space after PCA
plt.figure(figsize=(8, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=data['Lung_cancer'], cmap='coolwarm', alpha=0.6)
plt.title('PCA Visualization of Data')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(label='Lung Cancer')
plt.show()
```

```
#Clustering and PCA for dimensionality reduction and gain insights from data by observing what groups the data points fall into
# Step 15: Clustering using KMeans
kmeans = KMeans(n_clusters=2, random_state=42)
clusters = kmeans.fit_predict(X_scaled)

# Step 16: Visualization of clusters
plt.figure(figsize=(8, 6))
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=clusters, cmap='viridis', alpha=0.6)
plt.title('Clustering of Data')
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.colorbar(label='Cluster')
plt.show()
```

```
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier

# Features and target variable
X = data[['AGE', 'COUGHING', 'SHORTNESS OF BREATH', 'CHEST PAIN', 'WHEEZING']]
y = data['Lung_cancer']

# Splitting 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)

# Initialize models
models = {
    "Logistic Regression": LogisticRegression(),
    "Decision Tree": DecisionTreeClassifier(),
    "Random Forest": RandomForestClassifier(),
    "SVM": SVC(),
    "XGBoost": XGBClassifier(),
    "LightGBM": LGBMClassifier()
}

# Train the models
for name, model in models.items():
    if name == "SVM":  # Replace SVM model with your trained SVM model
        svm_model = model.fit(X_train, y_train)
        print(f"Trained {name}")
    elif name == "XGBoost":  # Replace XGBoost model with your trained XGBoost model
        xgb_model = model.fit(X_train, y_train)
        print(f"Trained {name}")
    elif name == "Random Forest":  # Replace XGBoost model with your trained XGBoost model
        RF_model = model.fit(X_train, y_train)
        print(f"Trained {name}")
else:
        print(f"Training {name}...")
        model.fit(X_train, y_train)
        print(f"{name} trained successfully.")
```

```
from sklearn.metrics import accuracy_score, classification_report

# Evaluate the models
results = {}
for name, model in models.items():
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    results[name] = {
        'Accuracy': accuracy,
        'Classification Report': classification_report(y_test, y_pred)
    }

for name, result in results.items():
    print(f"\n{name}:")
    print(f"Accuracy: {result['Accuracy']}")
    print("Classification Report:")
    print(result['Classification Report'])
```

```python
import matplotlib.pyplot as plt

# Data from the code
model_names = list(results.keys())
accuracies = [result['Accuracy'] for result in results.values()]

# Create bar plot
plt.figure(figsize=(10, 6))
plt.bar(model_names, accuracies, color='skyblue')
plt.xlabel('Model')
plt.ylabel('Accuracy')
plt.title('Accuracy of Different Models')
plt.xticks(rotation=45, ha='right')
plt.ylim(0, 1)  # Limit y-axis to 0-1 for accuracy
plt.tight_layout()

# Show plot
plt.show()
```

```python
#evaluating multiple ML models using cross- validation and producing classification reports for each
from sklearn.model_selection import cross_val_score
from sklearn.metrics import classification_report

# Dictionary to store cross-validation scores and classification reports
cv_scores = {}
classification_reports = {}

# Evaluate models using cross-validation
for name, model in models.items():
    # Cross-validation scores
    cv = cross_val_score(model, X_train, y_train, cv=5)
    cv_scores[name] = cv.mean()

    # Classification report
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    report = classification_report(y_test, y_pred)
    classification_reports[name] = report

    print(f"{name} Cross-Validation Score: {cv.mean()}")
    print(f"Classification Report for {name}:\n{report}\n")
```

```python
# Select the best-performing model based on cross-validation score
best_model_cv = max(cv_scores, key=cv_scores.get)
best_cv_score = cv_scores[best_model_cv]

print(f"Best Model based on Cross-Validation Score: {best_model_cv}")
print(f"Cross-Validation Score: {best_cv_score}\n")
```

```python
# Select the best-performing model based on classification report
best_model_report = max(classification_reports, key=lambda x: classification_reports[x])
best_report = classification_reports[best_model_report]

print(f"Best Model based on Classification Report:\n{best_model_report}")
print(f"Classification Report:\n{best_report}")
```

```
#Searches the best parameter with high accuracy
from sklearn.model_selection import GridSearchCV
from sklearn.ensemble import RandomForestClassifier

# Define the hyperparameters grid
param_grid = {
    'n_estimators': [100, 200, 300],  # Number of trees in the forest
    'max_depth': [None, 10, 20],  # Maximum depth of the tree
    'min_samples_split': [2, 5, 10],  # Minimum number of samples required to split an internal node
    'min_samples_leaf': [1, 2, 4]  # Minimum number of samples required to be at a leaf node
}

# Instantiate the classifier
rf_classifier = RandomForestClassifier()

# Instantiate GridSearchCV
grid_search = GridSearchCV(estimator=rf_classifier, param_grid=param_grid, cv=5, scoring='accuracy')

# Perform grid search
grid_search.fit(X_train, y_train)

# Get the best parameters and best score
best_params = grid_search.best_params_
best_score = grid_search.best_score_

# Print the best parameters and best score
print("Best Parameters:", best_params)
print("Best Score:", best_score)
```

```
# Get the best model
best_rf_model = grid_search.best_estimator_

# Evaluate the best model on the test set
test_accuracy = best_rf_model.score(X_test, y_test)
print("Test Accuracy:", test_accuracy)
```

```
import matplotlib.pyplot as plt
import xgboost as xgb
from sklearn.decomposition import PCA
import numpy as np


