Kinna Zhao
Priyonto Saha
Yacine Marouf

Datathon 1: Analysis of Lung Cancer and BMI Risk Factors

Introduction:

In Canada, lung cancer is a leading cause of cancer-related death, taking away the life of over 21,000 people per year (Statistics Canada, 2022). On the other hand, obesity, which is measured with a body mass index (BMI) above 30.0, increases the risk of health conditions such as cardiovascular diseases and adds financial burden to the healthcare system (Wharton et al., 2020; Statistics Canada, 2019). Lung cancer and obesity are both diseases with multiple behavioural, environmental, and hereditary factors. Thus the goal of this report is to analyse and visualise a lung cancer dataset from Ethiopia and a BMI dataset from Canada and predict the possible severity of cancer and rate of obesity through the use of linear regressions and K-nearest-neighbours (KNN).

Data Engineering Process

Due to our intent to use both linear regression and KNN as methods to predict cancer severity and BMI levels, we utilised descriptive statistics, histograms, and statistical tests to check for the distributions and scale of the datasets. Since KNN is a distance based non-parametric model, the accuracy of the model is sensitive to scaling of the data, as variables with larger ranges will dominate the variables with smaller ranges.

We first checked the descriptive statistics and histograms of both datasets to get a more thorough view of the data's distribution and magnitude. Next, we tested the normality of the data through the wilks Shapiro test to ensure that a regression would still be a robust prediction model. From this, we can then determine the efficacy of KNN and linear regressions as possible models. Finally, we used a correlation matrix to determine the relationship among the different variables and to remove any variables with 0 correlation. This would reduce the dataset's multidimensionality and improve model performance.

Analysis

The descriptive statistics for the BMI dataset showed a large disparity in the magnitude of the factors, which would necessitate scaling or normalisation for KNN to be effective. When testing for normality, only height and weight were normally distributed. The histograms further confirmed the distributions not being Gaussian. Finally, when looking at the correlation matrix, no factors other than height and weight seem to be correlated to BMI. However, this may be due to BMI being directly calculated through the height and weight of the patients, which introduces multicollinearity and would overpower any other possible correlation (supplementary figure 1). Due to the large range of scales and the non normal distribution among the different factors, the use of both linear regression or KNN would result in inaccurate models. As such, we decided to

not explore the BMI data further. Due to the data being mostly categorical for the Cancer dataset, we did not consider a linear regression. Furthermore, we used minMaxScaler instead of standardScaler since the dataset does not follow a normal distribution and the predictors are all ordinal with similar ranges. Since minMaxScaler does not rely on the normality assumption, it allows for better scaling of the data than standardScaler. When looking at the correlation matrix, age and gender had no correlation with the rest of the variables, Therefore, age and gender were removed to improve accuracy and model performance (supplementary figure 2).

Findings

Based on the elbow method, K values of 1 to 5 resulted in the lowest errors, so the model was fixed at K = 2 . When testing across multiple different random states, the confusion matrices would never return false negatives but would return some positives across the predicted cancer rates on actually healthy people (although the rate of false positives decreased as the random state increased)(supplementary figure 3). The recall values were always above 0.95, with most around 1.00. However, the precision scores would tend to be slightly lower (however still above 90%). This overall resulted in very high f1 scores of at least 0.97 across all models regardless of the starting random states. These results imply that the model can very accurately cluster the cancer severities based on the 4 groups and that they each are very defined in their factors. However, the lower precision to recall rates also imply that there are some crossovers between healthy people and people who have cancer where some healthy people would experience similar levels of risk factors as those with cancer. The confusion matrix supports this since a few actually healthy people would always be predicted to have some level of lung cancer.

Conclusion

Based on our KNN model, health practitioners can confidently utilise these risk factors to predict the cancer severity of a patient. Moreover, since the model had a low rate of false negatives, they can be confident that they would not miss some possible cancer diagnoses within the population. However, due to the weaker precision of the model, there may be some within the population who, although healthy, would flag as possibly having lung cancer. This may increase the possibility of additional diagnostics being done on healthy patients, which can cause additional healthcare stress on populations with weaker public health systems such as the ones found in lower- and middle-income countries like Ethiopia (the source of this dataset). These results also imply that there may be some unknown factors that would reduce the prevalence of cancer among people who live with high risk factors. Overall, this model would be a great benefit to use in the Ethiopian population to reduce lung cancer related burden of disease and can be used as justification for further research on factors that prevent lung cancer in Ethiopia.

References

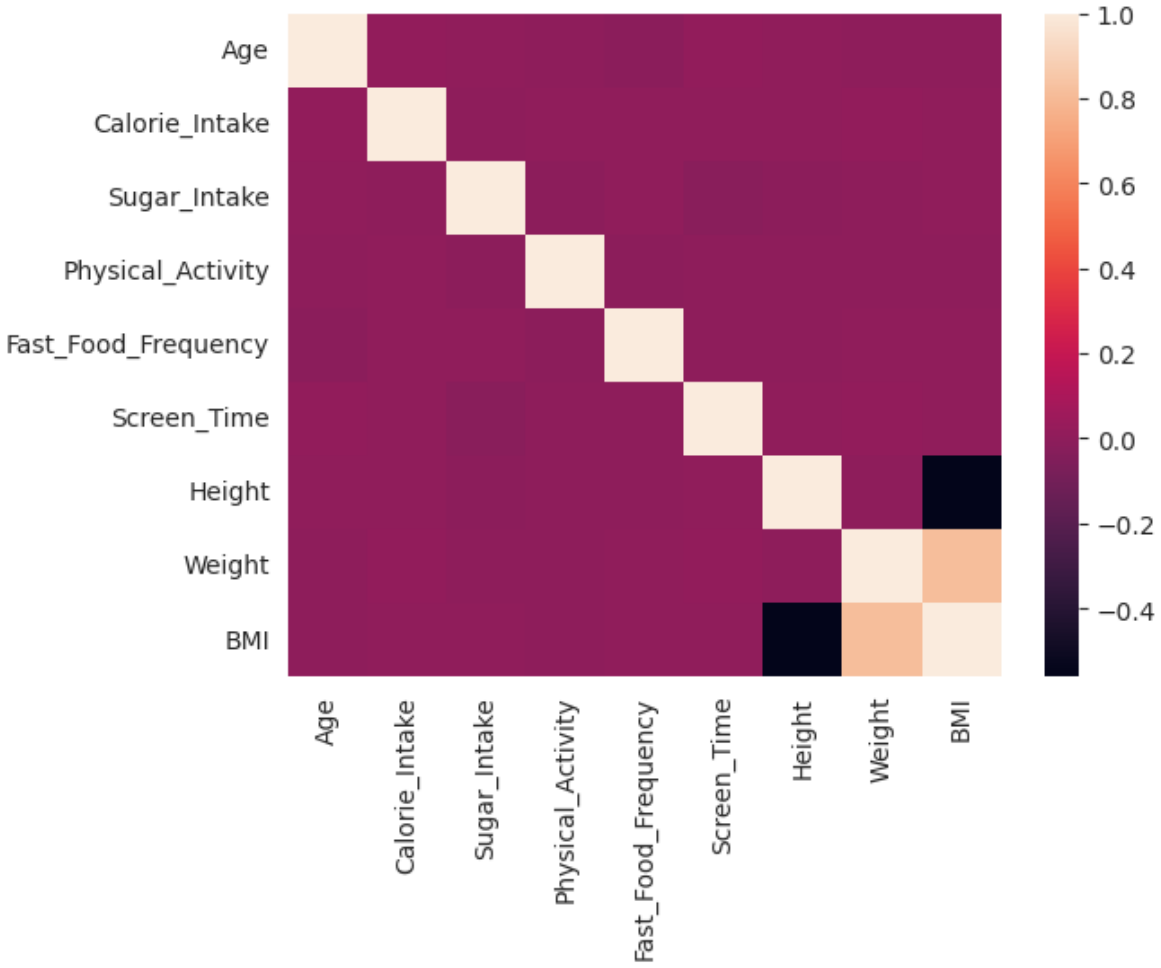
Government of Canada, S. C. (2022, January 4). *Lung cancer is the leading cause of cancer death in Canada*. https://www.statcan.gc.ca/o1/en/plus/238-lung-cancer-leading-cause-cancer-death-canada

