NSBM Green University

Faculty of Computing

BSc (Hons) of Data Science

DS201.3 Introduction to Data Science

Module Lecturer: Ms. Kavishka Rajapaksha

Final Report of Group E

Data Science Approach for Early Detection of Lung Cancer

Name	Student ID
M.I.L. Kumari	26964
M.A.M. Sajeeth	28026
R.T. Dulmina	27999
O.V.D.K.R. Weerasena	27986

Table Content

1. Introduction

Lung cancer remains a significant health concern globally, necessitating improved predictive models for early detection and intervention. This study aims to develop a machine learning-based decision tree algorithm to accurately predict the likelihood of individuals developing lung cancer. the model integrates various patient-specific factors including demographics, medical history, smoking habits, lifestyle, and genetic markers. Image preprocessing techniques, including geometric mean filtering and K-means segmentation, enhance image quality and enable precise identification of impacted lung regions. Subsequently, machine learning classification methods such as boxplot, heatmap, and Decision Trees (DT) are employed for predictive analysis. Results demonstrate that the decision tree algorithm outperforms other methods, notably achieving higher accuracy in predicting lung cancer risk. By facilitating early identification of high-risk individuals, this predictive model offers healthcare providers a valuable tool for proactive intervention, potentially leading to improved patient outcomes and reduced healthcare costs associated with advanced cancer treat.

2. Problem Background

Despite advancements in medical science, early detection of lung cancer remains elusive, often leading to diagnoses at advanced stages with limited treatment options and poor prognosis. The insidious nature of the disease underscores the critical need for predictive models capable of identifying individuals at heightened risk of developing lung cancer.

Several risk factors contribute to the development of lung cancer, including but not limited to, smoking history, environmental exposures, genetic predisposition, and comorbidities. Additionally, demographic factors such as age, gender, obesity, and ethnicity may influence susceptibility to the disease. Traditional methods of risk assessment rely heavily on clinical observations and self-reported data, which may be subjective and prone to bias.

Leveraging vast datasets and sophisticated algorithms, we have endeavored to create predictive models capable of accurately forecasting cancer. The development of a robust predictive model for lung cancer poses unique challenges.

By harnessing the power of machine learning we aim to develop lung cancer-level predictive models

3. Problem Statement

The problem statement revolves around developing a machine learning model that can accurately predict the likelihood of an individual developing lung cancer based on relevant medical data. By analyzing various factors such as patient demographics, medical history, smoking history, lifestyle habits, and genetic markers, we aim to create a predictive model that can identify individuals at high risk of developing lung cancer. This model will enable healthcare providers to intervene earlier, potentially improving patient outcomes and reducing healthcare costs associated with late-stage cancer treatment.

4. Project Objectives

4.1. Main Objective

- Develop a lung cancer predictive model.
  The main objective is to develop a machine learning model that is capable of accurately predicting cancer levels for patients by leveraging historical data and using forecasting techniques.

4.2. Sub Objectives

- Evaluate the performance.
  Evaluate the performance of the model using appropriate metrics such as accuracy, sensitivity, specificity, and area.
- Aims to identify individuals at high risk of developing lung cancer.
- Explore the data to identify key features and patterns associated with the development of lung cancer.

5.Theoretical Background

5.1. Decision Tree Algorithm Classification

While various statistical methods, including regression, are valuable in cancer-level prediction, decision tree algorithms offer a unique approach characterized by simplicity, interpretability, and robustness. Decision trees recursively split the data based on different features to create a tree-like structure, facilitating decision-making processes. In cancer prediction, decision tree algorithms can efficiently handle complex interactions between variables and provide accurate predictions.

Despite the diverse array of statistical methods employed, the project maintains a focus mainly on decision tree algorithms in achieving accuracy and reliability in cancer predictions.

5.2. Confusion Matrix

	Predicted Positive	Predicted Negative	
Actual Positive	TP *True Positive*	FN *False Negative*	Sensitivity $\frac{TP}{(TP+FN)}$
Actual Negative	FP *False Positive*	TN *True Negative*	Specificity $\frac{TN}{(TN+FP)}$
	Precision $\frac{TP}{(TP+FP)}$	Negative Predictive Value $\frac{TN}{(TN+FN)}$	Accuracy $\frac{TP+TN}{(TP+TN+FP+FN)}$

- Accuracy- The proportion of correctly classified instances out of the total instances.
- Precision- The proportion of true positive predictions out of all positive predictions made by the model.
- Recall (Sensitivity)- The proportion of true positive predictions out of all actual positive instances.
- Specificity- The proportion of true negative predictions out of all actual negative instances.
- F1 Score- The harmonic mean of precision and recall, providing a balance between the two metrics.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Interpretation:
- A high accuracy indicates overall good performance of the model.
- Precision is important when the cost of false positives is high.
- Recall is important when the cost of false negatives is high.
- F1 Score provides a balanced assessment of precision and recall.

By analyzing the confusion matrix and computing these evaluation metrics, we can assess the effectiveness of the classification model and make informed decisions about its performance and potential improvements.

5.3. Correlation Coefficient

$$r = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

- r is the correlation coefficient, which quantifies the strength and direction of the linear relationship between two variables x and y.
- x_i and y_i represent individual data points for variable x and y, respectively, within the dataset. These data points are paired observations from the two variables.
- $\bar{x}$ and $\bar{y}$ are the means (or averages) of variables x and y, respectively. They represent the central tendency of the data points within each variable.
- n is the total number of paired observations or data points in the dataset.

