UNIVERSITY OF CANBERRA
AUSTRALIA'S CAPITAL UNIVERSITY

Faculty of Science and Technology

Assignment Coversheet

Student ID number & Student Name	U3241507 Myeisha Jing Lin Foo
Unit name	Software Technology
Unit number	ST1(4483)
Unit Tutor	Pranav Gupta
Assignment name	ST1 Capstone Project – Semester 2 2023
Due date	29/10/2023
Date submitted	29/10/2023

You must keep a photocopy or electronic copy of your assignment.

Student declaration

I certify that the attached assignment is my own work. Material drawn from other sources has been appropriately and fully acknowledged as to author/creator, source and other bibliographic details.

Signature of student: Myeisha Foo **Date: 29/10/2023**

Table of Contents

Introduction

This report provides a comprehensive overview of the Python Capstone Project conducted for the ST1 unit, focusing on the specific dataset at hand. The chosen dataset pertains to cancer prediction, an immensely valuable resource that empowers individuals to assess their risk of developing cancer. Such predictive systems not only offer cost-effective insights but also aid individuals in making informed decisions based on their unique cancer risk status.

The dataset has been sourced from the online lung cancer prediction system, which aggregates pertinent information to facilitate accurate risk assessment. It comprises a total of 16 attributes and encompasses 284 instances, offering a substantial volume of data for analysis and prediction. The attributes within the dataset encompass critical factors such as gender, age, smoking history, and various health indicators, which collectively contribute to the assessment of lung cancer risk.

This initiative is of paramount importance considering the prevalence of lung cancer and its significant impact on public health. The dataset's attributes include gender, age, smoking history, presence of specific symptoms, and the ultimate outcome of whether the individual has been diagnosed with lung cancer (YES/NO).

For comprehensive details and access to the dataset, please refer to the Kaggle repository [1] (https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer/data).

The report delves into the methodology employed in the development of a prototype software platform, guided by a data-driven scientific approach. It encompasses essential phases such as exploratory data analysis, predictive analytics, and the implementation of software tools. These tools have been crafted as part of the capstone project and are designed to facilitate cancer risk assessment. The software is versatile, available both as a desktop Tkinter application and an online web-based Flask/Streamlit application, providing accessibility to a broad user base.

The initiative to develop this software tool is rooted in the need for more accessible and accurate cancer prediction systems. Currently, invasive diagnostic methods pose significant challenges, both in terms of invasiveness and cost. The alternative non-invasive diagnostics methods have shown limitations in terms of diagnostic accuracy. The goal of this project is to harness the potential of data-driven analysis and predictive modeling to enhance the diagnostic power of these non-invasive tests. Ultimately, the aim is to provide a reliable and accessible alternative to the current gold standard, significantly impacting the field of cancer diagnosis.

The subsequent sections of this report will provide a detailed account of the methodology employed in the project, showcasing the innovative tools and approaches used to advance cancer risk prediction.

Methodology

1. Design and Development

The project encompasses both Exploratory Data Analysis (EDA) and Predictive Data Analysis (PDA) to identify the most appropriate artificial intelligence learning model. These analytical processes are vital for recognizing patterns, associations, and making predictions, all of which are essential for effectively addressing real-world challenges.

2. Implementation

After determining the ideal AI model through PDA, it is incorporated into the Streamlit Application as the core component for creating the user interface. Integrating the AI model into Streamlit allows users to interact with the solution effortlessly.

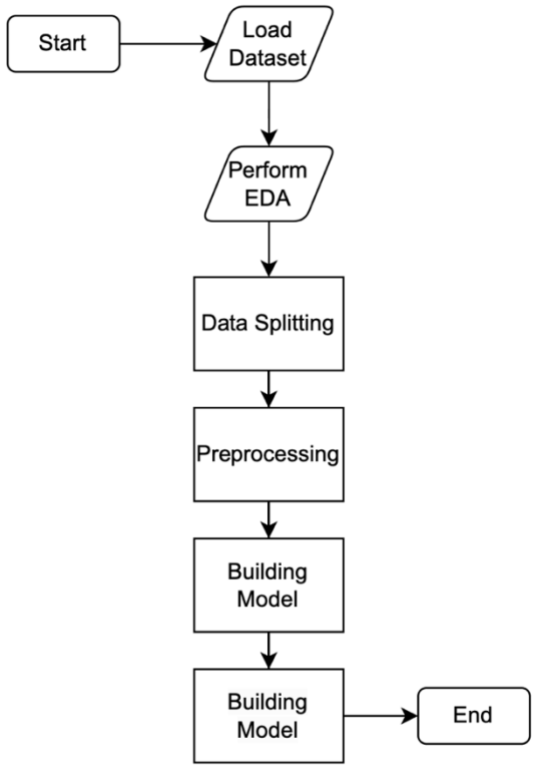
3. Deployment

Upon successful implementation, the tool is deployed as a web application using Streamlit.