Government of Canada, S. C. (2019, June 25). *Overweight and obese adults, 2018*. https://www150.statcan.gc.ca/n1/pub/82-625-x/2019001/article/00005-eng.htm

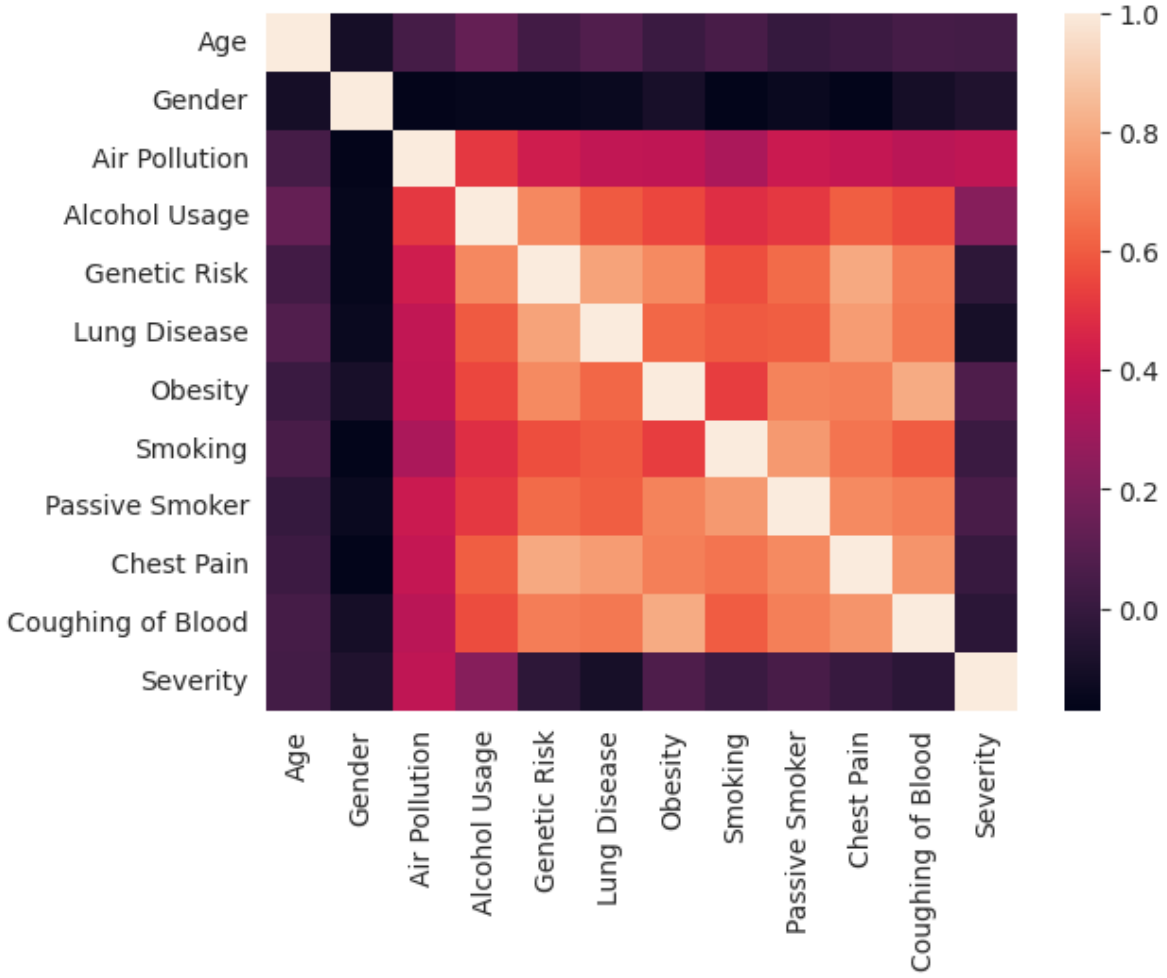
Wharton, S., Lau, D. C. W., Vallis, M., Sharma, A. M., Biertho, L., Campbell-Scherer, D., Adamo, K., Alberga, A., Bell, R., Boulé, N., Boyling, E., Brown, J., Calam, B., Clarke, C., Crowshoe, L., Divalentino, D., Forhan, M., Freedhoff, Y., Gagner, M., … Wicklum, S. (2020). Obesity in adults: a clinical practice guideline. *Canadian Medical Association Journal*, *192*(31), E875–E891. https://doi.org/10.1503/cmaj.191707

Appendix

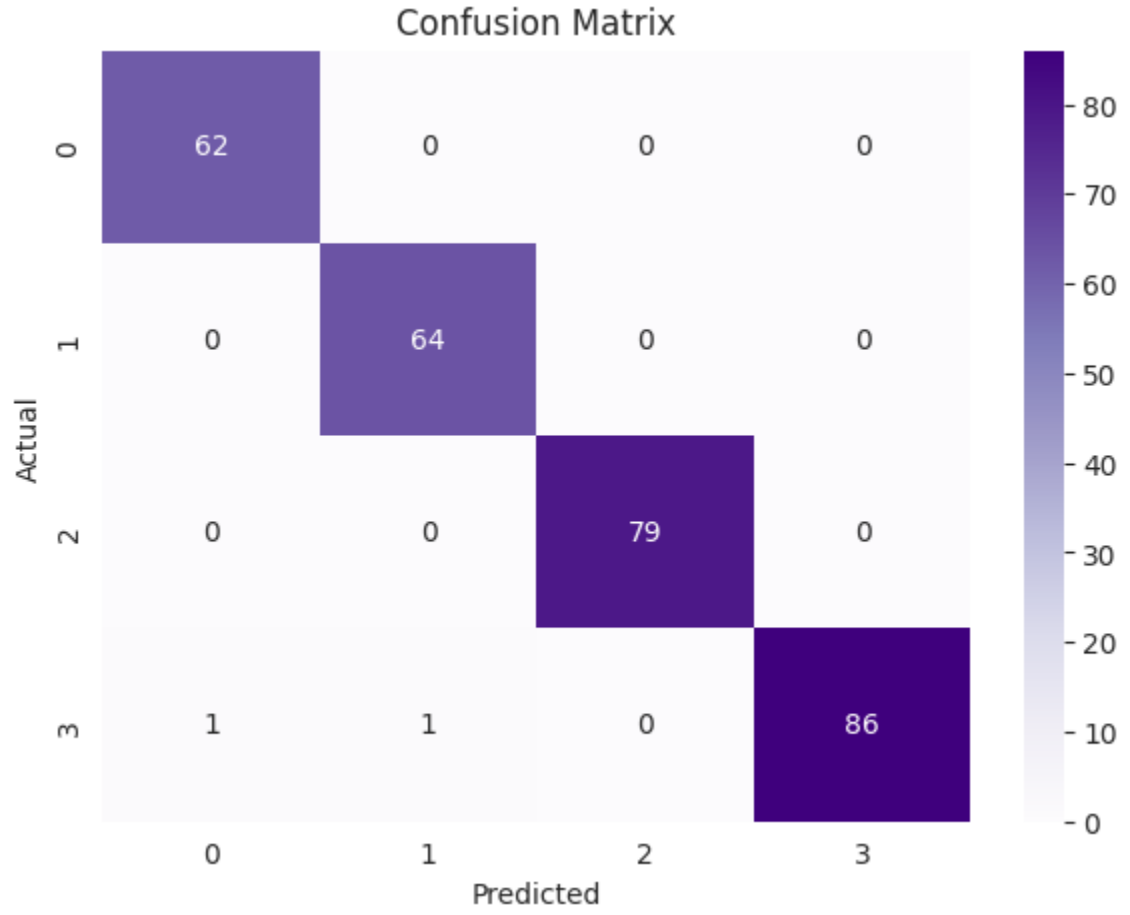
Supplementary Figure 1: Correlation Matrix for BMI Data