Calculate the correlation coefficient between different variables in our dataset. For example, calculate the correlation between smoking history and cancer level, between genetic markers and cancer risk, or between age and cancer incidence. A correlation coefficient close to 1 indicates a strong positive correlation, meaning that as one variable increases, the other variable also tends to increase. A coefficient close to -1 indicates a strong negative correlation, meaning that as one variable increases, the other tends to decrease. A coefficient close to 0 indicates little to no correlation. Overall, incorporating the correlation coefficient into our analysis can help us better understand the relationships between different variables and improve the accuracy and interpretability of the lung cancer prediction model.

6. Methodology

6.1. Data Collection Methods

a. Source

We will obtain the secondary dataset from the Kaggle website, covering the period from 2013 to 2018. A reputable source known for its diverse collection of datasets. These datasets are meticulously curated and vetted, ensuring reliability and relevance to our research objectives. Leveraging secondary data not only expedites the research process but also provides valuable insights into lung cancer detection trends and patterns.

Link: https://nsbm365-my.sharepoint.com/:x:/g/personal/milkumari_students_nsbm_ac_lk/EXFc6ZDh8H1JmzmqMEDEgNkBJkXzoy-IoIk6qlk_-d7Cjg?e=OmEQPN

b. Data Quality

Ensure the dataset's quality through rigorous validation processes, including outlier detection and missing data imputation.

6.2.Tools and Techniques

a. Programming Language

The primary programming language for data analysis, machine learning, and cancer level forecasting will be Python. Since it incorporates the analysis's rich ecosystem of libraries.

b. Libraries

use libraries for data manipulation, visualization, and machine learning methods, such as Pandas, NumPy, Matplotlib, Seaborn, and Scikit-learn.

[1]:
```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import confusion_matrix,accuracy_score,f1_score,precision_score,recall_score, ConfusionMatrixDisplay,classification_report
from sklearn import tree
import matplotlib.pyplot as plt
import seaborn as sns
```

6.3.Data Preprocessing

6.3.1. Remove the Unwanted Columns

```
[9]: # remove the unwanted columns
     df = df.loc[:,df.columns[2:26]]
```

identified the unwanted columns and removed them from the dataset. In this case, remove columns with index 0 and 1 (representing "index" and "patient Id" columns) using Data Frame manipulation techniques. Now, our data set has 24 columns.

6.3.2. Missing Data Handling

Implement techniques such as imputation to handle missing data.

```
[11]: # check the number of null values
      df.isnull().sum()
```

```
[11]: Age                        0
      Gender                     0
      Air Pollution              0
      Alcohol use                0
      Dust Allergy               0
      OccuPational Hazards       0
      Genetic Risk               0
      chronic Lung Disease       0
      Balanced Diet              0
      Obesity                    0
      Smoking                    0
      Passive Smoker             0
      Chest Pain                 0
      Coughing of Blood          0
      Fatigue                    0
      Weight Loss                0
      Shortness of Breath        0
      Wheezing                   0
      Swallowing Difficulty      0
      Clubbing of Finger Nails   0
      Frequent Cold              0
      Dry Cough                  0
      Snoring                    0
      Level                      0
      dtype: int64
```

Our dataset does not have null values. The “. isnull().sum()" method is called on this Data Frame, and the result shows that there are 0 null values in each columns.

6.3.1. Data Type

As we choose to use a heatmap for EDA part, alter the data types for the parameters.

[7]:
```
# Data type of each columm
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 26 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   index                     1000 non-null   int64
 1   Patient Id                1000 non-null   object
 2   Age                       1000 non-null   int64
 3   Gender                    1000 non-null   int64
 4   Air Pollution             1000 non-null   int64
 5   Alcohol use               1000 non-null   int64
 6   Dust Allergy              1000 non-null   int64
 7   OccuPational Hazards      1000 non-null   int64
 8   Genetic Risk              1000 non-null   int64
 9   chronic Lung Disease      1000 non-null   int64
 10  Balanced Diet             1000 non-null   int64
 11  Obesity                   1000 non-null   int64
 12  Smoking                   1000 non-null   int64
 13  Passive Smoker            1000 non-null   int64
 14  Chest Pain                1000 non-null   int64
 15  Coughing of Blood         1000 non-null   int64
 16  Fatigue                   1000 non-null   int64
 17  Weight Loss               1000 non-null   int64
 18  Shortness of Breath       1000 non-null   int64
 19  Wheezing                  1000 non-null   int64
 20  Swallowing Difficulty     1000 non-null   int64
 21  Clubbing of Finger Nails  1000 non-null   int64
 22  Frequent Cold             1000 non-null   int64
 23  Dry Cough                 1000 non-null   int64
 24  Snoring                   1000 non-null   int64
 25  Level                     1000 non-null   object
dtypes: int64(24), object(2)
memory usage: 203.3+ KB
```

The data type of the column "Patient ID" and “Level” are "object”. (we remove the “patient ID” column) In Pandas, "object" is often used to represent string data types. The data type of all other columns is "float64". This indicates that the columns contain numerical data.