Design and development

Algorithm for dataset:

Load Dataset
Perform EDA
Data Splitting
Preprocessing
Build Model
Performance Evaluation



Dataset Description

This project utilizes a publicly available dataset sourced from the UCI Machine Learning Repository and Kaggle [1]. The dataset in question is centered around the prediction of lung cancer risk, aiding individuals in assessing their susceptibility to lung cancer and making informed decisions based on their risk status. The dataset encompasses 284 instances, each characterized by 16 attributes, serving as critical indicators for cancer prediction.

Exploratory Data Analysis

The exploratory data analysis (EDA) process was conducted in the Google Colab environment due to its efficiency and simplicity, which allows it to be executed directly from a web browser. Python was chosen as the programming language for developing the EDA code.

Questions to answer when analysing the dataset:

1. What is the distribution of lung cancer cases in the dataset?
2. How does age related to the likelihood of lung cancer?
3. Are there any significant correlations between smoking and other attributes with the presence of lung cancer?
4. What are the most common symptoms and conditions associated with lung cancer?
5. Does Gender have correlation to lung cancer

The initial step in the EDA process involves gaining a fundamental understanding of the data's characteristics and attributes:

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list all files under the input directory

import os
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))

import matplotlib.pyplot as plt
import seaborn as sns
import missingno as msno
import plotly.graph_objects as go
import plotly.express as px
%matplotlib inline
import warnings
warnings.filterwarnings('ignore')
```

Findings : where various libraries are imported

```
df = pd.read_csv("/content/drive/MyDrive/CapstoneProject/surveylungcancer.csv")
```

Findings : To read the CSV (Comma-Separated Values) file named "surveylungcancer.csv" from a specific file path

```
# Display first five data
df.head().T
```

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

Finding : To display the first five rows of the DataFrame df and transpose the result for a more convenient visualization of the data.

```
# Display the last 5 data
df.tail().T
```

	304	305	306	307	308
GENDER	F	M	M	M	M
AGE	56	70	58	67	62
SMOKING	1	2	2	2	1
YELLOW_FINGERS	1	1	1	1	1
ANXIETY	1	1	1	2	1
PEER_PRESSURE	2	1	1	1	2
CHRONIC DISEASE	2	1	1	1	1
FATIGUE	2	2	1	2	2
ALLERGY	1	2	2	2	2
WHEEZING	1	2	2	1	2
ALCOHOL CONSUMING	2	2	2	2	2
COUGHING	2	2	2	2	1
SHORTNESS OF BREATH	2	2	1	2	1
SWALLOWING DIFFICULTY	2	1	1	1	2
CHEST PAIN	1	2	2	2	1
LUNG_CANCER	YES	YES	YES	YES	YES

Finding : To display the last five rows of the DataFrame df and transpose the result for a more convenient visualization of the data.

```
[ ]  # Display the shape of the DataFrame to show the number of rows and columns
     df.shape
```