Supplementary Figure 2: Correlation Matrix for Lung Cancer Data



Supplementary figure 3: Confusion Matrix for Lung Cancer Data



Individual Contributions

Kinna Zhao : data preprocessing and visualisation, exploratory data analysis
Priyonto Saha: exploratory data analysis, model design and analysis
Yacine Marouf: model analysis, writing report+presentation

Github Repository with Presentation and Jupiter Notebook

https://github.com/P-Saha/k-NN_and_Clustering_Analysis

Setup and Data Import

In the obsesity dataset, the following variables were continuous: Age, Calorie_Intake, Sugar_Intake, Physical_Activity,Fast_Food_Frequency, Screen_Time, Height, Weight, BMI. These variables exhibited notable differences in magnitude. Therefore, to avoid potential biases towards variables with higher magnitudes, all continous variables were normalized before applying the KNN algorithm.

```
import numpy as np
import pandas as pd
import seaborn as sns
import statsmodels.api as sm
import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn import preprocessing
from sklearn.neighbors import KNeighborsClassifier
from sklearn.cluster import KMeans
from sklearn.metrics import classification_report, confusion_matrix, accuracy_score
from statsmodels.miscmodels.ordinal_model import OrderedModel
from google.colab import files
import random
from scipy.stats import shapiro
import io

# Set the style for seaborn
sns.set_style('whitegrid')
sns.color_palette()

dfBMI = pd.read_csv('https://raw.githubusercontent.com/P-Saha/k-NN_and_Clustering_Analysis/main/BMI_Dataset.csv')
dfCancer = pd.read_csv('https://raw.githubusercontent.com/P-Saha/k-NN_and_Clustering_Analysis/main/Lung_Cancer_Dataset.csv')

#dfBMI['Gender'] = dfBMI['Gender'].map({'Male': 1, 'Female': 0})
#dfCancer['Gender'] = dfCancer['Gender'].map({1: 1, 2: 0})

#dfBMI = dfBMI.drop(['Height','Weight'], axis=1)

print(dfBMI.dtypes, "\n")
print(dfCancer.dtypes)
```

```
Age                     int64
Gender                 object
Calorie_Intake          int64
Sugar_Intake            int64
Physical_Activity       int64
Fast_Food_Frequency     int64
Screen_Time             int64
Height                float64
Weight                float64
BMI                   float64
dtype: object

Age                  int64
Gender               int64
Air Pollution        int64
Alcohol Usage        int64
Genetic Risk         int64
Lung Disease         int64
Obesity              int64
Smoking              int64
Passive Smoker       int64
Chest Pain           int64
Coughing of Blood    int64
Severity             int64
dtype: object
```

```
dfCancer.head()
```

	Age	Gender	Air Pollution	Alcohol Usage	Genetic Risk	Lung Disease	Obesity	Smoking	Passive Smoker	Chest Pain	Coughing of Blood
0	33	1	2	4	3	2	4	3	2	2	4
1	17	1	3	1	4	2	2	2	4	2	3
2	35	1	4	5	5	4	7	2	3	4	8

Exploratory Data Analysis and Visualization

In this section we create various plots to visualise the data in order to have a general initial understanding of the datasets and identify clear or obvious patterns and trends in the dataset.

Obesity Dataset

```
# BMI Variables
var_obsese = list(dfBMI.columns)
print(var_obsese)
```

```
['Age', 'Gender', 'Calorie_Intake', 'Sugar_Intake', 'Physical_Activity', 'Fast_Food_Frequency', 'Screen_Time', 'Height', 'Weight', 'BMI
```

```
# Examining Missing Data
print("Missing Value Distribution:")
print(dfBMI.isnull().mean())
print("")
```

```
Missing Value Distribution:
Age                    0.0
Gender                 0.0
Calorie_Intake         0.0
Sugar_Intake           0.0
Physical_Activity      0.0
Fast_Food_Frequency    0.0
Screen_Time            0.0
Height                 0.0
Weight                 0.0
BMI                    0.0
dtype: float64
```

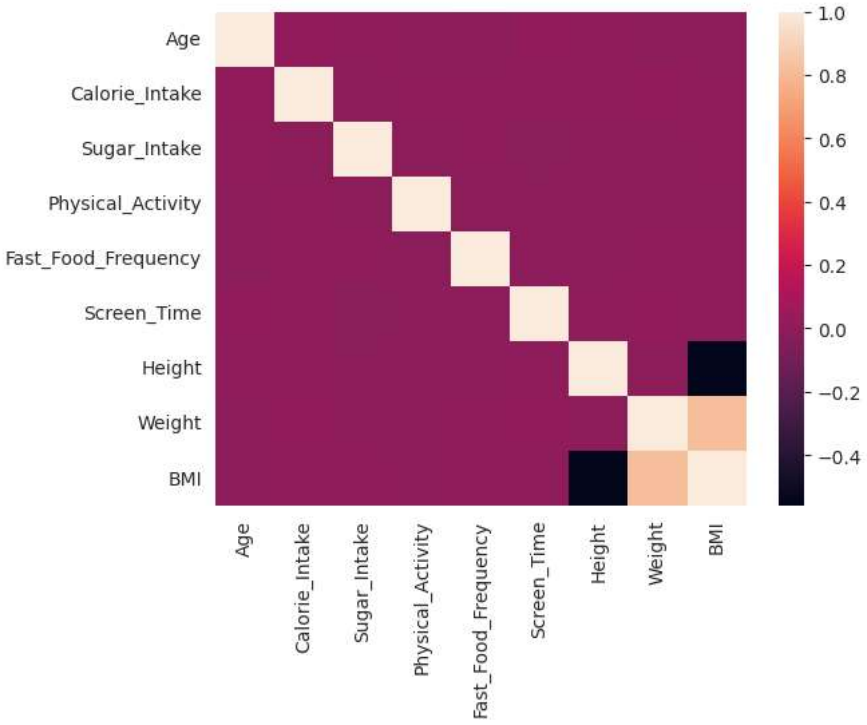
```
# Descriptive Statistics
print(dfBMI.describe())
```

```
                Age  Calorie_Intake  Sugar_Intake  Physical_Activity  \
count  23535.000000    23535.000000  23535.000000       23535.000000
mean      48.532993     2495.845634     64.824559          89.496707
std       17.872736      577.798752     32.004657          52.120810
min       18.000000     1500.000000     10.000000           0.000000
25%       33.000000     1991.000000     37.000000          45.000000
50%       48.000000     2495.000000     65.000000          90.000000
75%       64.000000     2994.000000     93.000000         135.000000
max       79.000000     3499.000000    119.000000         179.000000

       Fast_Food_Frequency   Screen_Time        Height        Weight  \
count         23535.000000  23535.000000  23535.000000  23535.000000
mean              2.018441      5.486127      1.650279     70.019142
std               1.414634      3.454036      0.119714     15.016866
min               0.000000      0.000000      1.196086     10.010016
25%               1.000000      2.000000      1.569815     59.859242
50%               2.000000      6.000000      1.650122     70.022501
75%               3.000000      8.000000      1.731470     80.038009
max               4.000000     11.000000      2.154243    126.493504

                BMI
count  23535.000000
mean      26.127641
std        6.875510
min        3.442184
25%       21.337576
50%       25.661507
75%       30.329032
max       70.475419
```

```
# Correlation Heatmap
```

```
sns.heatmap(dfBMI.corr(), annot=False)
plt.show()
```