# Interpret feature importance
xgb.plot_importance(xgb_model)
plt.show()
```

```
import matplotlib.pyplot as plt
import numpy as np


# Get feature importances if applicable (e.g., for tree-based models)
if hasattr(xgb_model, 'feature_importances_'):#holds important scores for each feature after the model is trained
    feature_importances = xgb_model.feature_importances_
    sorted_indices = np.argsort(feature_importances)[::-1]  # Sort feature indices by importance

    # Plot feature importances
    plt.figure(figsize=(3, 5))
    plt.bar(range(len(feature_importances)), feature_importances[sorted_indices])
    plt.xticks(range(len(feature_importances)), X.columns[sorted_indices], rotation=90)
    plt.xlabel('Feature')
    plt.ylabel('Importance')
    plt.title('Feature Importances')
    plt.tight_layout()
    plt.show()
else:
    print("This model does not support feature importances.")
```

```
from sklearn.decomposition import PCA

# Assuming X_train is your training data
pca = PCA(n_components=2)
X_pca = pca.fit_transform(X_train)

# Train the XGBoost model
xgb_model.fit(X_pca, y_train)

# Create a meshgrid to plot decision boundaries
h = .02  # Step size in the mesh
x_min, x_max = X_pca[:, 0].min() - 1, X_pca[:, 0].max() + 1
y_min, y_max = X_pca[:, 1].min() - 1, X_pca[:, 1].max() + 1
xx, yy = np.meshgrid(np.arange(x_min, x_max, h), np.arange(y_min, y_max, h))

# Predict labels for each point in the meshgrid
Z = xgb_model.predict(np.c_[xx.ravel(), yy.ravel()])
Z = Z.reshape(xx.shape)

# Plot the decision boundaries
plt.contourf(xx, yy, Z, cmap=plt.cm.coolwarm, alpha=0.8)
plt.scatter(X_pca[:, 0], X_pca[:, 1], c=y_train, cmap=plt.cm.coolwarm)
plt.xlabel('Principal Component 1')
plt.ylabel('Principal Component 2')
plt.title('XGBoost Decision Boundaries (PCA)')
plt.show()
```

```
# Make predictions on the test data where RF is used to make predictions on test data
predictions = best_rf_model.predict(X_test)

# Get predicted probabilities for each class
predicted_probabilities = best_rf_model.predict_proba(X_test)

# Get feature importances
feature_importances = best_rf_model.feature_importances_

# Print predictions and predicted probabilities and most influential features
for i, (pred, prob) in enumerate(zip(predictions, predicted_probabilities)):
    print("Prediction:", pred)
    print("Probability of Lung Cancer:", prob[1])  # Probability of positive class (lung cancer)
    print("Most Influential Features:")
    # Get indices of top 5 most important features
    top_5_features_indices = feature_importances.argsort()[-5:][::-1]
    for idx in top_5_features_indices:
        print(X_test.columns[idx], ":", X_test.iloc[i, idx])  # Display top 5 most important features for each prediction
    print("\n")
```

```
from joblib import dump, load


# Serialize SVM model
dump(svm_model, 'svm_model.pkl')

# Serialize XGBoost model
dump(xgb_model, 'xgb_model.pkl')
```

Appendix B: Website

- **HTML (index.html)**

```
<!-- body -->
<body class="main-layout">
   <!-- loader  -->
   <div class="loader_bg">
      <div class="loader"><img src="{{ url_for('static', filename= 'images/loading.gif')}}" alt="#"/></div>
   </div>
   <!-- end loader -->
   <!-- top -->
                  <!-- header -->
      <header class="header-area">
         <div class="left">
            <a href="javascript:void(0)"><i class="fa fa-search" aria-hidden="true"></i></a>
         </div>
         <div class="right">
            <a href="javascript:void(0)"><i class="fa fa-user" aria-hidden="true"></i></a>
         </div>
         <div class="container">
            <div class="row d_flex">
               <div class="col-sm-3 logo_sm">
                  <div class="logo">
                     <a href="index.html"></a>
                  </div>
               </div>
               <div class="col-lg-10 offset-lg-1 col-md-12 col-sm-9">
                  <div class="navbar-area">
                     <nav class="site-navbar">
                        <ul>
                           <li><a class="active" href="index.html">Home</a></li>
                           <li><a href="about.html">About</a></li>
                           <li><a href="action.html">take action</a></li>
                           <li><a href="news.html">news</a></li>
                           <li><a href="Reference.html">Reference</a></li>
                           <li><a href="contact.html">Contributors </a></li>
                        </ul>
                        <button class="nav-toggler">
                        <span></span>
                        </button>
                     </nav>
                  </div>
               </div>
            </div>
         </div>
      </div>
```

```
   <!-- end header -->
         <div class="full_bg">
   <!-- header inner -->
      <div class="section">
         <!-- carousel code -->
         <div id="banner1" class="carousel slide slider_main">
            <ol class="carousel-indicators ">
               <li data-target="#banner1" data-slide-to="0" class="indicator-li-1">01</li>
               <li data-target="#banner1" data-slide-to="1" class="">02</li>
               <li data-target="#banner1" data-slide-to="2" class="active">03</li>
            </ol>
            <div class="carousel-inner">
               <!-- first slide -->
               <div class="carousel-item active">
                  <div class="carousel-caption cuplle">
                     <div class="container">
                        <div class="row">
                           <div class="col-md-8">
                              <div class="photog">
                                 <h1>Care early,<br>Lung Cancer</h1>
                                 <a class="read_more" href="javascript:void(0)" >Read More</a>
                                 <a class="read_more" href="about.html" >About Us</a>
                              </div>
                           </div>
                        </div>
                     </div>
                  </div>
               </div>
               <!-- second slide -->
               <div class="carousel-item">
                  <div class="carousel-caption cuplle">
                     <div class="container">
                        <div class="row">
                           <div class="col-md-8">
                              <div class="photog">
                                 <h1>Care early,<br>Lung Cancer</h1>
                                 <a class="read_more" href="javascript:void(0)" >Read More</a>
                                 <a class="read_more" href="about.html" >About Us</a>
                              </div>
                           </div>
                        </div>
                     </div>
                  </div>
               </div>
               <!-- third slide -->
               <div class="carousel-item">
                  <div class="carousel-caption cuplle">
                     <div class="container">
                        <div class="row">
                           <div class="col-md-8">
                              <div class="photog">
```