[14]:
```
# Define the mapping for encoding
level_mapping = {'Low': -1, 'Medium': 0, 'High': 1}

# Encode the values in the "Level" column using the mapping
df['Level'] = df['Level'].map(level_mapping)
```

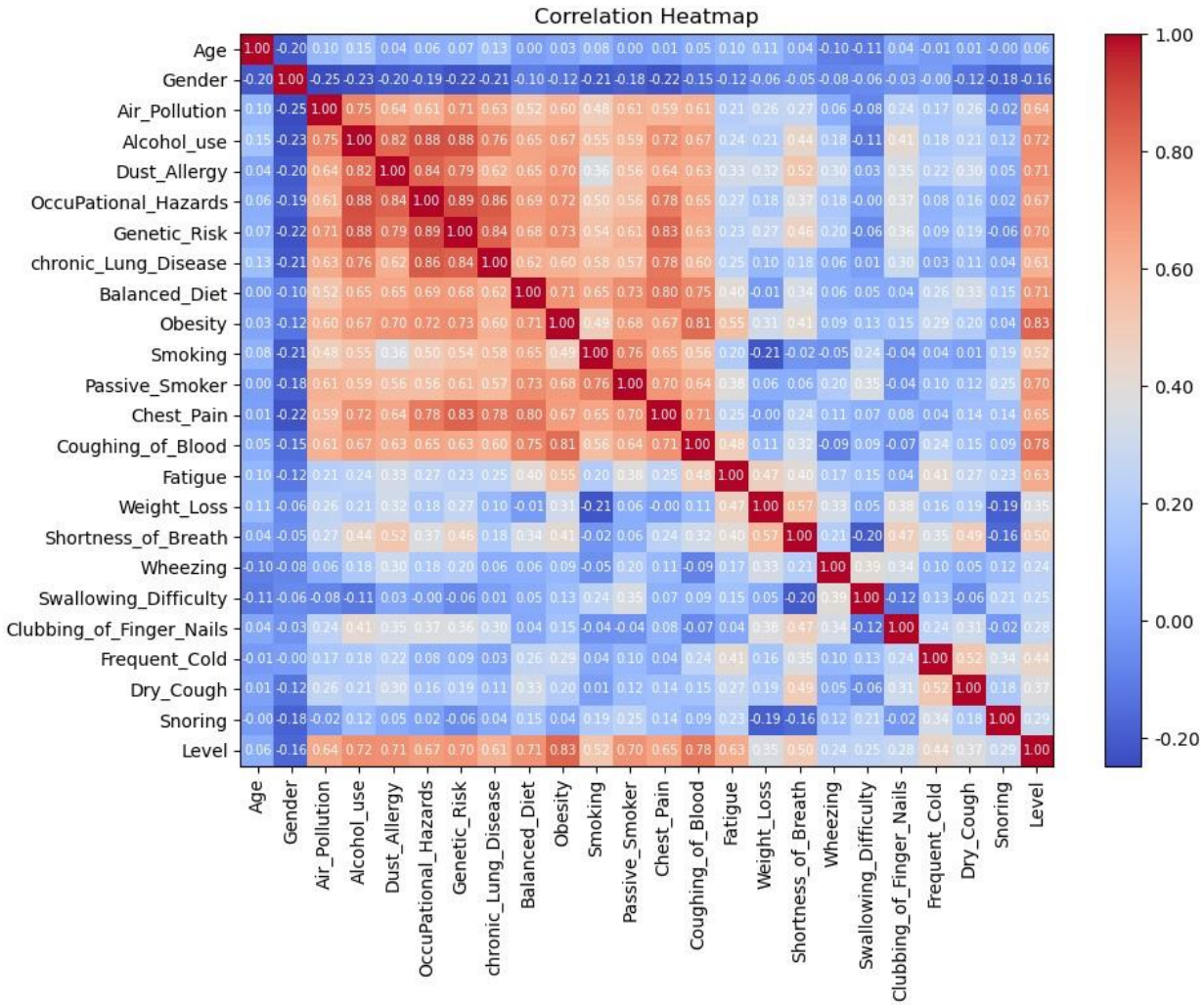
Does not use string-type data drawing heatmap, because converts the ‘Level’ column into numerical format.

6.4. Exploratory Data Analysis (EDA)

Use EDA to learn about patterns, correlations, and data distributions.

- Correlation Heatmap

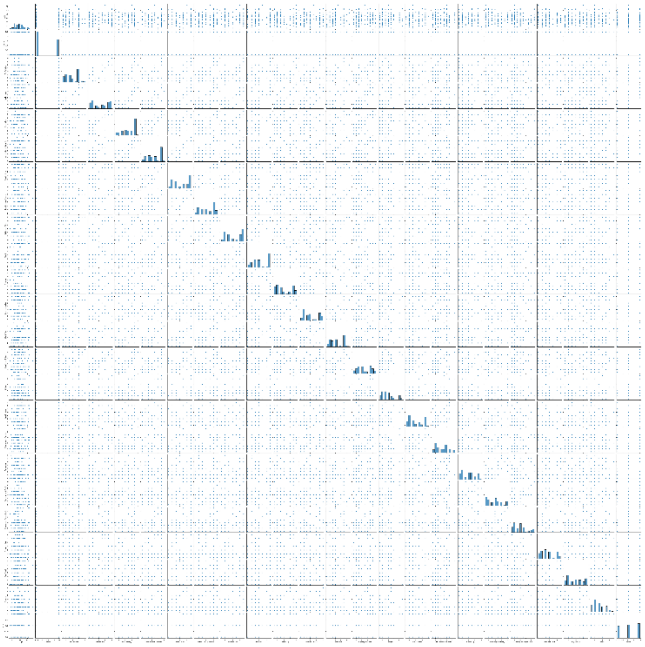
```
[16]: # create heapmap
plt.figure(figsize=(10,8))
plt.imshow(corr_matrix,cmap='coolwarm',aspect='auto')
plt.colorbar(format='%.2f')
plt.title('Correlation Heatmap')
for (i, j), val in np.ndenumerate(corr_matrix):
    plt.text(j, i, f'{val:.2f}', ha='center', va='center', color='white', fontsize=7)
plt.xticks(range(len(corr_matrix)), corr_matrix.columns, rotation=90)
plt.yticks(range(len(corr_matrix)), corr_matrix.columns)
plt.tight_layout()

plt.show()
```