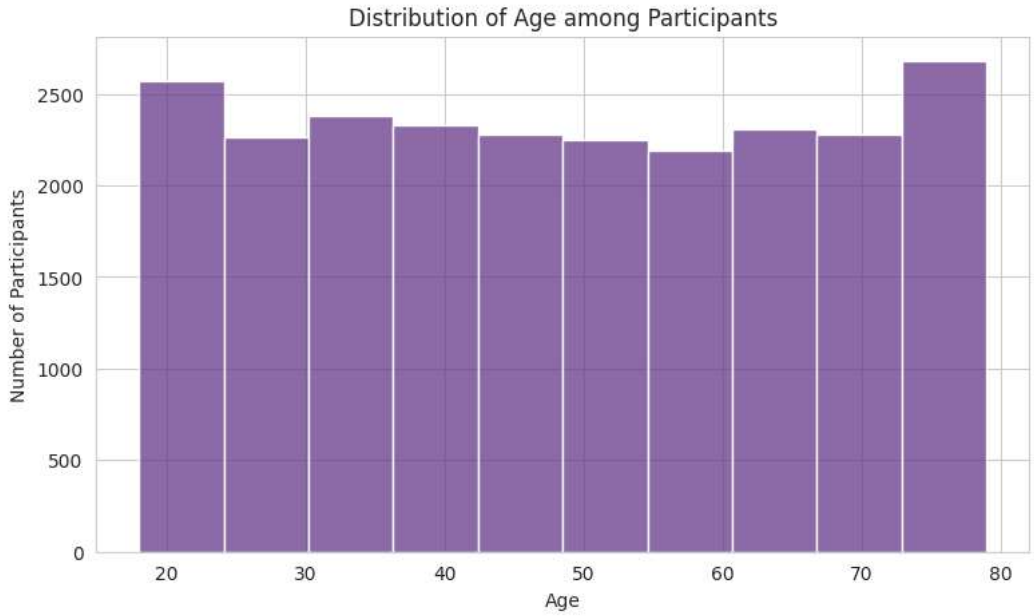
```
<ipython-input-6-9b41611829cd>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version,
  sns.heatmap(dfBMI.corr(), annot=False)
```



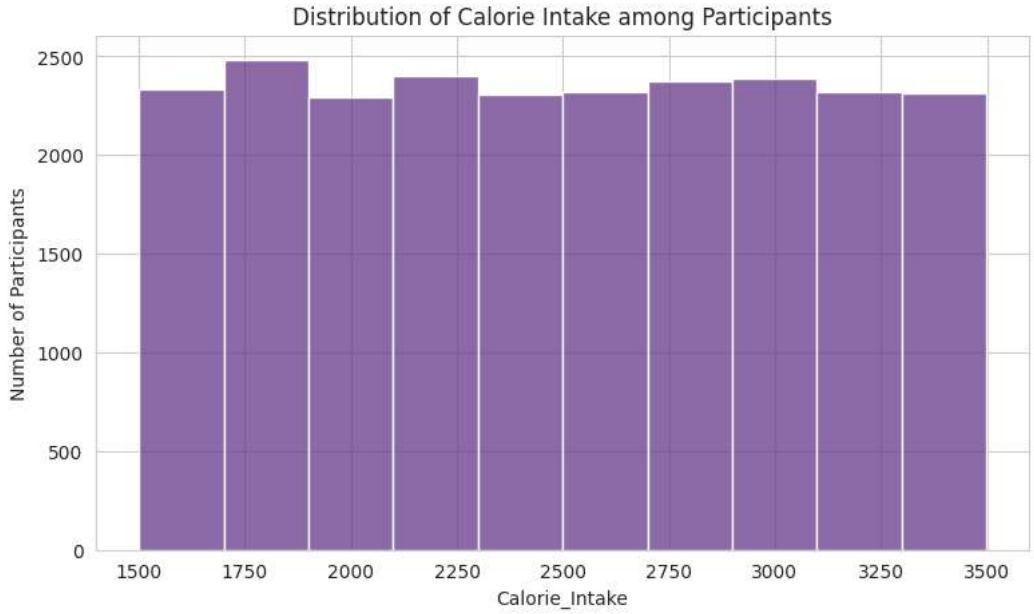

```
# Testing normality in the Canadian obesity data

# Set the style for seaborn
sns.set_style('whitegrid')

# Create a histogram for the 'Age' feature
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Age'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Age among Participants')
plt.xlabel('Age')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.Age)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")

#########################################################################
#Similar method was applied to the remaining continuous variables:

##Caloric Intake
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Calorie_Intake'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Calorie Intake among Participants')
plt.xlabel('Calorie_Intake')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.Calorie_Intake)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")
```

```
#########################################################################
##Sugar Intake
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Sugar_Intake'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Sugar Intake among Participants')
plt.xlabel('Sugar_Intake')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.Sugar_Intake)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")

#########################################################################
##Physical Activity
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Physical_Activity'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Physical Activities among Participants')
plt.xlabel('Physical_Activities')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.Physical_Activity)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")


#########################################################################
##Fast Food Frequency
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Fast_Food_Frequency'], bins=5, kde=False, color='#66378a')
plt.title('Distribution of Fast Food Frequency among Participants')
plt.xlabel('Fast_Food_Frequency')
plt.ylabel('Number of Participants')
plt.show() #ordinal

#########################################################################
##Screen Time
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Screen_Time'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Screen Time among Participants')
plt.xlabel('Screen_Time')
plt.ylabel('Number of Participants')
plt.show() #ordinal

#########################################################################
##Height
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['Height'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Height among Participants')
plt.xlabel('Height')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.Height)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")

#########################################################################
##Weight
plt.figure(figsize=(9, 5))
```

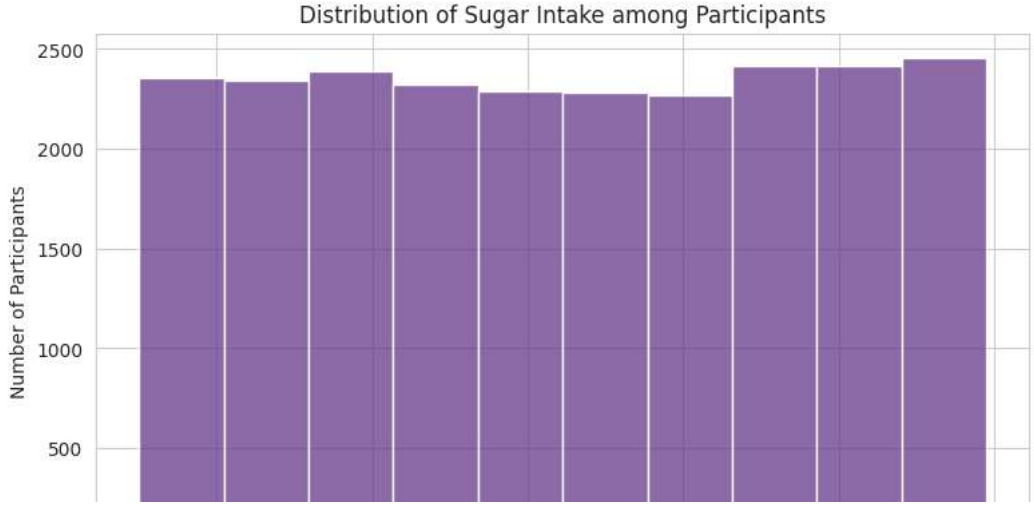
```
sns.histplot(dfBMI['Weight'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Weight among Participants')
plt.xlabel('Weight')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.Weight)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")

#########################################################################
##BMI
plt.figure(figsize=(9, 5))
sns.histplot(dfBMI['BMI'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of BMI among Participants')
plt.xlabel('BMI')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfBMI.BMI)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")
```

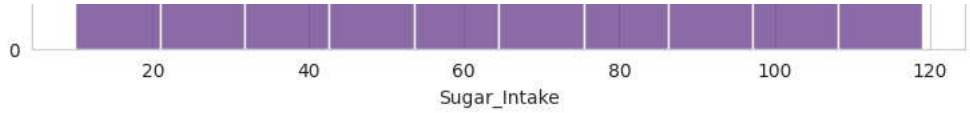



```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
Statistics=0.954, p=0.000
Sample does not look Gaussian, thus reject H0
```