```
    <!-- end banner -->
    <!-- about -->
    <div class="about">
       <div class="container_width">
          <div class="row d_flex">
                <div class="col-md-7">
             <div class="titlepage text_align_left">
                <h2>About Lung Cancer </h2>
                <p>Lung cancer happens when cells in the lungs grow out of control, forming a mass called a tumor. It's usually caused by smoking, but sometimes it can happen to non-
                </p>
                <a class="read_more" href="about.html">About More</a>
             </div>
          </div>
          <div class="col-md-5">
             <div class="about_img text_align_center">
                <figure><img src="{{ url_for('static', filename= 'images/about.png')}}" alt="#"/></figure>
             </div>
          </div>

          </div>
       </div>
    </div>
    <!-- end about -->
   <!-- coronata -->
    <div class="coronata">
       <div class="container">
          <div class="row d_flex grid">
             <div class="col-md-7">
                <div class="coronata_img text_align_center">
                   <figure><img src="{{ url_for('static', filename= 'images/corona.png')}}" alt="#"/></figure>
                </div>
             </div>
             <div class="col-md-5 oder1">
                <div class="titlepage text_align_left">
                   <h2>Lung Cancer what it is?</h2>
                   <p>Lung cancer is a type of cancer that starts in the lungs. It usually begins in the cells that line the air passages. There are two main types: non-small cell lung
                   </p>
                   <a class="read_more" href="coronata.html">About More</a>
                </div>
             </div>
          </div>
       </div>
    </div>
    <!-- end coronata -->

      <!-- protect -->
    <div class="container">
      <div class="titlepage text_align_center ">
        <h2>Lung Cancer Prediction</h2>
      </div>
      <form id="prediction-form">
```

```
      <!-- protect -->
    <div class="container">
      <div class="titlepage text_align_center ">
        <h2>Lung Cancer Prediction</h2>
      </div>
      <form id="prediction-form">
          <label for="age">Age:</label>
          <input type="number" id="age" name="age" required><br><br>

          <label for="coughing">Coughing (1-2):</label>
          <input type="number" id="coughing" name="coughing" min="1" max="2" required><br><br>

          <label for="breath">Shortness of Breath (1-2):</label>
          <input type="number" id="breath" name="breath" min="1" max="2" required><br><br>

          <label for="chest_pain">Chest Pain (1-2):</label>
          <input type="number" id="chest_pain" name="chest_pain" min="1" max="2" required><br><br>

          <label for="wheezing">Wheezing (1-2):</label>
          <input type="number" id="wheezing" name="wheezing" min="1" max="2" required><br><br>

          <button type="submit">Predict</button>
      </form>

      <div id="result"></div>
          <p id="svm-prediction"></p>
          <p id="svm-probability"></p>
          <p id="xgb-prediction"></p>
          <p id="xgb-probability"></p>
          <p id="rf_prediction"></p>
          <p id="rf_probability"></p>
      </div>
    </div>
    <script src="{{ url_for('static', filename='js/script.js') }}"></script>
<div class="protect">
       <div class="container">
          <div class="row">
             <div class="col-md-12">
                <div class="titlepage text_align_center">
                   <h2>How to Protect Yourself</h2>
                   <p>To protect yourself from lung cancer, avoid smoking and exposure to secondhand smoke. Additionally, limit your exposure to environmental pollutants like radon gas
                   </p>
                </div>
             </div>
          </div>
       </div>
      <div class="protect_bg">
       <div class="container">
          <div class="row">
             <div class="col-md-12">
```

```
                <!--  Demos -->
                <div class="owl-carousel owl-theme">
                    <div class="item">
                        <div class="protect_box text_align_center">
                            <div class="desktop">
                                <i><img src="{{ url_for('static', filename= 'images/pro1.png')}}" alt="#"/></i>
                             <h3> Quit Smoking</h3>
                             <span> It  is one of the most important steps you can take to reduce your risk of lung cancer and improve your overall health. Seek support from healthcare profe
                            </div>
                        </div>
                    </div>
                    <div class="item">
                        <div class="protect_box text_align_center">
                        <div class="desktop">
                            <i><img src="{{ url_for('static', filename= 'images/pro2.png')}}" alt="#"/></i>
                         <h3> Do Yoga</h3>
                         <span> It can promote relaxation, reduce stress, and improve lung function through deep breathing exercises. Practicing yoga regularly may also help enhance over
                        </div>
                        </div>
                    </div>
                    <div class="item">
                        <div class="protect_box text_align_center">
                         <div class="desktop">
                            <i><img src="{{ url_for('static', filename= 'images/pro3.png')}}" alt="#"/></i>
                         <h3> Consult with Doctor</h3>
                         <span> It's crucial to consult with a doctor regularly, especially if you have a history of smoking or other risk factors for lung cancer. Your doctor can provid
                        </div>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </div>
      </div>
    </div>
    <!-- end protect -->
  <!-- cases -->
    <!-- end cases -->
      <!-- contributors -->
    <div class="contributors">
       <div class="container">
          <div class="row">
             <div class="col-md-12">
                <div class="titlepage text_align_center ">
                   <h2>Contributors</h2>
                </div>
             </div>
          </div>
          <div class="row d_flex">
             <div class=" col-md-4">
```