Ideal for identifying relationships between variables. Our target variable is 'Level' so the 'Level' variable is the y-axis when drawing graphs. For that, we can find the x-axis variable that has more relationships with the level using the heat map.

- Pair Plot

```
[18]: # Pair plot
      sns.pairplot(df)
```

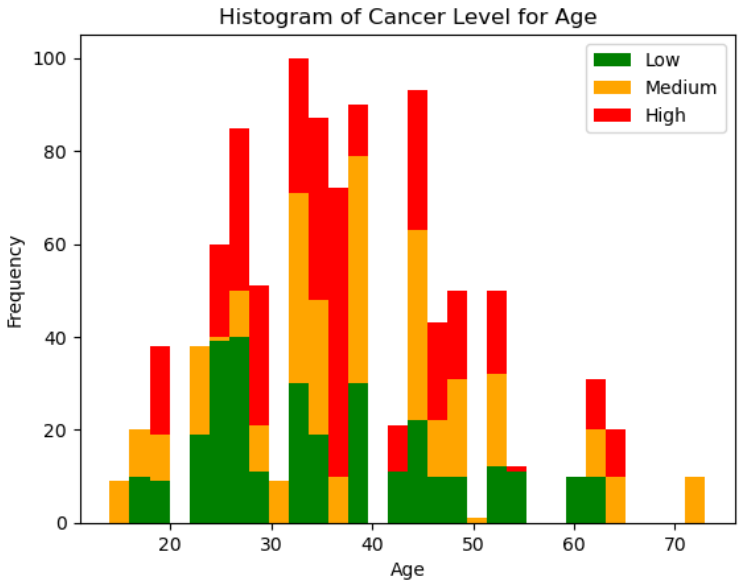


we utilize pair plots to visually explore the relationships between multiple variables in our dataset.

Pair plots, also known as scatterplot matrices, display pairwise relationships between variables in a dataset. Each scatterplot in the matrix shows the relationship between two variables, with one variable plotted on the x-axis and the other on the y-axis.

- Histogram

```
[24]: # Plot the histogram with segmented bars
plt.hist([cancer_patient[cancer_patient['Level'] == 'Low']['Age'],
          cancer_patient[cancer_patient['Level'] == 'Medium']['Age'],
          cancer_patient[cancer_patient['Level'] == 'High']['Age']],
         bins=30, stacked=True, color=['green', 'orange', 'red'], label=['Low', 'Medium', 'High'])

# Set labels and title
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.title("Histogram of Cancer Level for Age")

# Add legend
plt.legend()

# Show plot
plt.show()
```



We constructed a histogram to visualize the distribution of cancer levels across different age groups in our dataset.

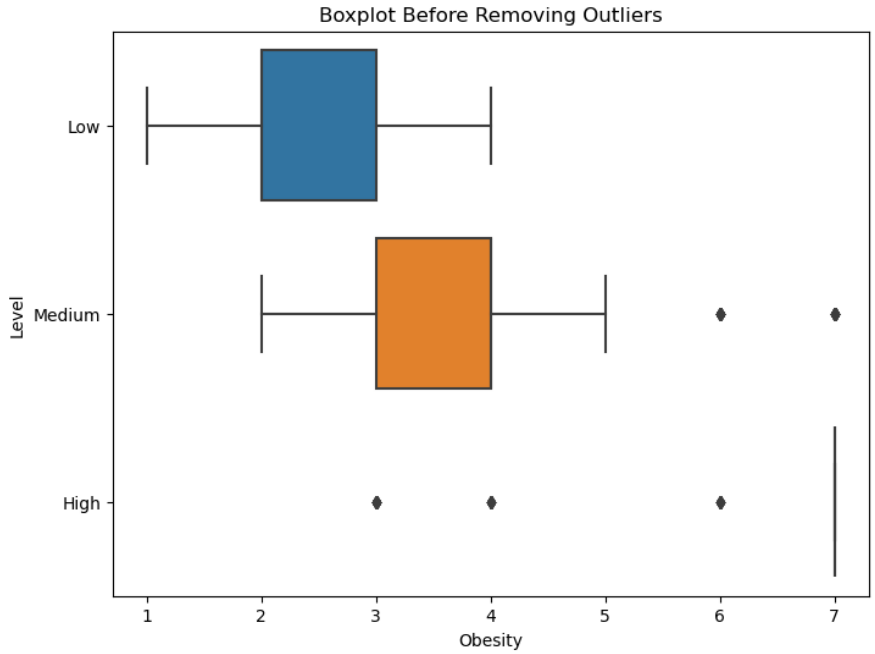
A histogram is a graphical representation of the distribution of numerical data. It consists of bars where the height of each bar represents the frequency or count of data points falling within a specific range or bin.