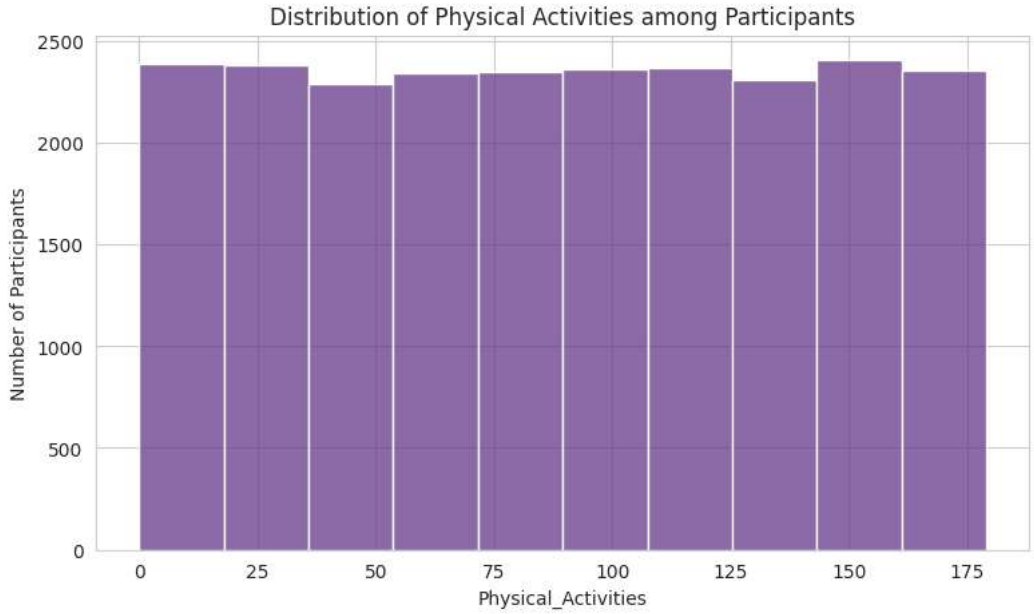



```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
Statistics=0.955, p=0.000
Sample does not look Gaussian, thus reject H0
```

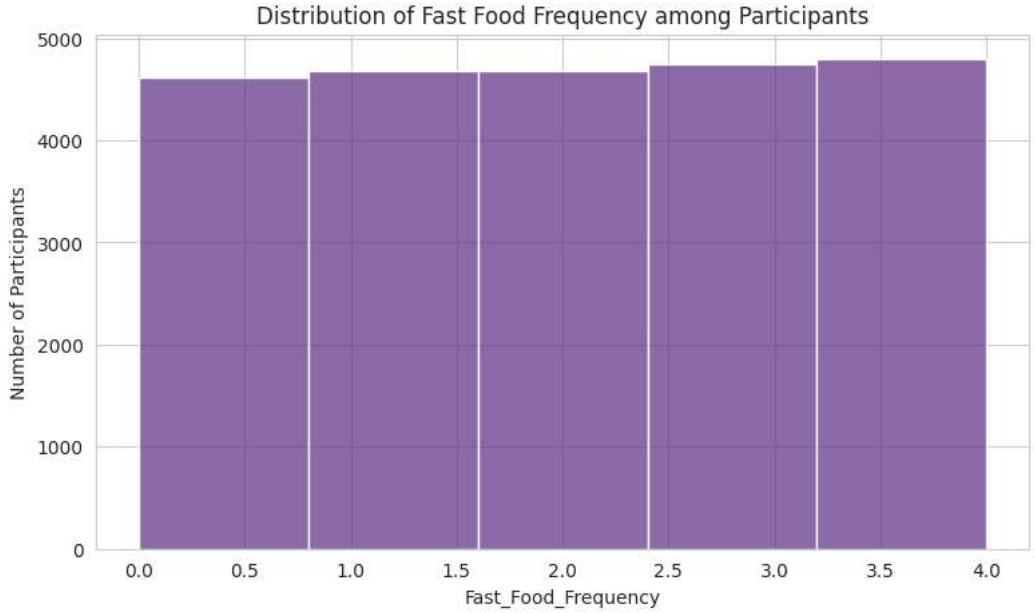


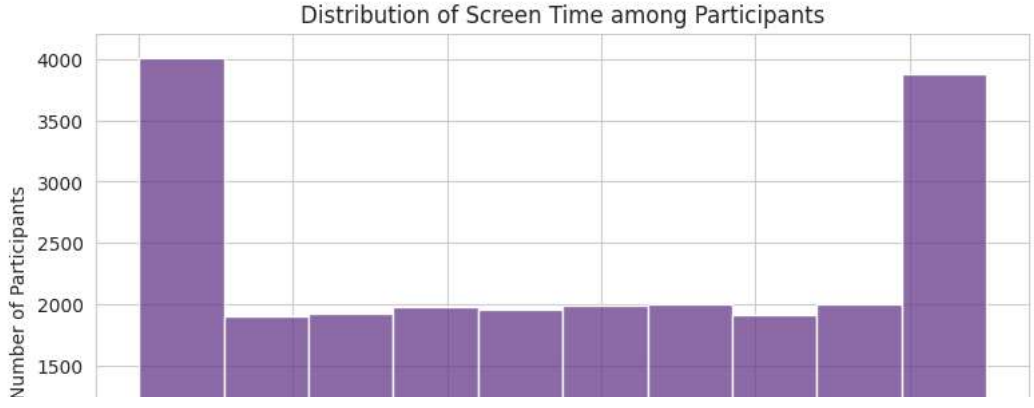


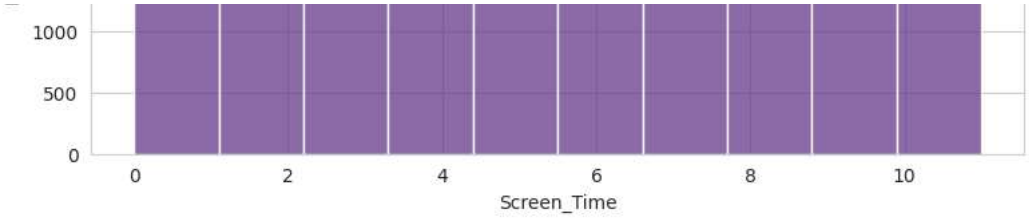

```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
Statistics=0.952, p=0.000
Sample does not look Gaussian, thus reject H0
```

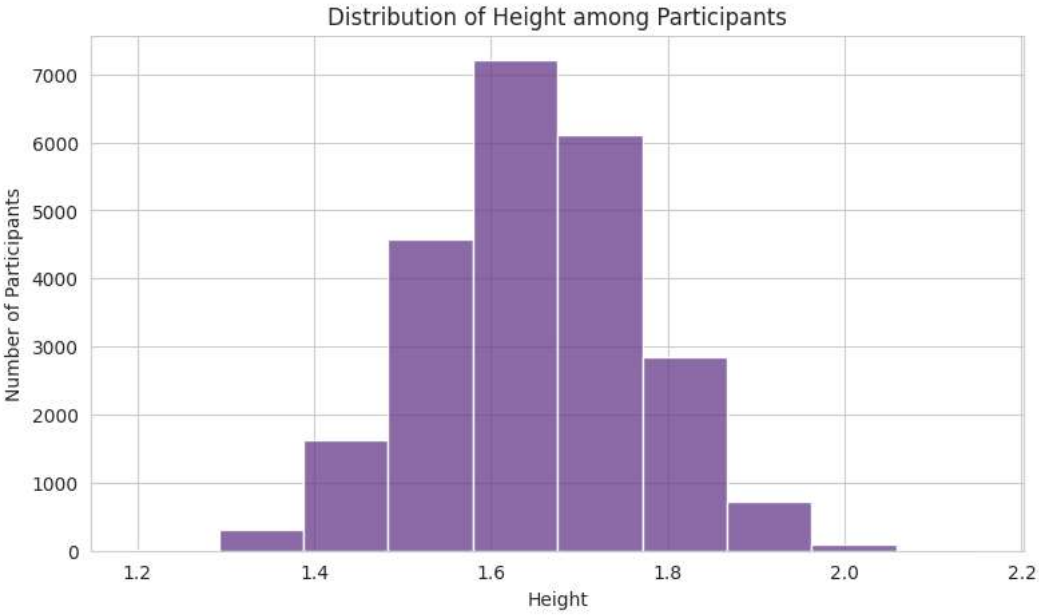



```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
Statistics=0.954, p=0.000
Sample does not look Gaussian, thus reject H0
```