```
                   <h2>Contributors</h2>
                </div>
             </div>
          </div>
          <div class="row d_flex">
             <div class=" col-md-4">
                <div id="ho_efcet" class="reader text_align_center">
                   <i><img src="{{ url_for('static', filename= 'images/doctor1.png')}}" alt="#"/></i>
                   <h3>Harshkumar Soni</h3>
                </div>

             </div>
             <div class=" col-md-4">
                <div id="ho_efcet" class="reader text_align_center">
                   <i><img src="{{ url_for('static', filename= 'images/doctor2.png')}}" alt="#"/></i>
                   <h3>Tanvitha Doradla</h3>

                </div>
             </div>
             <div class=" col-md-4">
                <div id="ho_efcet" class="reader text_align_center">
                   <i><img src="{{ url_for('static', filename= 'images/doctor3.png')}}" alt="#"/></i>
                   <h3>Adewale Obalanlege</h3>
                </div>
             </div>
          </div>
       </div>
    </div>
    <!-- end cases -->
<!-- update -->
<div class="update">
   <div class="cevery_white">
      <div class="container">
         <div class="row">
            <div class="col-md-12">
               <div class="titlepage">
                  <h2>Get Every Update.... </h2>
               </div>
            </div>
         </div>
      </div>
   </div>
   <div class="cevery_bg">
      <div class="container">
         <div class="row">
            <div class="col-md-12">
               <form id="colof" class="form_subscri">
                  <input class="newsl" placeholder="Your Email" type="text" name="Email">
                  <button class="subsci_btn">Subscribe Now</button>
               </form>
            </div>
```

```
<footer>
   <div class="footer">
      <div class="container">
         <div class="row">
                  <div class="col-lg-2 col-md-6 col-sm-6">
                     <div class="hedingh3 text_align_left">
                        <h3>Resources</h3>
                        <ul class="menu_footer">
                           <li><a href="index.html">Home</a><li>
                           <li><a href="javascript:void(0)">What we do</a><li>
                           <li> <a href="javascript:void(0)">Media</a><li>
                           <li><a href="javascript:void(0)">Protection</a><li>
                           <li><a href="javascript:void(0)">Care</a><li>
                        </ul>


                     </div>
                  </div>
                  <div class="col-lg-3 col-md-6 col-sm-6">
                     <div class="hedingh3 text_align_left">
                        <h3>About</h3>
                        <p>We are final semester student of datascience. This is our capstone project.</p>
                     </div>
                  </div>


                  <div class="col-lg-3 col-md-6 col-sm-6">
                     <div class="hedingh3  text_align_left">
                        <h3>Contact  Us</h3>
                           <ul class="top_infomation">
                  <li><i class="fa fa-map-marker" aria-hidden="true"></i>
                     University of Texas at Arlington
                  </li>
                  <li><i class="fa fa-phone" aria-hidden="true"></i>
                     Call : +1(929)-620-6716
                  </li>
                  <li><i class="fa fa-envelope" aria-hidden="true"></i>
                     <a href="Javascript:void(0)">Email : harsh8100@gmail.com</a>
                  </li>
               </ul>

               </div>
            </div>
               <div class="col-lg-4 col-md-6 col-sm-6">
                     <div class="hedingh3 text_align_left">
                        <h3>countrys</h3>
                        <div class="map">
                          <img src="{{ url_for('static', filename= 'images/map.png')}}" alt="#"/>
                        </div>
                     </div>
                  </div>

         </div>
      </div>
```

- **CSS(style.css)**

```css
/*----------------------------------------------------------------------
File Name: style.css
----------------------------------------------------------------------*/


/*----------------------------------------------------------------------
import Fonts
----------------------------------------------------------------------*/

@import url('https://fonts.googleapis.com/css?family=Poppins:100,100i,200,200i,300,300i,400,400i,500,500i,600,600i,700,700i,800,800i,900,900i');
@import url('https://fonts.googleapis.com/css?family=Lato:400,700,700i,900&display=swap');
@import url('https://fonts.googleapis.com/css?family=Roboto:300,400,500,700,900&display=swap');

/*****---------------------------------------- 1)
 2) font-family: 'Poppins', sans-serif;
 ----------------------------------------*****/


/*----------------------------------------------------------------------
import Files
----------------------------------------------------------------------*/

@import url(normalize.css);
@import url(icomoon.css);
@import url(font-awesome.min.css);
@import url(owl.carousel.min.css);
@import url(nice-select.css);
@font-face {
    font-family: 'helveticaregular';
    src: url('../fonts/helvetica_400-webfont.woff2') format('woff2'), url('../fonts/helvetica_400-webfont.woff') format('woff');
    font-weight: normal;
    font-style: normal;
}


/*----------------------------------------------------------------------
basic
----------------------------------------------------------------------*/

* {
    box-sizing: border-box !important;
}

.container {
    max-width: 1170px;
}

html {
    scroll-behavior: smooth;
}

body {
    color: #666666;
    font-size: 14px;
    font-family: 'Poppins', sans-serif;
    line-height: 1.80857;
    font-weight: normal;
}
```

```
    transition: all .3s ease-in-out;
}

h1,
h2,
h3,
h4,
h5,
h6 {
    letter-spacing: 0;
    font-weight: normal;
    position: relative;
    padding: 0;
    font-weight: normal;
    line-height: normal;
    color: #111111;
    margin: 0
}

h1 {
    font-size: 24px;
}

h2 {
    font-size: 22px;
}

h3 {
    font-size: 18px;
}

h4 {
    font-size: 16px
}

h5 {
    font-size: 14px
}

h6 {
    font-size: 13px
}

*,
*::after,
*::before {
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}

h1 a,
h2 a,
h3 a,
h4 a,
h5 a,
h6 a {
    color: #212121;
    text-decoration: none!important;
```