```
     (309, 16)
```

Findings : To determine and display the dimensions of the DataFrame, showing the number of rows and columns it contains. This provides an overview of the dataset's size.

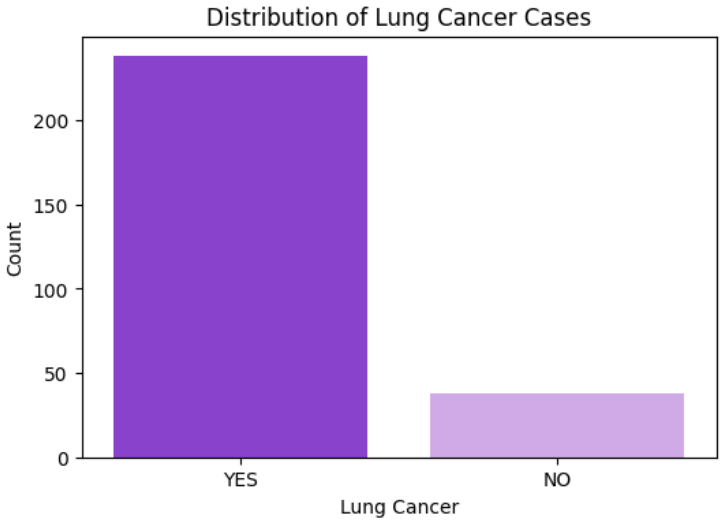
```
[ ]  # Show all the headers
     df.columns
```

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

Findings : To retrieve and display the column headers (or feature names) of the DataFrame df. This step helps to identify the variables or attributes in the dataset.

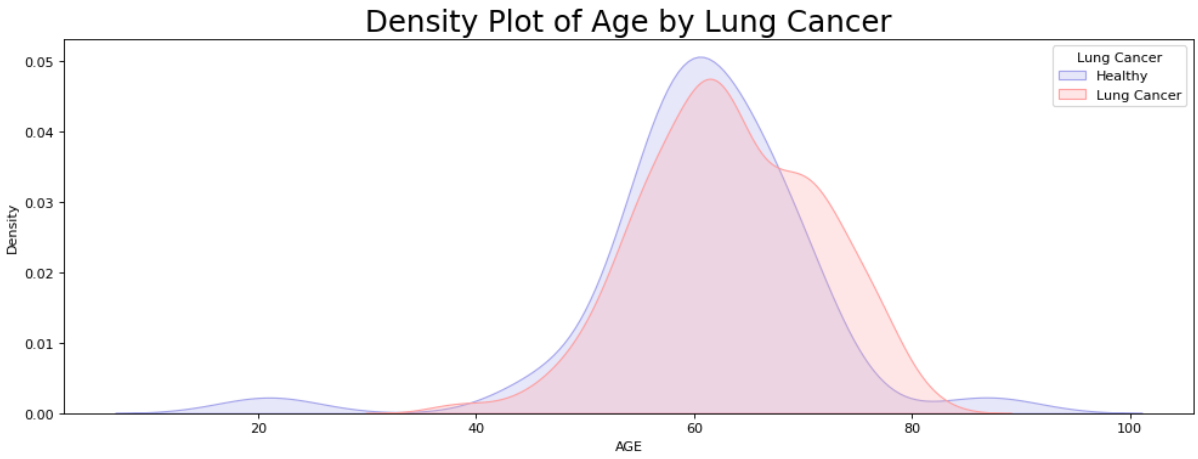
The subsequent stage involves addressing the following questions:

1. What is the distribution of lung cancer cases in the dataset?