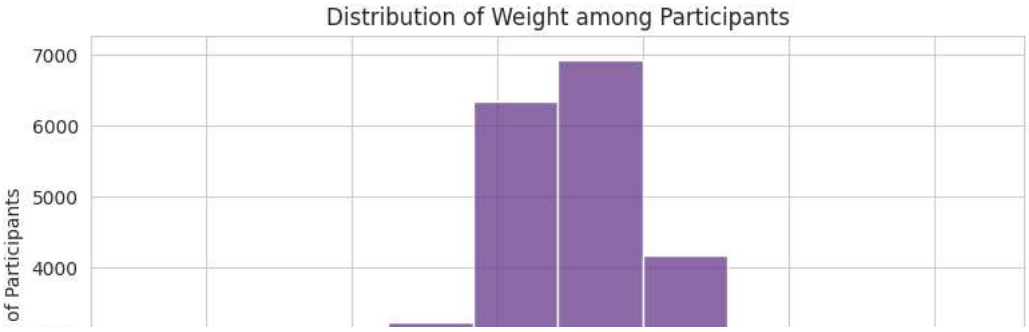









```
/usr/local/lib/python3.10/dist-packages/scipy/stats/_morestats.py:1882: UserWarning: p-value may not be accurate for N > 5000.
  warnings.warn("p-value may not be accurate for N > 5000.")
Statistics=1.000, p=0.869
Sample looks Gaussian, thus failed to reject H0
```



Lung Cancer Dataset

```
# Cancer Variables
var_cancer = list(dfCancer.columns)
print(var_cancer)
```

```
['Age', 'Gender', 'Air Pollution', 'Alcohol Usage', 'Genetic Risk', 'Lung Disease', 'Obesity', 'Smoking', 'Passive Smoker', 'Chest Pain
```

```
# Examining Missing Data
print("Missing Value Distribution:")
print(dfCancer.isnull().mean())
print("")
```

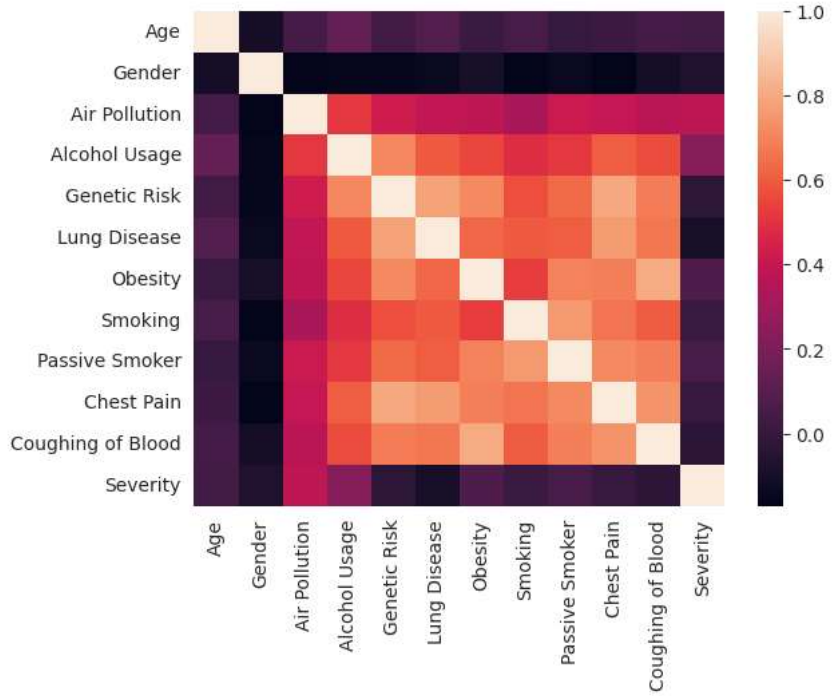
```
Missing Value Distribution:
Age                 0.0
Gender              0.0
Air Pollution       0.0
Alcohol Usage       0.0
Genetic Risk        0.0
Lung Disease        0.0
Obesity             0.0
Smoking             0.0
Passive Smoker      0.0
```

```
Chest Pain           0.0
Coughing of Blood    0.0
Severity             0.0
dtype: float64
```

```
# Descriptive Statistics
print(dfCancer.describe())
```

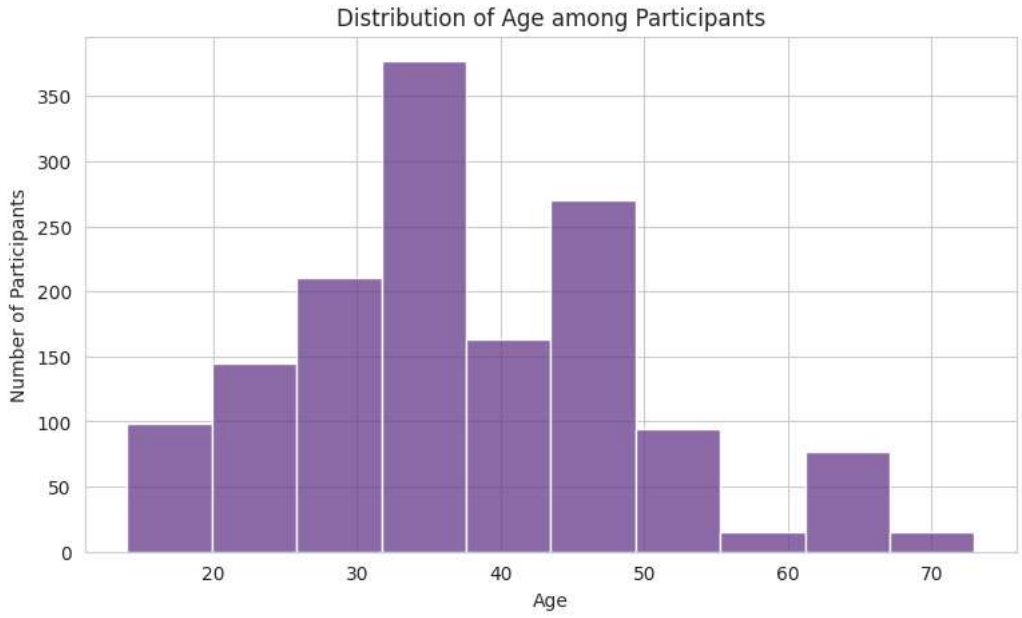
```
                Age       Gender  Air Pollution  Alcohol Usage  Genetic Risk  \
count  1465.000000  1465.000000    1465.000000    1465.000000   1465.000000
mean     37.238225     1.404778       3.958362       4.258020      3.892150
std      12.078575     0.491017       2.033272       2.564265      2.134635
min      14.000000     1.000000       1.000000       1.000000      1.000000
25%      28.000000     1.000000       2.000000       2.000000      2.000000
50%      36.000000     1.000000       4.000000       4.000000      3.000000
75%      45.000000     2.000000       6.000000       7.000000      6.000000
max      73.000000     2.000000       8.000000       8.000000      7.000000

       Lung Disease      Obesity      Smoking  Passive Smoker   Chest Pain  \
count   1465.000000  1465.000000  1465.000000     1465.000000  1465.000000
mean       3.741297     3.844369     3.425256        3.606826     3.817065
std        1.899984     2.072213     2.273040        2.148123     2.143897
min        1.000000     1.000000     1.000000        1.000000     1.000000
25%        2.000000     2.000000     2.000000        2.000000     2.000000
50%        3.000000     3.000000     3.000000        3.000000     3.000000
75%        6.000000     6.000000     5.000000        4.000000     6.000000
max        7.000000     7.000000     8.000000        8.000000     9.000000

       Coughing of Blood     Severity
count        1465.000000  1465.000000
mean            3.990444     1.677133
std             2.420860     1.126034
min             1.000000     0.000000
25%             2.000000     1.000000
50%             3.000000     2.000000
75%             7.000000     3.000000
max             9.000000     3.000000
```

```
# Correlation Heatmap
sns.heatmap(dfCancer.corr(), annot=False)
plt.show()
```