```css
*,
*::after,
*::before {
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}

h1 a,
h2 a,
h3 a,
h4 a,
h5 a,
h6 a {
    color: #212121;
    text-decoration: none!important;
    opacity: 1
}

button:focus {
    outline: none;
}

ul,
li,
ol {
    margin: 0px;
    padding: 0px;
    list-style: none;
}

p {
    margin: 0px;
    font-weight: 500;
    font-size: 15px;
    line-height: 24px;
}

a {
    color: #222222;
    text-decoration: none;
    outline: none !important;
}

a,
.btn {
    text-decoration: none !important;
    outline: none !important;
    -webkit-transition: all .3s ease-in-out;
    -moz-transition: all .3s ease-in-out;
    -ms-transition: all .3s ease-in-out;
    -o-transition: all .3s ease-in-out;
    transition: all .3s ease-in-out;
}

img {
    max-width: 100%;
    height: auto;
```

```
    transition: all .3s ease-in-out;
}

img {
    max-width: 100%;
    height: auto;
}

 :focus {
    outline: 0;
}

.btn-custom {
    margin-top: 20px;
    background-color: transparent !important;
    border: 2px solid #ddd;
    padding: 12px 40px;
    font-size: 16px;
}

.lead {
    font-size: 18px;
    line-height: 30px;
    color: #767676;
    margin: 0;
    padding: 0;
}

.form-control:focus {
    border-color: #ffffff !important;
    box-shadow: 0 0 0 .2rem rgba(255, 255, 255, .25);
}

.navbar-form input {
    border: none !important;
}

.badge {
    font-weight: 500;
}

blockquote {
    margin: 20px 0 20px;
    padding: 30px;
}

button {
    border: 0;
    margin: 0;
    padding: 0;
    cursor: pointer;
}

.full {
    float: left;
    width: 100%;
}

.full {
    width: 100%;
```

```
/**-- heading section --**/

/*---------------------------- preloader area -----------------------------*/
.loader_bg {
    position: fixed;
    z-index: 9999999;
    background: #fff;
    width: 100%;
    height: 100%;
}

.loader {
    height: 100%;
    width: 100%;
    position: absolute;
    left: 0;
    top: 0;
    display: flex;
    justify-content: center;
    align-items: center;
}

.loader img {
    width: 280px;
}

.titlepage {
    text-align: center;
    padding-bottom: 60px;
}

.titlepage h2 {
   text-transform: uppercase;
    font-size: 45px;
    color: #353d47;
    line-height: 50px;
    font-weight: bold;
    padding: 0;
    display: inline-block;
}

.d_flex {
    display: flex;
    align-items: center;
    flex-wrap: wrap;
}

.read_more {
    font-size: 17px;
    background-color: #05254d;
    color: #fff;

    width: 100%;
    max-width: 275px;
    text-align: center;
```

```
    display: inline-block;
    transition: ease-in all 0.5s;
    font-weight: bold;
    height: 75px;
    text-transform: uppercase;
    line-height: 75px;
}

.read_more:hover {
    background: #ed1c24;
    color: #fff;
    transition: ease-in all 0.5s;
}

.text_align_left {
    text-align: left;
}

.text_align_right {
    text-align: right;
}

.text_align_center {
    text-align: center;
}


/*-- header --*/

.header-area  {
    padding: 30px 16px;
    color: #f1f1f1;
    width: 100%;
    z-index: 9999999;
    height: 87px;
    background:#ed242e;
}


.logo_sm {display: none;}



.logo_midle {color: #fef6fc; font-weight: bold; color: #fff !important;}
.logo_midle::before {
    position: absolute;
    content: "";
    bottom: -93px;
    width: 158px;
    height: 96px;
    background: url(../images/logo.png);
    z-index: 999;
    margin: 0 auto;
    left: 0;
    right: 26px;
}
.site-navbar {
    display: flex;
    justify-content: space-between;
    align-items: center;
```

```css
.logo_midle {color: #fef6fc; font-weight: bold; color: #fff !important;}
.logo_midle::before {
    position: absolute;
    content: "";
    bottom: -93px;
    width: 158px;
    height: 96px;
    background: url(../images/logo.png);
    z-index: 999;
    margin: 0 auto;
    left: 0;
    right: 26px;
}
.site-navbar {
    display: flex;
    justify-content: space-between;
    align-items: center;
}

.site-navbar ul {
    margin: 0;
    padding: 0;
    list-style: none;
    display: flex;
}

.site-navbar ul li a {
    color: #fef6fc;
    font-size: 17px;
    display: block;
    text-decoration: none;
    text-transform: uppercase;
}

.site-navbar ul li {
    padding-right: 70px;
}

.site-navbar ul li:last-child {
    padding-right: 0;
}

.site-navbar ul li a:hover {
    color: #00132b;
}
.site-navbar ul li a.active {
    color: #00132b;
}

/* navbar regular css end */


/* nav-toggler css start */

.nav-toggler {
    border: 3px solid #fff;
    padding: 5px;
    background-color: transparent;
    cursor: pointer;
```

```css
.nav-toggler {
    border: 3px solid #fff;
    padding: 5px;
    background-color: transparent;
    cursor: pointer;
    height: 39px;
    display: none;
    z-index: 99999;
}

.nav-toggler span,
.nav-toggler span:before,
.nav-toggler span:after {
    width: 28px;
    height: 3px;
    background-color: #fff;
    display: block;
    transition: .3s;
}