By constructing a histogram of cancer levels for age, we gain insights into the distribution of cancer levels across different age groups in our dataset. This visualization helps us understand how cancer levels vary with age and identify potential patterns or trends in the data. It serves as a valuable exploratory tool in our analysis, providing insights that inform further investigation and modeling decisions.

- Outlier Detection. (Using Box Plot)

Boxplot Before Removing Outliers

[23]:
```
# Set the figure size
plt.figure(figsize=(8, 6))  # Adjust the size as needed
plt.title('Boxplot Before Removing Outliers')
# Create the boxplot
sns.boxplot(x='Obesity', y='Level', data=df)
# Show the plot
plt.show()
```

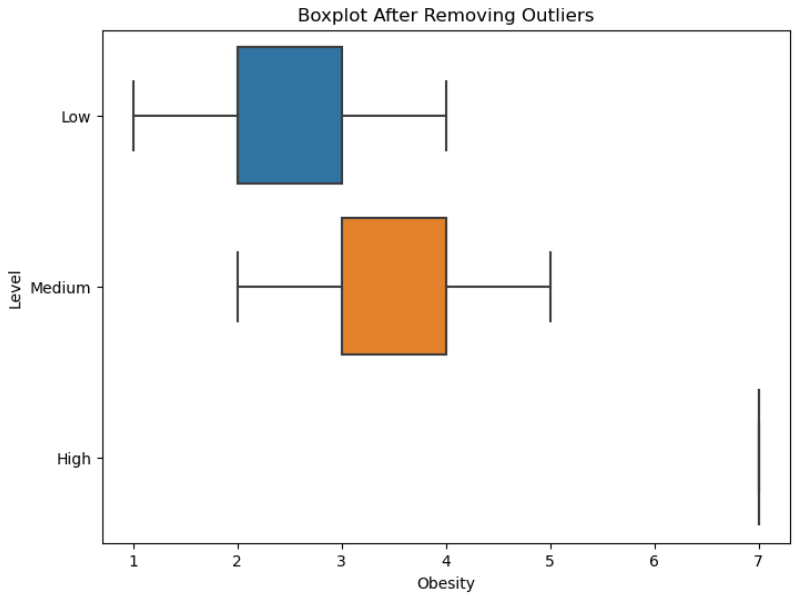


We utilized a heatmap to identify the strongest relationships between variables in our dataset. Through this analysis, we identified a significant correlation between obesity and cancer levels. To further explore this relationship, we drew a box plot using these two variables.

- Calculate Outliers & Remove

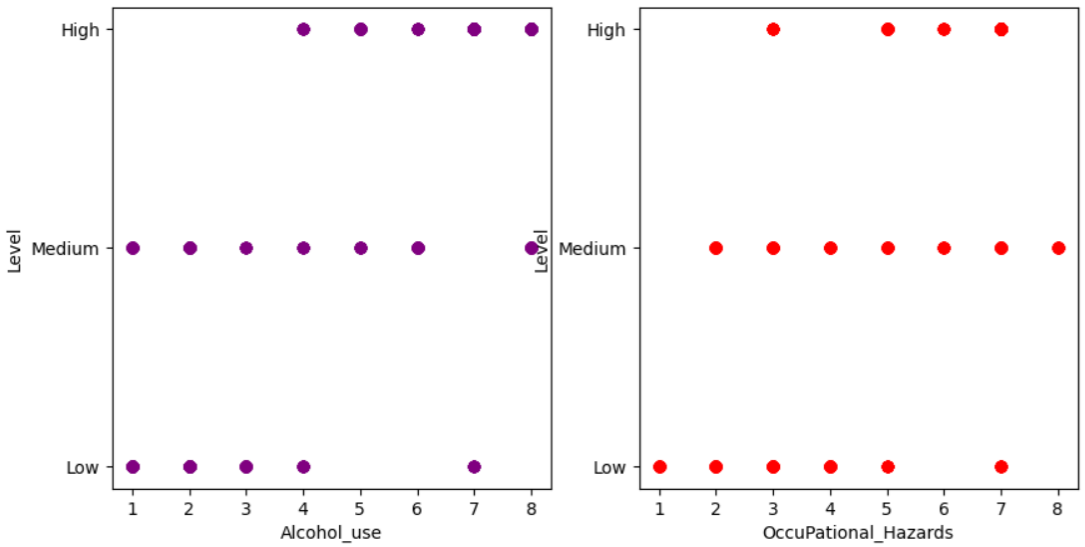
```
[24]: Q1=df['Obesity'].quantile(0.25)
      Q3=df['Obesity'].quantile(0.75)
      IQR=Q3-Q1

      #identify outliers
      outliers=((df['Obesity']< Q1-1.5*IQR) | df['Obesity'] > Q3+1.5*IQR)

      #count the number of outliers
      outliers.sum()
```

```
[36]: df=df[~outliers]
```



Our initial exploration of the data using a boxplot revealed the presence of 5 outliers. These outliers are data points that fall outside the expected range of the data distribution. They can have a significant impact on statistical measures like the mean and can potentially mislead any analysis. We decided to address these outliers by removing them from the dataset.