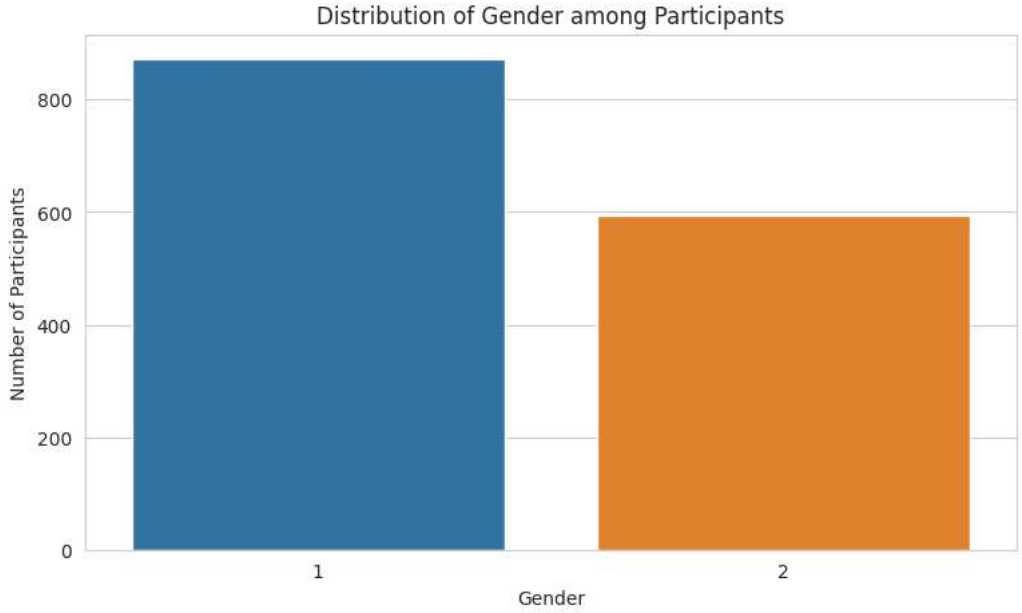


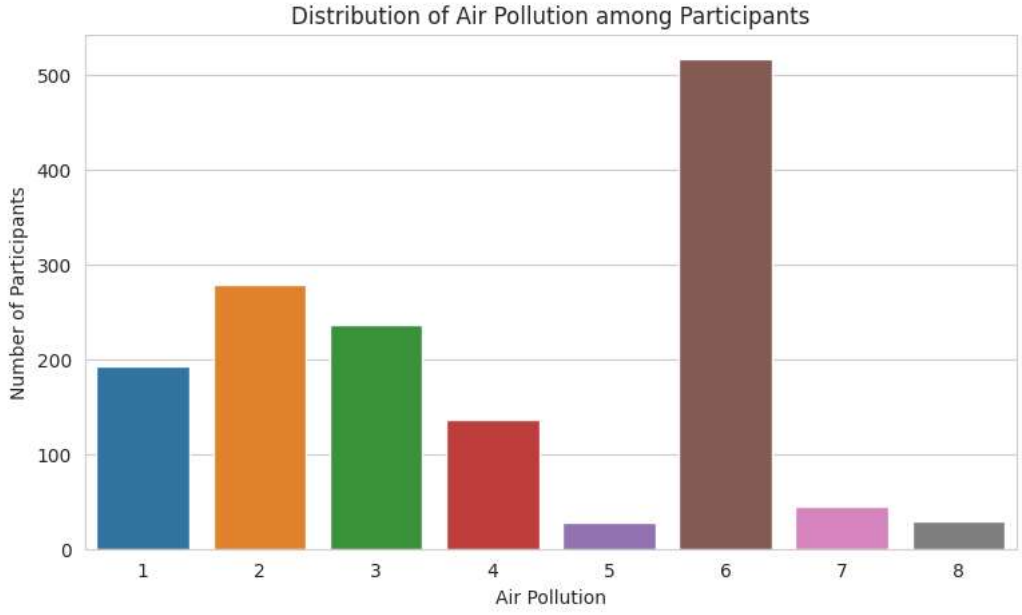

```
XCancer = dfCancer[['Air Pollution','Alcohol Usage','Genetic Risk','Lung Disease','Obesity','Smoking','Passive Smoker','Chest Pain','Coughing
yCancer = dfCancer['Severity']

# Plotting histogram for all variables

##Age
```

```
plt.figure(figsize=(9, 5))
sns.histplot(dfCancer['Age'], bins=10, kde=False, color='#66378a')
plt.title('Distribution of Age among Participants')
plt.xlabel('Age')
plt.ylabel('Number of Participants')
plt.show() #continuous

#Shapiro-Wilk test for normality
stat,p=shapiro(dfCancer.Age)
print('Statistics=%.3f, p=%.3f' % (stat, p))
alpha = 0.05
if p>alpha:
    print("Sample looks Gaussian, thus failed to reject H0")
else:
    print("Sample does not look Gaussian, thus reject H0")

######################################################################
##Gender

plt.figure(figsize=(9, 5))
sns.countplot(data=dfCancer, x='Gender')
plt.title('Distribution of Gender among Participants')
plt.xlabel('Gender')
plt.ylabel('Number of Participants')
plt.show() #categorical

# Ordinal Variables
for col in XCancer.columns:
  plt.figure(figsize=(9, 5))
  sns.countplot(data=dfCancer, x=col)
  plt.title('Distribution of ' + col + ' among Participants')
  plt.xlabel(col)
  plt.ylabel('Number of Participants')
  plt.show() #ordinal

plt.figure(figsize=(9, 5))
sns.countplot(data=dfCancer, x='Severity')
plt.title('Distribution of Severity among Participants')
plt.xlabel('Severity')
plt.ylabel('Number of Participants')
plt.show() #ordinal
```