.nav-toggler span:before {
    content: '';
    transform: translateY(-9px);
}

.nav-toggler span:after {
    content: '';
    transform: translateY(6px);
}

.nav-toggler.toggler-open span {
    background-color: transparent;
}

.nav-toggler.toggler-open span:before {
    transform: translateY(0px) rotate(45deg);
}

.nav-toggler.toggler-open span:after {
    transform: translateY(-3px) rotate(-45deg);
}


/* nav-toggler css start */


/* intro-area css start */

.intro-area {
    height: calc(100vh - 61px);
    display: flex;
    align-items: center;
    text-align: center;
    color: #fff;
}
```

```css
/* intro-area css end */


.left {

    position: absolute;
    background: #00132b;
    width: 80px;
    left: 0;
    height: 87px;
    top: 0;
    text-align: center;
    line-height: 87px;

}

.left  a i {
    color: #fff;

}
.right {
    position: absolute;
    background: #00132b;
    width: 80px;
    right: 0;
    height: 87px;
    top: 0;
    text-align: center;
    line-height: 87px;

}
.right  a i {
    color: #fff;

}
/** end header **/



/** end header **/


/** banner section **/

.full_bg {
     background: url(../images/banner.png);
     background-size: cover;
     background-position: center;
     background-repeat: no-repeat;
     height: 100vh;
    padding-bottom: 150px;
     position: relative;
}

.full_bg .carousel-indicators {
    bottom: -43px;
```

```css
.full_bg .carousel-indicators li.active {
    display: inline;
}

.full_bg .carousel-indicators li {
    text-indent: unset;
    opacity: 1;
    color: #fff;
    font-size: 18px;
    text-align: center;
    line-height: 59px;
    margin-right: 0px;
    background: transparent;
    border: none;
    text-indent: unset;
    display: none;
}



.slider_main {
    padding-top: 11%;
}

.cuplle {
    position: inherit;
    bottom: 0;
    padding: 0;
    z-index: 9999;
    left: 0;
    right: 0;
}

.photog {
    z-index: 9999;
    position: relative;
    text-align: left;
}

.photog .read_more:focus {
    box-shadow: none;
}

.photog h1 {
    padding: 0;
    font-size: 89px;
    line-height: 90px;
    color: #fff;
    font-weight: bold;
    text-align: left;
    padding-bottom: 70px;
}


.photog .read_more {
   margin-right: 15px;
   font-weight: normal;
   background:#fff;
   color: #000;
   height: 50px;
```

```
.photog .read_more:hover {background: #ed1c24; color: #fff;}
.slider_main .carousel-item {
     height: 100%;
     width: 100%;
}

#banner1 .carousel-control-prev,
#banner1 .carousel-control-next {
     width:80px;
     height: 80px;
     background-color: #ed1c24;
     color: #fff;
     font-size: 35px;
     opacity: 1;
     top: 50%;
     border-radius: 50%;

}

#banner1 .carousel-control-prev {

      left: 0%;
}

#banner1 .carousel-control-next {

     right: 0%;
}

#banner1 .carousel-control-next:focus,
#banner1 .carousel-control-next:hover,
#banner1 .carousel-control-prev:focus,
#banner1 .carousel-control-prev:hover {
     background-color: #fff;
     color: #ed1c24;
}


/** end banner section **/


/** about section **/

.about {
    background: #fff;
    padding: 80px 50px;
    position: relative;
    margin: 0 70px;
    box-shadow: 3px 0 62px rgba(13, 3, 3, 0.18);
    border-radius: 60px;
    margin-top: -130px;
}

.container_width {max-width: 1380px;
 padding: 0 15px;
```

```
    margin-top: -130px;
}

.container_width {max-width: 1380px;
 padding: 0 15px;
 display: block;
 margin: 0 auto;
 clear: both;


}

.about .titlepage {
    padding-bottom: 0;
}
.about .titlepage h2 {
    position: relative;
    padding-bottom: 18px;
}

.about .titlepage h2::before {
    position: absolute;
    content: "";
    bottom: 0px;
    background: #ed1c24;
    width: 93px;
    height: 11px;
    left: 0;
    border-radius: 20px;
}
.about .titlepage p {
    font-weight: normal;
    font-size: 17px;
    line-height: 30px;
    color: #111111;
    padding: 35px 0px 50px 0px;
}


.about_img figure {
    margin: 0;
}

/** end about section **/


/** coronata section **/

.coronata {
    background: #fff;
   padding-top: 100px;

}

.coronata .titlepage {
    padding-bottom: 0;
}

.coronata .titlepage p {
```

```css
/** protect **/

.protect {
    padding: 0 30px;
     padding-top: 80px;
    padding-bottom: 25px;
}


.protect_bg  {
       background:url(../images/project.png);
     background-repeat: no-repeat;
     background-position: center;
     background-size: 100% 100%;
       padding-top: 80px;
      padding-bottom: 48px;
}

.protect .titlepage p {
     padding-top: 5px;

}
.desktop {
    padding: 45px 20px 30px 20px;
    background: #fff;
    border-radius: 20px;
}


.protect_box h3 {
     color: #0d0e0e;
     font-size: 17px;
     font-weight: bold;
     padding-top: 25px;
     padding-bottom: 5px;

}

.protect_box span {
     color: #081419;
     font-weight:normal;
}
.protect .read_more {
    margin: 0 auto;
    display: block;
    max-width: 174px;
    height: 51px;
    line-height: 51px;
    font-weight: bold;
    font-size: 15px !important;
    margin-top: 40px;

}
.owl-carousel .owl-item img {
    width: inherit !important;
    text-align: center;
    margin: 0 auto;
    height: 90px;
}
```