Finding : In the dataset there are more people with lung cancer than without.

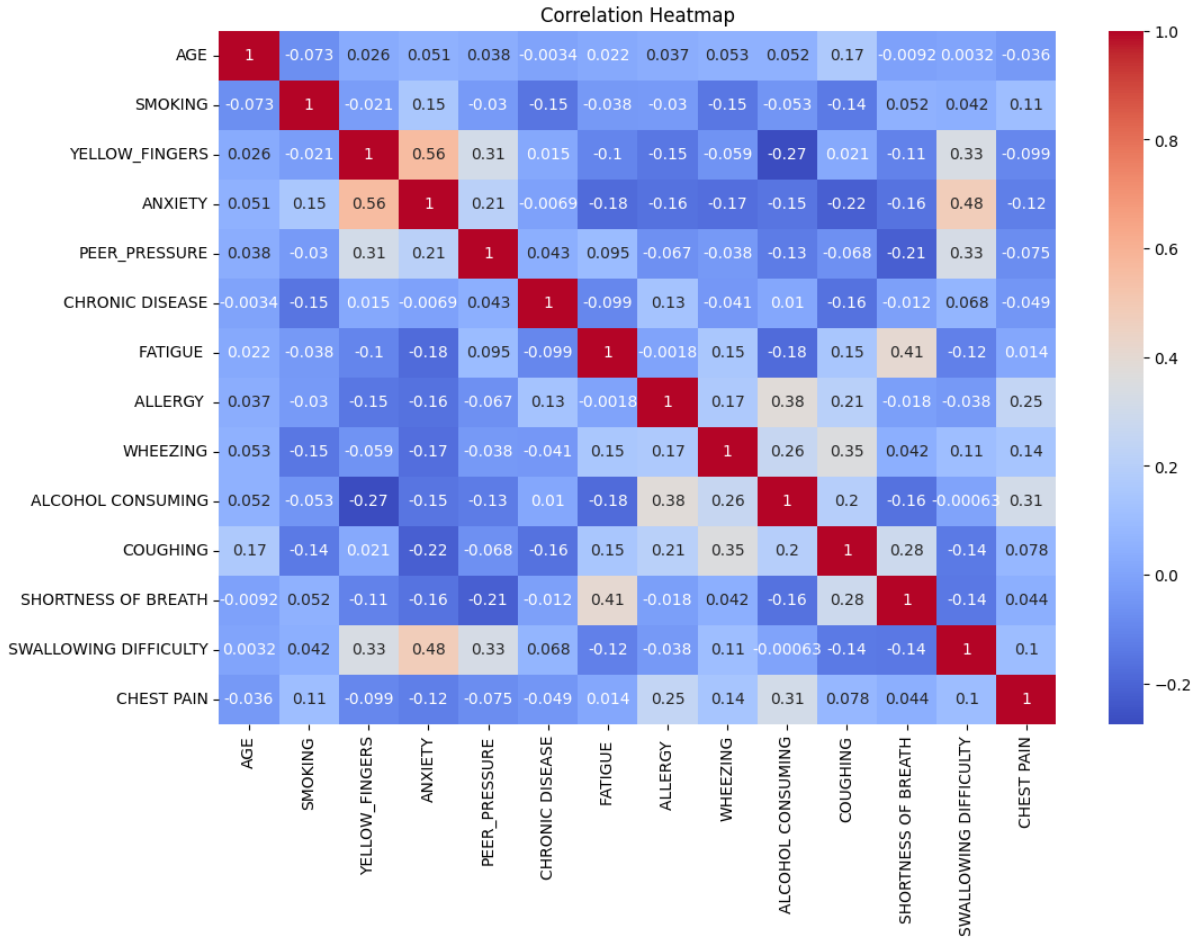
2. How does age relate to the likelihood of lung cancer?



Findings :

- People who are typically in the age 20 range tend to have healthy lungs.
- The age ranges between 30 to 90 it can vary whether you have a healthy or unhealthy lungs.

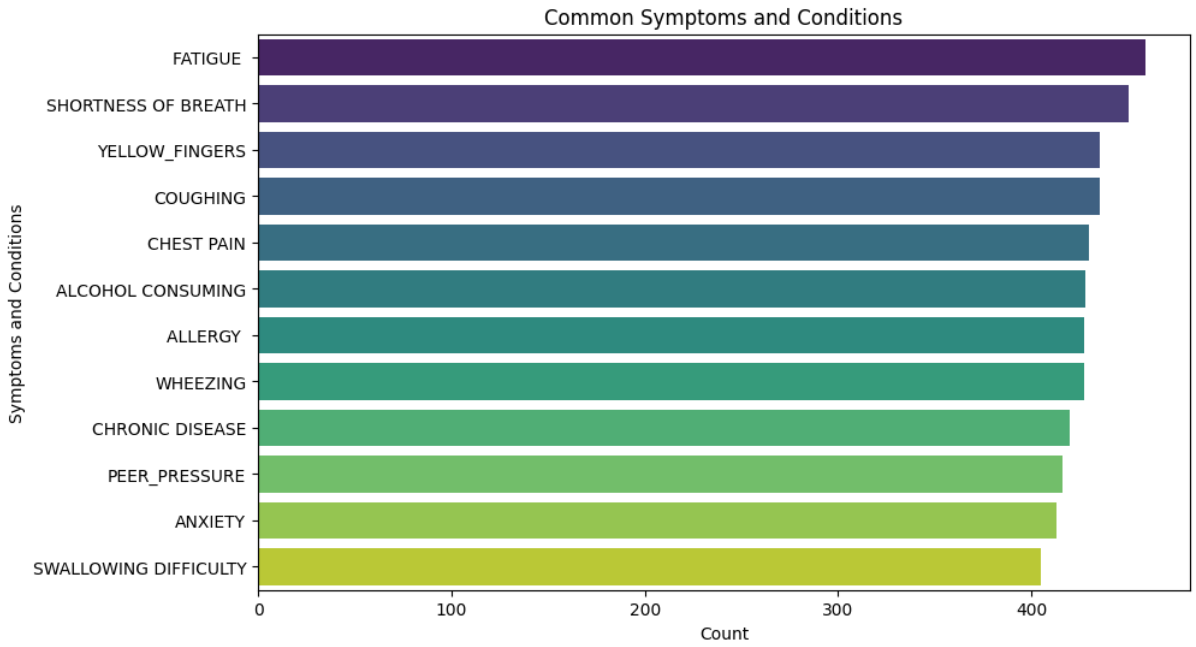
3. Are there any significant correlations between smoking and other attributes with the presence of lung cancer?



Findings :

- According to the heat correlation map majourity of the symptons and conditions have no correlation to one another.
- With the exception of Anxiety and Yellow fingers.

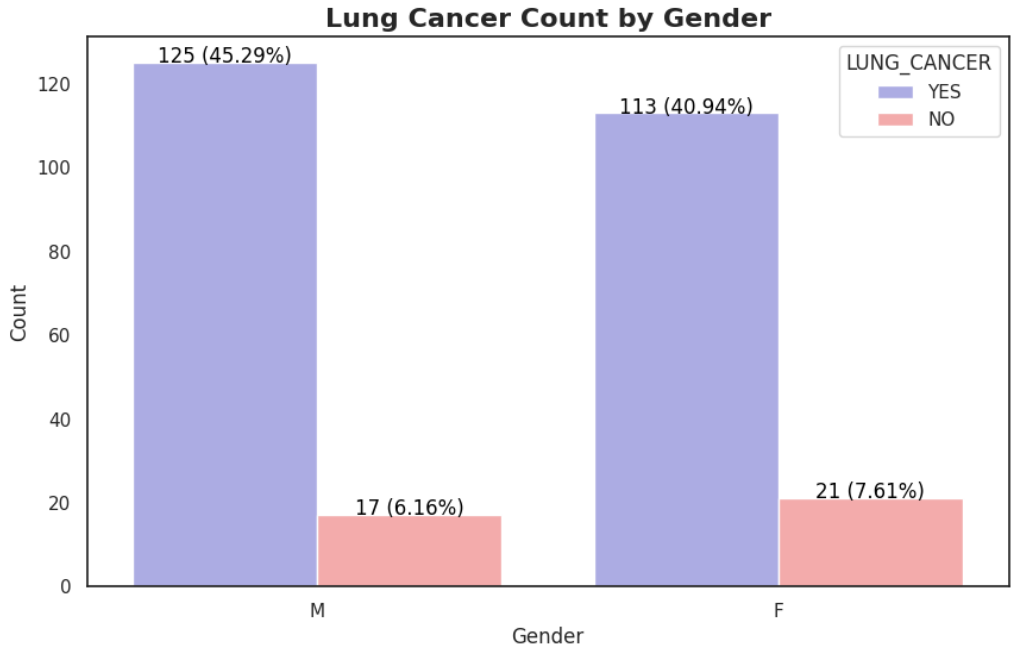