- Scatter Plot

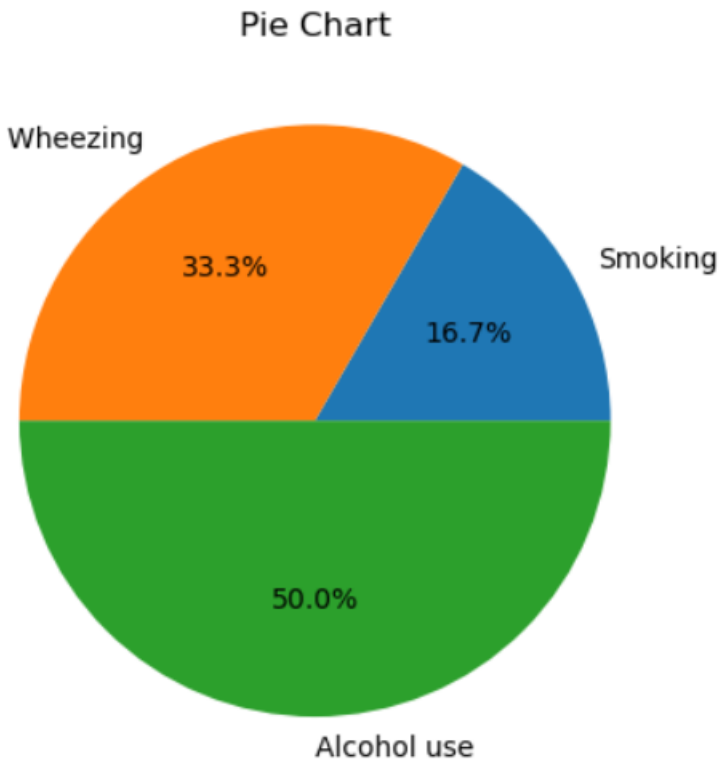


This scatter plot shows data points (represented as purple dots) scattered across different levels of alcohol use.
The y-axis is divided into three categories: “Low”, “Medium”, and “High”, indicating levels of risk or intensity.
Other scatter plot shows data points (represented as red dots) scattered across various levels of occupational hazards.
Like the first plot, the y-axis is divided into “Low”, “Medium”, and “High” categories.

- Pie chart

We selected 3 of the most impactful symptoms/conditions from the correlation heatmap and created a pie chart analyzing them.
According to the pie chart, representing 50% of the total patients among the three categories, alcohol use is the most prevalent.
smoking is the least common habit among the three categories in the population which is represented by 16.7%.

[25]:
```
# pie chart
values= [1,2,3]
labels=['Smoking','Wheezing ','Alcohol use',]
# Create a pie chart
plt.pie(values, labels=labels, autopct='%1.1f%%')
plt.title("Pie Chart")
plt.show()
```



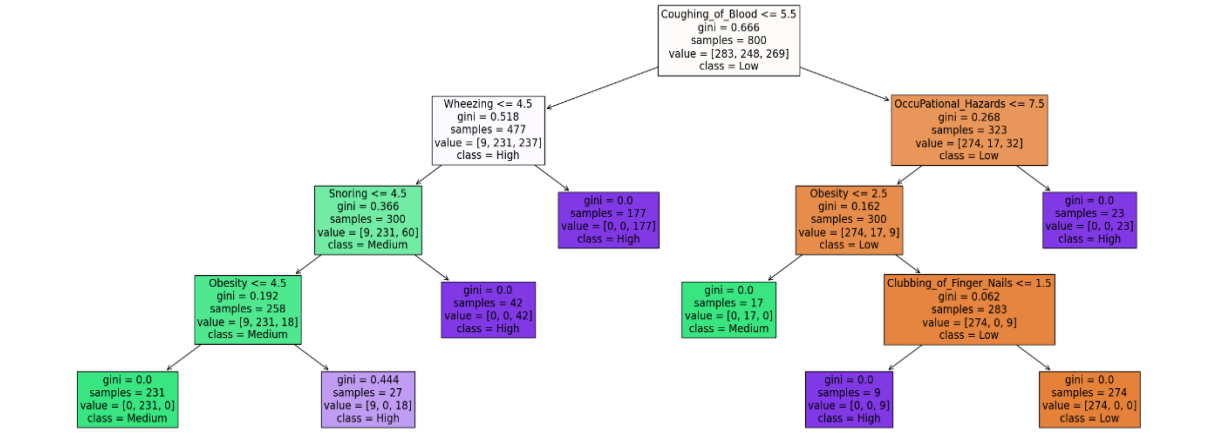
6.5. Data Modeling

6.5.1. Machine Learning Model

Decision Tree Classification Algorithm

Jupyter Notebook Link: https://github.com/IsuriNSBM/DT_Classification.git

```
[29]: dt = DecisionTreeClassifier(max_depth=4)
      dt.fit(X_train,y_train)

[29]: DecisionTreeClassifier
      DecisionTreeClassifier(max_depth=4)

[30]: feature_names = X.columns
      labels = y.unique()

[31]: plt.figure(figsize=(30,10),facecolor='white')
      a = tree.plot_tree(dt,feature_names=feature_names,class_names=labels,filled=True,fontsize=14)
      plt.show()
```



This decision tree is used for classification based on certain medical symptoms and conditions. It starts with the root node.
of "Coughing of Blood <= 5.5". If this condition is true, it moves to the left branch. If false, it considers.
"Occupational Hazards" and then moves down. Each node provides a classification based on the gini impurity, number of samples,
value array representing class distribution, and the resulting class. Each node in the tree represents a decision based on a condition.
The tree branches out based on whether the condition is met (<=) or not (>), leading to different outcomes. The final classification
is determined by the path taken through the tree based on the conditions at each node. The colors are based on the level of cancer.