```
}
.owl-carousel .owl-item img {
    width: inherit !important;
    text-align: center;
    margin: 0 auto;
    height: 90px;
}

.owl-carousel .owl-nav.disabled {
    display: inherit !important;
    padding-top: 80px;
}


.owl-carousel .owl-nav button.owl-next,
.owl-carousel .owl-nav button.owl-prev {
    width: 90px;
    height: 90px;
    background-color: #ed1c24 !important;
    color: #fff !important;
    font-size: 55px !important;
    line-height: 30px !important;
    opacity: 1;
    border-radius: 50px;
    transition: ease-in all 0.5s;
    box-shadow: 0 0 80px rgba(13, 3, 3, 0.11);
}
.owl-carousel .owl-nav button.owl-next:hover,
.owl-carousel .owl-nav button.owl-prev:hover {
    background-color: #fff !important;
    transition: ease-in all 0.5s;
    color: #000 !important;
}
.owl-next {
    position: absolute;
    right: 42%;
}
.owl-prev {
    position: absolute;
    left: 42%;
}


/** end protect **/



/** cases section **/

.cases {
    background: #fff;
    padding: 0 30px 0 30px;
   padding-top: 70px;
```

```css
/** cases section **/

.cases {
    background: #fff;
    padding: 0 30px 0 30px;
   padding-top: 70px;


}
.cases .titlepage {padding-bottom: 30px;}
.cases .titlepage p {padding-top: 5px;}
.latest {
    margin-top: 30px;
    background: #fff;
    padding: 25px 25px 0px 25px;
    box-shadow: 3px 0 62px rgba(13, 3, 3, 0.20);
    margin-bottom: 70px;
}


.latest figure {margin: 0;}

.latest .read_more {
    margin: 0 auto;
    display: block;
    position: absolute;
    z-index: 99999;
    max-width: 174px;
    height: 51px;
    line-height: 51px;
    font-weight: normal;
    top: 22%;
    right: 0;
    left: 0;
    bottom: 0;
}

.nostrud h3 {
    color: #151515;
    font-size: 30px;
    line-height: 30px;
    font-weight: bold;
    padding-top: 52px;
}

.nostrud p {
    padding: 5px 0 60px 0;
    color: #0f100d;
    line-height: 28px;
    font-weight: normal;
}

.nostrud {position:relative;}
.nostrud::before {
    position: absolute;
    content: "";
    left: 0;
    right: 0;
    background: url(../images/corona_icon.png);
```

```css
.reader i {
    margin: 0;
}

.reader i img {
   border-radius: 50%;
      box-shadow: 3px 0 62px rgba(13, 3, 3, 0.18);  margin-top: -50px;
}



.reader h3 {
    color: #151515;
    font-size: 20px;
    line-height: 20px;
    font-weight: bold;
    padding-top: 42px;
    text-transform: uppercase;
}

.reader p {
    padding: 20px 0 40px 0;
    color: #0f100d;
    line-height: 28px;
    font-weight: normal;
}

#ho_efcet:hover .reader {background: #fafaf9;  transition: ease-in all 0.5s;}




.doctors::before {
    position: absolute;
    content: "";
    right: 4%;
    background: url(../images/case.png);
    top: 34%;
    width: 120px;
    height: 121px;
}

/** end doctors section **/


/** contact **/

.contact {
    background: #fff;
    padding-top: 50px;
}



.contact .titlepage {padding-bottom: 0;}
.main_form {padding-top:45px;}
.main_form .contactus {
     padding: 0px 20px;
     margin-bottom: 25px;
```

```css
ul.top_infomation li {
    display: block;
    font-size: 17px;
    color: #fff;
}

ul.top_infomation li:last-child {
    padding-right: 0;
}

ul.top_infomation li i {
    margin-right: 5px;
    color: #fff;
    text-align: center;
    transition: ease-in all 0.5s;
}

ul.top_infomation li a {
    color: #fff;
}

ul.top_infomation li a:hover {
    color: #e92f0b;
}

ul.top_infomation li i:hover {
    background: #eb4746;
    color: #fff;
    border-radius: 25px;
    cursor: pointer;
    transition: ease-in all 0.5s;
}



.inner_page .full_bg {
     height: 106px;
     padding-bottom: 0;