4. What are the most common symptoms and conditions associated with lung cancer?



Finding :

- The most common symptom and condition is Fatigue , Shortness Of Breath and Coughing.
- The least common symptom and condition is Peer Pressure , Anxiety and Swallowing Difficulty.

5. Does Gender have correlation to lung cancer



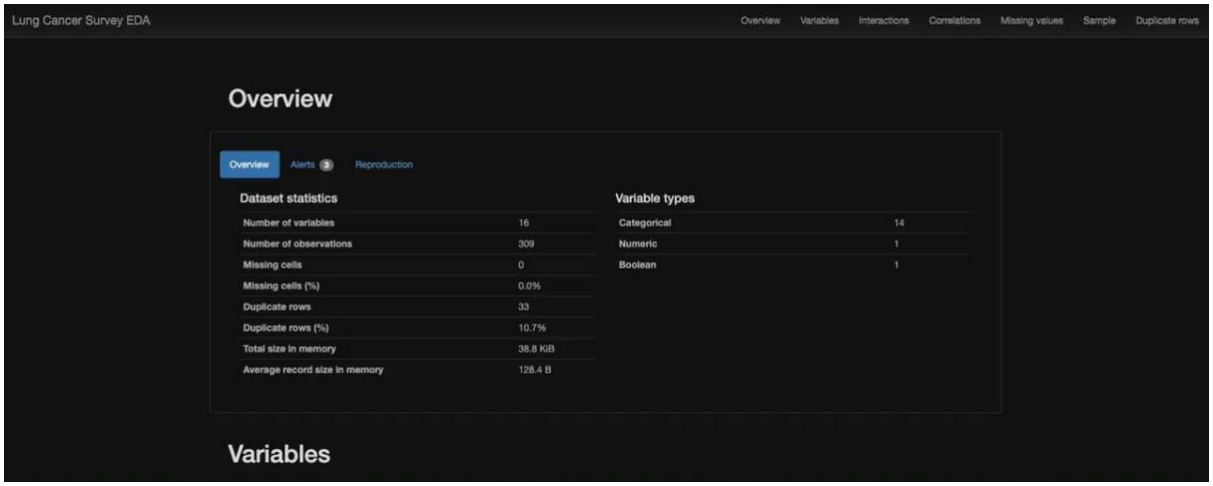
Findings :

- Gender does not have correlation to having or not having lung cancer
- It shows 46.93% of men have lung cancer and 40.45% of women have lung cancer, having a 6.48% difference.
- It shows 5.50% of men do not have lung cancer and 7.12% of women do not have lung cancer, having a 2.48% difference.

Pandas profiler report