In our project, we utilized the Decision Tree (DT) classification algorithm as our machine learning model to predict cancer levels based on relevant features.

Decision Tree Classifier Overview

A Decision Tree Classifier is a machine learning algorithm that makes decisions based on asking a series of questions. Given a set of input features, the model splits the data into subsets using rules inferred from the data. These rules are represented as nodes in the tree, where each node corresponds to a feature in the dataset that provides the best split to classify the data based on some metric like Gini impurity.

Generally, there is an inverse relationship between Gini impurity and Information Gain. A feature with lower Gini impurity (i.e., more purity) is likely to have higher Information Gain, indicating that it provides more discriminatory power for classification. Therefore, the decision tree algorithm tends to select features with lower Gini impurity and higher Information Gain for splitting the data.

Model Creation and Training

The model is created using a Decision Tree Classifier with a maximum depth of 4, limiting complexity and overfitting, and trained using the fit method on the training dataset.

Visual Representation of the Decision Tree

The decision tree visualizes decision-making processes, starting with the root node representing the initial split based on a feature. Each branch offers possible answers, leading to further nodes or leaves for final class labels.

Nodes and Splits

Each node in the tree specifies:

The tree structure consists of nodes that specify features, Gini impurity index, sample distribution, and class labels. Decisions are made at each node based on the feature value and a threshold, moving to the left or right child until reaching a leaf node, which provides the classification outcome.

6.6. Optimization and Development

a. Model Evaluation

Use appropriate metrics, such as Confusion Metrix and Correlation Coefficient to evaluate the model's performance.

Model	Evaluation
Decision Tree Classification model	Confusion Metrix: [[81 0 1] [0 55 0] [0 0 63]] Accuracy: 0.995 Precision: 0.995078125 Recall: 0.995 F1 Score: 0.9950043476160572 Correlation Coefficient between Obesity and Level: 0.8274350995887104

Metrics for Evaluation

We used Mean Absolute Error, Confusion Matrix, and Correlation Coefficient to evaluate model performances.

In the evaluation of our model, we utilized several performance metrics, including a confusion matrix, accuracy, precision, recall, and F1 score. Here's how we interpret and report these results:

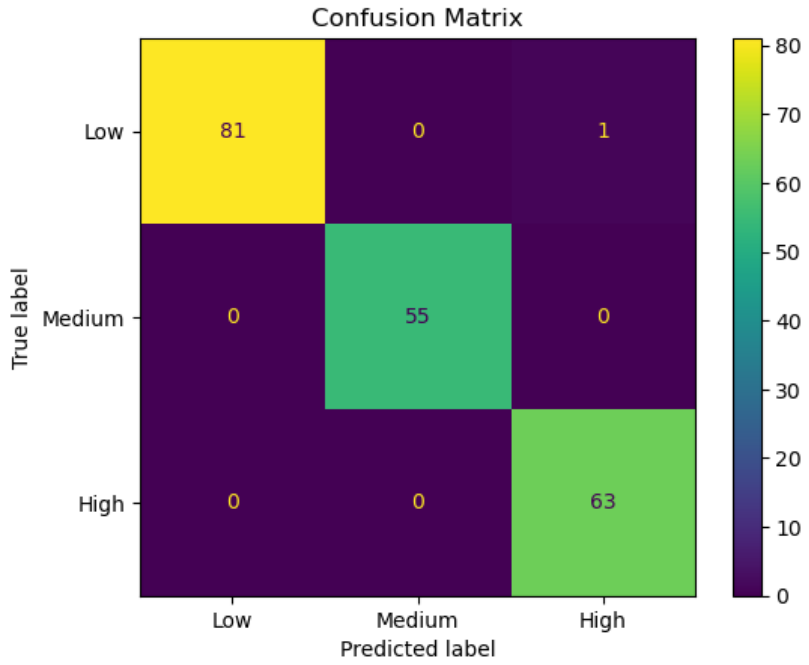
Confusion Matrix:

The confusion matrix provides a detailed breakdown of the model's predictions compared to the actual values. It is presented as a table where the rows represent the actual classes, and the columns represent the predicted classes. For our model,

[32]:
```
# evaluate model
y_pred = dt.predict(X_test)
cm = confusion_matrix(y_test,y_pred)
print("Confusion Metrix:\n",cm)
print("Accuracy: ", accuracy_score(y_test,y_pred))
print("Precision: ",precision_score(y_test,y_pred,average='weighted'))
print("Recall: ",recall_score(y_test,y_pred,average='weighted'))
print("F1 Score: " ,f1_score(y_test,y_pred,average='weighted'))
```

```
Confusion Metrix:
 [[81  0  1]
 [ 0 55  0]
 [ 0  0 63]]
Accuracy:  0.995
Precision:  0.995078125
Recall:  0.995
F1 Score:  0.9950043476160572
```

[33]:
```
disp = ConfusionMatrixDisplay(confusion_matrix=cm, display_labels=['Low','Medium','High'])
disp.plot()
plt.title('Confusion Matrix')
plt.show()
```



In the confusion matrix:

- 81 represents the number of instances where the actual class was predicted as "Low" (True Negatives).
- 55 represents the number of instances where the actual class was predicted as "Medium" (True Positives).
- 63 represents the number of instances where the actual class was predicted as "High" (True Negatives).
- 1 represents the number of instances where the actual class was "High" but predicted as "Low" (False Negative).
- 0 represents the number of instances where the actual class was "Low" but predicted as "Medium" (False Positive).
- 0 represents the number of instances where the actual class was "Medium" but predicted as "High" (False Positive).