}
.inner_page .about {
    margin-top: 145px;
    margin-bottom: 80px;
}

.inner_page .doctors { padding-top: 140px;  padding-bottom: 50px;}
.inner_page .coronata {
    padding-top: 130px;
    padding-bottom: 80px;
}
.inner_page .cases {padding-top: 140px;
padding-bottom: 40px;}
.inner_page .contact {
    padding-top: 145px;
    padding-bottom: 80px;
}
```

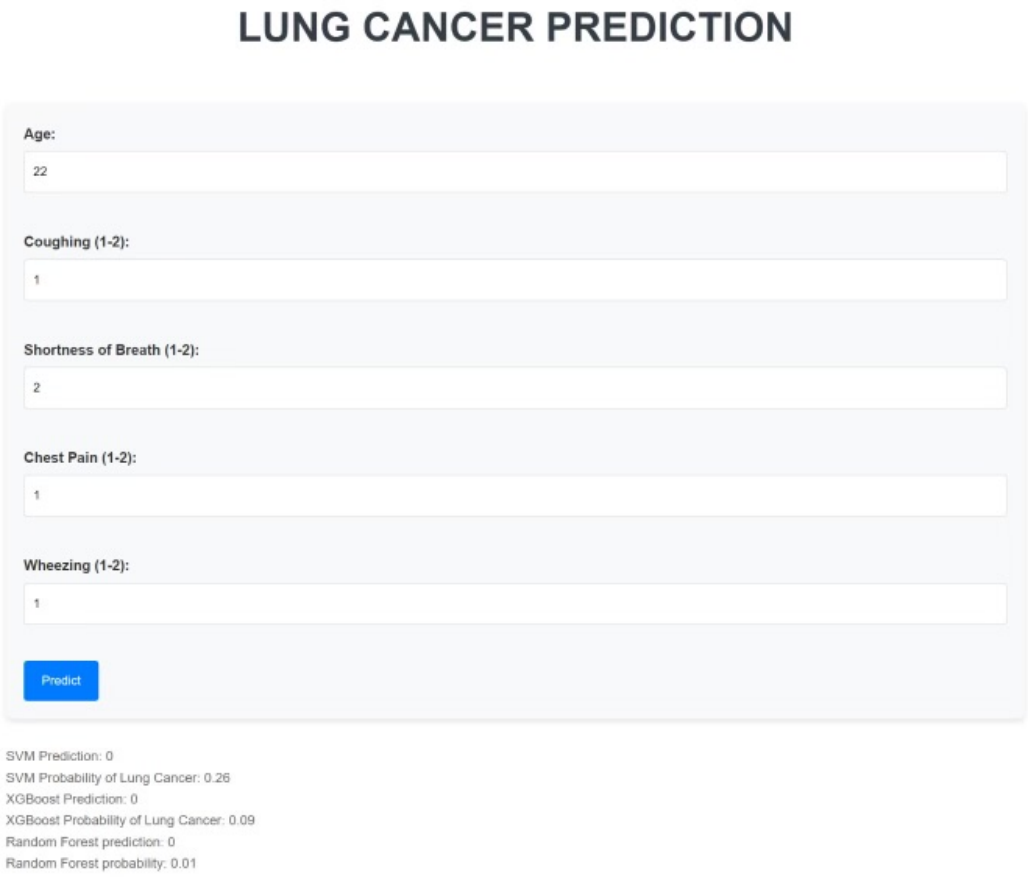
- **Javascript**

```
document.getElementById('prediction-form').addEventListener('submit', function(e) {
    e.preventDefault();

    const formData = new FormData(this);
    const age = formData.get('age');
    const coughing = formData.get('coughing');
    const breath = formData.get('breath');
    const chest_pain = formData.get('chest_pain');
    const wheezing = formData.get('wheezing');

    fetch('/predict', {
        method: 'POST',
        body: JSON.stringify({ age, coughing, breath, chest_pain, wheezing }),
        headers: {
            'Content-Type': 'application/json'
        }
    })
    .then(response => response.json())
    .then(data => {
        document.getElementById('svm-prediction').innerText = 'SVM Prediction: ' + data.svm_prediction;
        document.getElementById('xgb-prediction').innerText = 'XGBoost Prediction: ' + data.xgb_prediction;
        document.getElementById('svm-probability').innerText = 'SVM Probability of Lung Cancer: ' + data.svm_probability;
        document.getElementById('xgb-probability').innerText = 'XGBoost Probability of Lung Cancer: ' + data.xgb_probability;
        document.getElementById('rf_prediction').innerText = 'Random Forest prediction: ' + data.rf_prediction;
        document.getElementById('rf_probability').innerText = 'Random Forest probability: ' + data.rf_probability;
    })
    .catch(error => console.error('Error:', error));
});
```

- Flask File (app.py)

```python
from flask import Flask, render_template, request, jsonify
from joblib import load
from math import exp

app = Flask(__name__)

# Load models
svm_model = load('svm_model.pkl')
xgb_model = load('xgb_model.pkl')
rf_model = load('best_random_forest_model.pkl')

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

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json()
    age = float(data['age'])
    coughing = float(data['coughing'])
    breath = float(data['breath'])
    chest_pain = float(data['chest_pain'])
    wheezing = float(data['wheezing'])

    # Predictions
    svm_prediction = svm_model.predict([[age, coughing, breath, chest_pain, wheezing]])[0]
    xgb_prediction = xgb_model.predict([[age, coughing, breath, chest_pain, wheezing]])[0]
    rf_prediction = rf_model.predict([[age, coughing, breath, chest_pain, wheezing]])[0]

    # Probability calculations for SVM
    svm_decision = svm_model.decision_function([[age, coughing, breath, chest_pain, wheezing]])[0]
    probability_of_lung_cancer_svm = 1 / (1 + exp(-svm_decision))

    # Probability calculations for XGBoost
    xgb_probabilities = xgb_model.predict_proba([[age, coughing, breath, chest_pain, wheezing]])[0]
    probability_of_lung_cancer_xgb = xgb_probabilities[1]

    # Probability calculations for Random Forest
    rf_probabilities = rf_model.predict_proba([[age, coughing, breath, chest_pain, wheezing]])[0]
    probability_of_lung_cancer_rf = rf_probabilities[1]

    result = {
        'svm_prediction': int(svm_prediction),
        'xgb_prediction': int(xgb_prediction),
        'rf_prediction': int(rf_prediction),
        'svm_probability': round(float(probability_of_lung_cancer_svm), 2),
        'xgb_probability': round(float(probability_of_lung_cancer_xgb), 2),
        'rf_probability': round(float(probability_of_lung_cancer_rf), 2)
    }

    return jsonify(result)

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

Website Results:



LUNG CANCER PREDICTION

Age:

68

Coughing (1-2):

1

Shortness of Breath (1-2):

2

Chest Pain (1-2):

1

Wheezing (1-2):

2

Predict

SVM Prediction: 1
SVM Probability of Lung Cancer: 0.54
XGBoost Prediction: 1
XGBoost Probability of Lung Cancer: 0.85
Random Forest prediction: 1
Random Forest probability: 0.72