```
# Installing Pandas Profiling module
!pip install https://github.com/pandas-profiling/pandas-profiling/archive/master.zip
```

```
#obtain full profiler report
#restart kernel
#re-run import libraries and data
import pandas as pd
import numpy as np
from pandas_profiling import ProfileReport
profile = ProfileReport(df,title="Lung Cancer Survey EDA", html={'style':{'full_width':True}})
profile.to_notebook_iframe()
```



Predictive Data Analytics Stage

Conducting PDA involves a series of essential steps, such as pre-processing, comparing classifiers to determine the most suitable machine learning classifier, and evaluating performance using various objective metrics like accuracy, classification report, confusion matrix, ROC-AUC curve, and prediction report, all of which are obtained through the utilization of Python's scikit-learn library.

Pre-processing:

- The dataset includes attributes with both continuous and categorical values.
- To handle these attributes, we perform attribute transformation, standardization, and normalization, utilizing tools like scikit-learn's OrdinalEncoder().

Normalisation:

- Remove the target variable from the data frame, apply normalization to it, and then rejoin the target variable with the data frame.

```python
# Import necessary libraries
from sklearn.exceptions import DataDimensionalityWarning  # Import a warning class
# Encode object columns to integers
from sklearn import preprocessing  # Import preprocessing module from scikit-learn
from sklearn.preprocessing import OrdinalEncoder  # Import OrdinalEncoder from preprocessing

# Loop through each column in the DataFrame
for col in df:
    # Check if the column's data type is 'object' (categorical)
    if df[col].dtype == 'object':
        # Use OrdinalEncoder to convert categorical values to integers
        df[col] = OrdinalEncoder().fit_transform(df[col].values.reshape(-1, 1))
```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY
0	1.0	69	1	2	2	1	1	2	1
1	1.0	74	2	1	1	1	2	2	2
2	0.0	59	1	1	1	2	1	2	1
3	1.0	63	2	2	2	1	1	1	1
4	0.0	63	1	2	1	1	1	1	1
...	...	...	...	...	...	...	...	...	...
304	0.0	56	1	1	1	2	2	2	1
305	1.0	70	2	1	1	1	1	2	2
306	1.0	58	2	1	1	1	1	1	2
307	1.0	67	2	1	2	1	1	2	2
308	1.0	62	1	1	1	2	1	2	2

```python
# Store the 'LUNG_CANCER' column in 'class_label' variable
class_label = df['LUNG_CANCER']

# Remove the 'LUNG_CANCER' column from the DataFrame
df = df.drop(['LUNG_CANCER'], axis=1)

# Normalize the DataFrame using min-max scaling
df = (df - df.min()) / (df.max() - df.min())

# Re-add the 'LUNG_CANCER' column to the DataFrame
df['LUNG_CANCER'] = class_label
```

```
# Create a label encoder
le = preprocessing.LabelEncoder()

# Encode the new variables
GENDER = le.fit_transform(list(df["GENDER"]))
AGE = le.fit_transform(list(df["AGE"]))
SMOKING = le.fit_transform(list(df["SMOKING"]))
YELLOW_FINGERS = le.fit_transform(list(df["YELLOW_FINGERS"]))
ANXIETY = le.fit_transform(list(df["ANXIETY"]))
PEER_PRESSURE = le.fit_transform(list(df["PEER_PRESSURE"]))
CHRONIC_DISEASE = le.fit_transform(list(df["CHRONIC DISEASE"]))
FATIGUE = le.fit_transform(list(df["FATIGUE "]))  # Note the space in column name
ALLERGY = le.fit_transform(list(df["ALLERGY "]))  # Note the space in column name
WHEEZING = le.fit_transform(list(df["WHEEZING"]))
ALCOHOL_CONSUMING = le.fit_transform(list(df["ALCOHOL CONSUMING"]))
COUGHING = le.fit_transform(list(df["COUGHING"]))
SHORTNESS_OF_BREATH = le.fit_transform(list(df["SHORTNESS OF BREATH"]))
SWALLOWING_DIFFICULTY = le.fit_transform(list(df["SWALLOWING DIFFICULTY"]))
CHEST_PAIN = le.fit_transform(list(df["CHEST PAIN"]))
LUNG_CANCER = le.fit_transform(list(df["LUNG_CANCER"]))  # Note the column name change
```

Model Preparation and Development

In order to prepare and develop this model, we start by partitioning the data frame into validation subsets. We achieve this by randomly selecting 80% of the data for training the AI model, while reserving the remaining 20% for testing purposes.

The development process involves creating a test set by excluding all rows from the data frame. We then create two sets: 'x,' comprising all attributes except the last column, and 'y,' representing the target class found in the last column.

```
# Create a list of tuples, where each tuple contains values from different variables
x = list(zip(GENDER, AGE, SMOKING, YELLOW_FINGERS, ANXIETY, PEER_PRESSURE, CHRONIC_DISEASE, FATIGUE, ALLERGY,
             WHEEZING, ALCOHOL_CONSUMING, COUGHING, SHORTNESS_OF_BREATH, SWALLOWING_DIFFICULTY, CHEST_PAIN, LUNG_CANCER))

# Create a list for the target variable
y = list(LUNG_CANCER)

# Define options and evaluation metric for model testing
num_folds = 5  # Number of cross-validation folds
seed = 7  # Random seed for reproducibility
scoring = 'accuracy'  # Metric to evaluate model performance (accuracy in this case)
```

```
# Model Test/Train
# Splitting what we are trying to predict into 4 different arrays -
# X train is a section of the x array(attributes) and similarly for Y(features)
# The test data will test the accuracy of the model created
import sklearn.model_selection
x_train, x_test, y_train, y_test = sklearn.model_selection.train_test_split(x, y, test_size = 0.20, random_state=seed)
# 0.2 means 80% training 20% testing
#splitting 20% of our data into test samples. If we train the model with higher data it already has seen that information and knows
```

```
# Size of train and test subsets after splitting
import numpy as np
np.shape(x_train), np.shape(x_test)
```

```
((247, 16), (62, 16))
```

```
# Predictive analytics model development by comparing different Scikit-learn classification algorithms
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.model_selection import KFold
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
from sklearn.metrics import accuracy_score
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import AdaBoostClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier

models = []
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
models.append(('GBM', GradientBoostingClassifier()))
models.append(('RF', RandomForestClassifier()))
# evaluate each model in turn
results = []
names = []
print("Performance on Training set")
for name, model in models:
  kfold = KFold(n_splits=num_folds,shuffle=True,random_state=seed)
  cv_results = cross_val_score(model, x_train, y_train, cv=kfold, scoring='accuracy')
  results.append(cv_results)
```