In summary:
- The diagonal elements (81, 55, and 63) represent the correct predictions for each class.
- Off-diagonal elements represent incorrect predictions, where the row indicates the actual class and the column indicates the predicted class.

The diagonal elements represent the number of correct predictions (true positives).
Off-diagonal elements represent incorrect predictions (false positives and false negatives).
For example, there were 81 instances where the model correctly predicted a 'Low' cancer level, 55 instances where it correctly predicted a 'Medium' level, and 63 instances where it correctly predicted a 'High' level.

Accuracy:
Accuracy is the proportion of correct predictions out of the total predictions made by the model. It is calculated as the ratio of the sum of true positives and true negatives to the total number of predictions. In our case, the accuracy is 0.995, indicating that the model correctly predicted 99.5% of the cases.

Precision:
Precision is the proportion of true positive predictions out of all positive predictions made by the model. It is calculated as the ratio of true positives to the sum of true positives and false positives. In our case, the precision is 0.995, indicating that 99.5% of the positive predictions made by the model were correct.

Recall:
Recall, also known as sensitivity, is the proportion of true positive predictions out of all actual positive cases in the dataset. It is calculated as the ratio of true positives to the sum of true positives and false negatives. In our case, the recall is 0.995, indicating that 99.5% of the actual positive cases were correctly identified by the model.

F1 Score:
The F1 score is the harmonic mean of precision and recall. It provides a balance between precision and recall and is useful when dealing with imbalanced datasets. In our case, the F1 score is 0.995, indicating a high level of model performance considering both precision and recall.
Overall, these evaluation metrics demonstrate the effectiveness of our model in accurately predicting cancer levels, with high precision, recall, and F1 score, and a near-perfect accuracy rate.

Correlation Coefficient:

[35]:
```
import numpy as np

# Assuming df is your DataFrame containing the 'Obesity' and 'Level' columns

correlation_coefficient = np.corrcoef(df['Obesity'], df['Level'])

print("Correlation Coefficient between Obesity and Level:", correlation_coefficient[0, 1])
```

```
Correlation Coefficient between Obesity and Level: 0.8274350995887104
```

This correlation coefficient r will range between -1 and 1, where:

r = 1 indicates a perfect positive linear relationship,
r = -1 indicates a perfect negative linear relationship, and
r = 0 indicates no linear relationship between the variables.
For our report, we will interpret the correlation coefficient to understand the degree and direction of the relationship between cancer severity (Level) and obesity. A positive r value suggests a positive relationship, indicating that as obesity increases, cancer severity tends to increase. Conversely, a negative r value implies a negative relationship, suggesting that as obesity increases, cancer severity tends to decrease. Finally, a r value close to 0 indicates a weak or no linear relationship between the variables.

7. Project Timeline

Weeks	1	2	3	4	5	6	7	8	9
Identify the Problem									
Data Collection and Preprocessing									
Exploratory Data Analysis (EDA)									
Data Modeling									
Model Development and Optimization									
Documentation and Final Report									

8. References

1. Jay Kumar Raghavan Nair, Umar Abid Saeed, Connor C. McDougall, Ali Sabri, Mmed, Bojan Kovacina, B. V. S. Raidu, Riaz Ahmed Khokhar, Stephan, Vera Hirsh, Chankowsky Jeffrey, Leon C. Van Kempen, and Jana Taylor, "Radiogenomic Models Using Machine Learning Techniques to Predict EGFR Mutations in Non-Small Cell Lung Cancer", https://doi.org/10.1177/0846537119899526, The Author(s) 2020.

2. Kun-Hsing Yu, Tsung-Lu Michael Lee, Ming-Hsuan Yen, S C Kou, Bruce Rosen, Jung-Hsien Chiang and Isaac S Kohane, "Reproducible Machine Learning Methods for Lung Cancer Detection Using Computed Tomography Images: Algorithm Development and Validation", http://dx.doi.org/10.2196/16709 , JOURNAL OF MEDICAL INTERNET RESEARCH, J Med Internet Res 2020.

3. Genomics, Proteomics & Bioinformatics, Volume 20, Issue 5, October 2022, Pages 850–866, https://doi.org/10.1016/j.gpb.2022.11.003 Published: 01 December 2022

4. Janee Alam, Sabrina Alam and Alamgir Hossan, "Multi-Stage Lung Cancer Detection and Prediction Using Multi-class SVM Classifier", 2019.

5. www.javatpoint.com. (n.d.). Machine Learning Tutorial | Machine Learning with Python - Javatpoint. [online] Available at: https://www.javatpoint.com/machine-learning

9. Contribution

Name	Student ID	Contribution
M.A.M. Sajeeth	28026	Introduction, Problem background and statement, objectives & Visualization part (pie chat, Scatter Plot, Box plot)
RT Dulmina	27999	Theoretical background DT algorithm
OVDKR Weerasena	27986	Data Preprocessing & Evaluate Model
DMIL Kumari	26964	EDA, Machine Learning Model(create DT classification algorithm), Evaluate Model & Documentation and Final Report