```
  names.append(name)
  msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
  msg += '\n'
  print(msg)
```

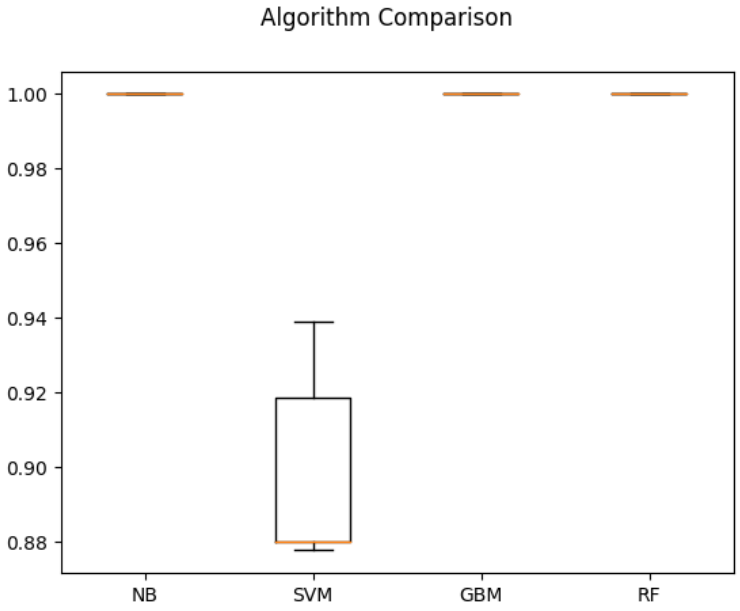
```
Performance on Training set
NB: 1.000000 (0.000000)

SVM: 0.898939 (0.025057)

GBM: 1.000000 (0.000000)

RF: 1.000000 (0.000000)
```

```
# Import the necessary library for data visualization
import matplotlib.pyplot as plt
# Create a new figure for the plot
fig = plt.figure()
# Set a title for the figure
fig.suptitle('Algorithm Comparison')
# Add a subplot (axes) to the figure
ax = fig.add_subplot(111)
# Generate a boxplot to compare algorithm performance
plt.boxplot(results)
# Set x-axis tick labels to the names of the classification algorithms
ax.set_xticklabels(names)
# Display the plot
plt.show()
```




```
# Import necessary libraries for model evaluation and prediction
from sklearn.tree import DecisionTreeClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score

# Extend the list of models with additional classifiers for evaluation
models.append(('DT', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC()))
models.append(('GBM', GradientBoostingClassifier()))
models.append(('RF', RandomForestClassifier()))

# Instantiate specific models
dt = DecisionTreeClassifier()
nb = GaussianNB()
gb = GradientBoostingClassifier()
rf = RandomForestClassifier()

# Select the best model for evaluation (in this case, 'rf' is used)
best_model = rf

# Train the selected best model on the training data
best_model.fit(x_train, y_train)

# Make predictions on the test dataset
y_pred = best_model.predict(x_test)

# Calculate and print the accuracy score of the best model on the test set
print("Best Model Accuracy Score on Test Set:", accuracy_score(y_test, y_pred))
```

```
Best Model Accuracy Score on Test Set: 1.0
```

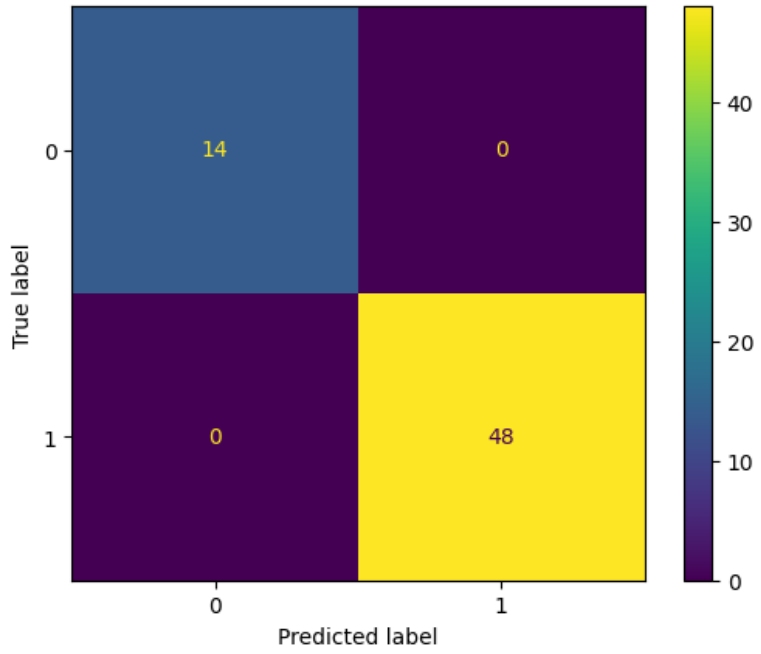
```python
# Model Performance Evaluation Metric 1 - Classification Report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        14
           1       1.00      1.00      1.00        48

    accuracy                           1.00        62
   macro avg       1.00      1.00      1.00        62
weighted avg       1.00      1.00      1.00        62
```

```python
# Model Performance Evaluation Metric 2
# Confusion matrix
from sklearn.metrics import confusion_matrix, ConfusionMatrixDisplay
cm = confusion_matrix(y_test, y_pred)
disp = ConfusionMatrixDisplay(confusion_matrix=cm)
disp.plot()
plt.show()
```




```python
#Model Evaluation Metric 3 - prediction report
for x in range(len(y_pred)):
  print("Predicted: ", y_pred[x], "Actual: ", y_test[x], "Data: ", x_test[x],)
```

```
Predicted:  0 Actual:  0 Data:  (0, 13, 0, 0, 0, 0, 1, 0, 0, 1, 0, 0, 1, 1, 0, 0)
Predicted:  0 Actual:  0 Data:  (0, 14, 0, 1, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0)
Predicted:  0 Actual:  0 Data:  (0, 38, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0)
Predicted:  1 Actual:  1 Data:  (1, 27, 0, 1, 0, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1)
Predicted:  0 Actual:  0 Data:  (1, 26, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0)
Predicted:  1 Actual:  1 Data:  (0, 27, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 1, 1)
Predicted:  0 Actual:  0 Data:  (0, 14, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0)
Predicted:  1 Actual:  1 Data:  (1, 13, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1)
Predicted:  1 Actual:  1 Data:  (1, 19, 1, 0, 1, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1)
Predicted:  1 Actual:  1 Data:  (1, 8, 1, 0, 0, 0, 0, 1, 0, 1, 1, 1, 1, 0, 1, 1)
Predicted:  1 Actual:  1 Data:  (1, 25, 1, 1, 1, 1, 0, 0, 1, 0, 1, 0, 0, 1, 1, 1)
Predicted:  1 Actual:  1 Data:  (1, 15, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 1, 1)
Predicted:  1 Actual:  1 Data:  (0, 18, 1, 1, 1, 1, 0, 1, 0, 0, 0, 1, 1, 1, 1, 1)
Predicted:  1 Actual:  1 Data:  (0, 28, 0, 1, 0, 0, 1, 1, 1, 1, 0, 1, 1, 0, 0, 1)
Predicted:  1 Actual:  1 Data:  (1, 18, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 1, 1)
Predicted:  1 Actual:  1 Data:  (0, 19, 1, 1, 1, 0, 1, 0, 1, 0, 1, 0, 0, 0, 0, 1)
Predicted:  1 Actual:  1 Data:  (1, 19, 1, 1, 1, 0, 1, 1, 1, 1, 0, 0, 1, 0, 0, 1)
```

Implementation and Deployment

Once the most effective AI model for predicting the risk of a heart attack is selected, the program is further advanced and eventually launched on the web via Streamlit.

The initial step involves implementing the program on the web by importing the model and presenting it through Streamlit.
Subsequently, the program is deployed on the web using Streamlit, as demonstrated here:

https://drive.google.com/drive/folders/1q3bJCPs7zwjkg7hr75Mm2pBm0tWFy2Ep?usp=sharing

Conclusions

This project is a notable milestone in predictive healthcare, with a focus on lung cancer. Using data-driven predictive analytics, it explores the complex dynamics of lung cancer and its risk factors. The developed model showcases AI's potential in accurately predicting lung cancer risk, benefiting healthcare professionals and researchers. It exemplifies how data-driven technologies are reshaping healthcare, offering a promising path to a healthier society.

References

1. MySarahMadBhat. (Year of dataset publication). Lung Cancer Prediction Dataset. Kaggle. Available at: https://www.kaggle.com/datasets/mysarahmadbhat/lung-cancer/data [Accessed 2 Oct.2023].

Journal

Week 10:

I came across a different dataset that I'd like to utilize for my capstone project, so I requested a change, and it was granted. Additionally, I have formulated five questions that I aim to address during my Exploratory Data Analysis.

Week 11:

I have finished conducting my Exploratory Data Analysis (EDA), Predictive Data Analysis (PDA), and model development. I've determined that RandomForestClassifier is the most suitable AI learning model for my predictions and intend to use it for my project implementation and deployment.

Week 12:

My project implementation and deployment using Streamlit are now complete, and I have prepared my PowerPoint presentation slides, which I delivered. I have also successfully completed my deployment on Streamlit.

Week 13:

I have successfully created a Git Repository containing all my project work.