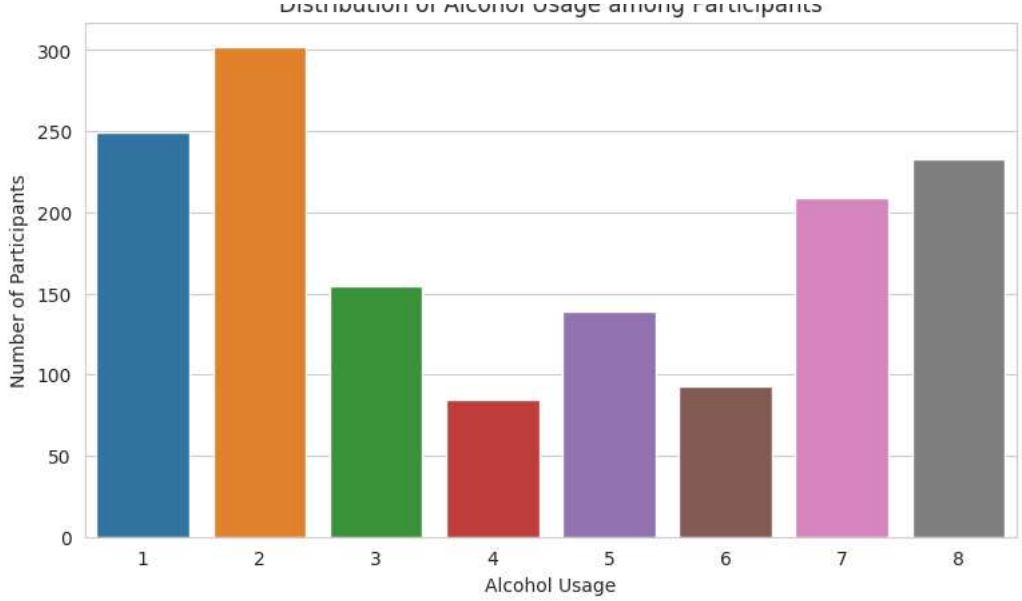


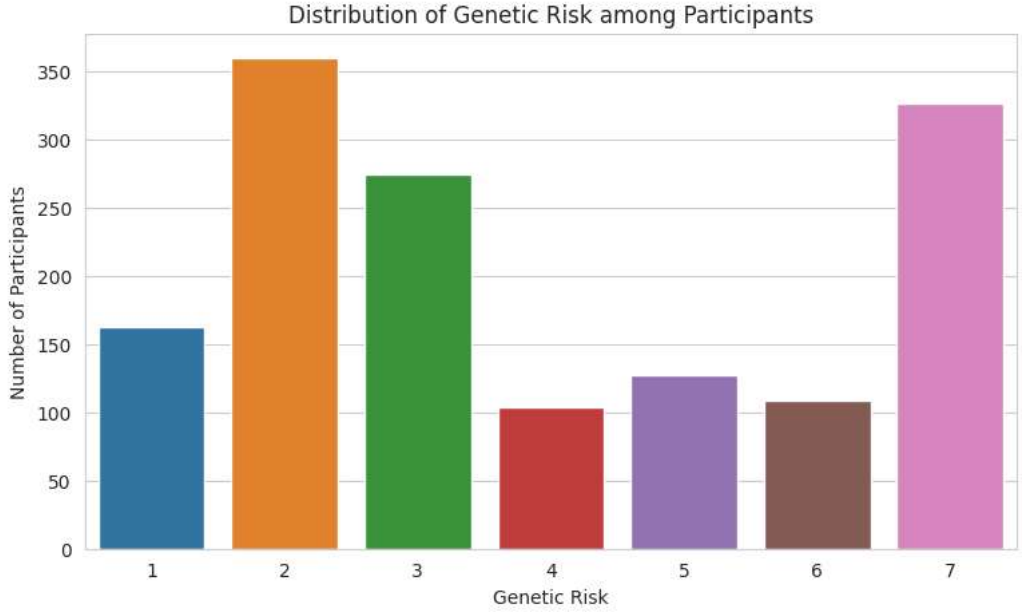

```
Statistics=0.970, p=0.000
Sample does not look Gaussian, thus reject H0
```

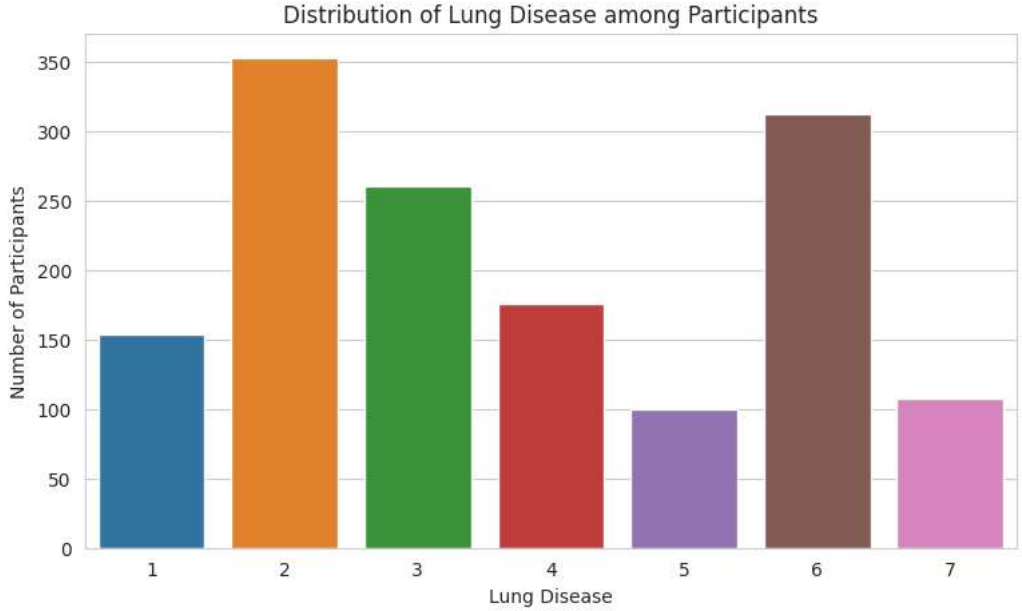


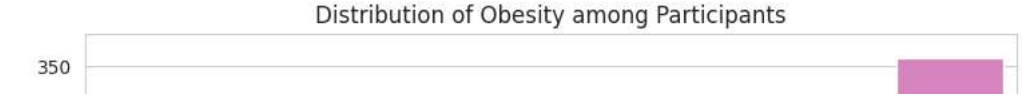


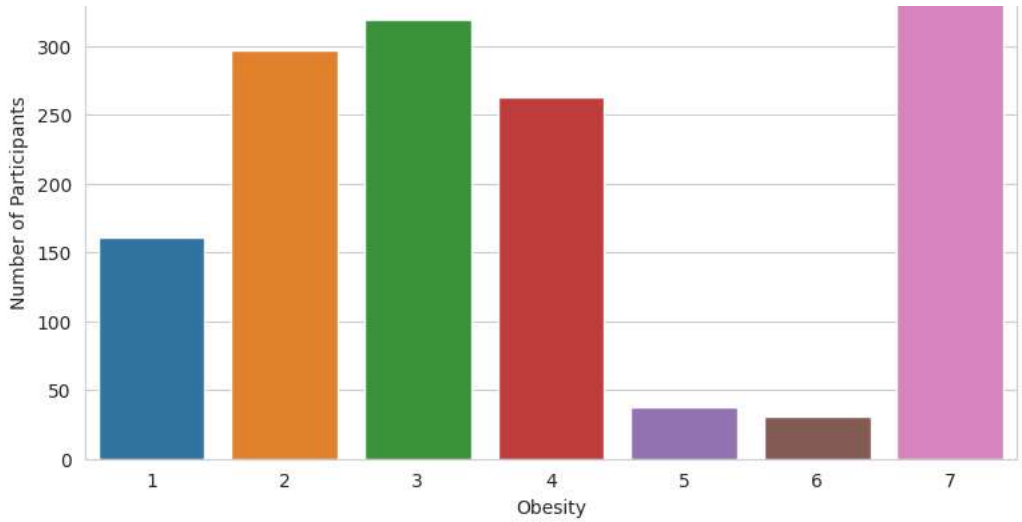
Distribution of Alcohol Usage among Participants

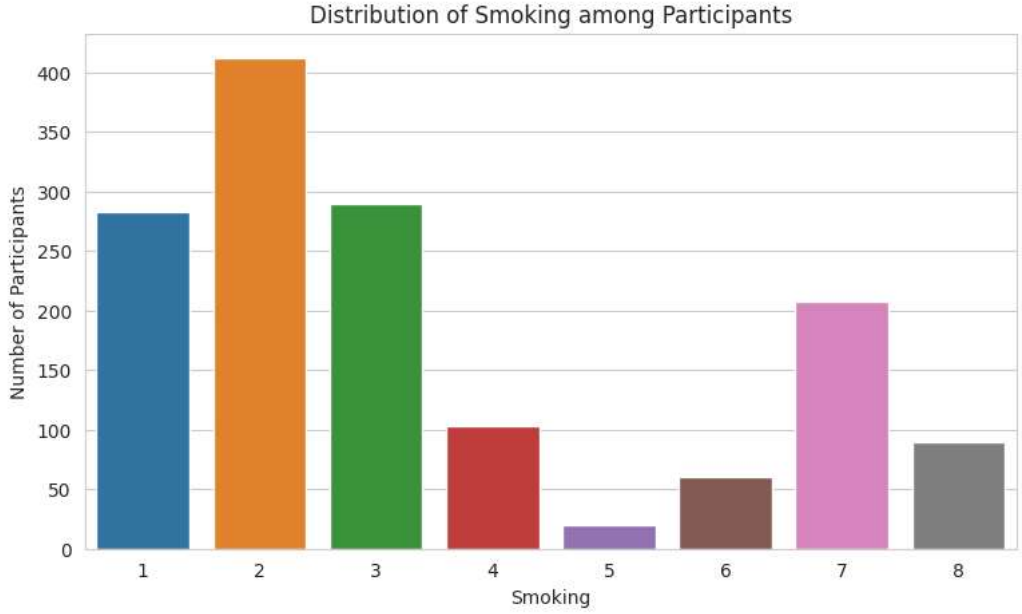

Number of Participants
0
50
100
150
200
250
300
1
2
3
4
5
6
7
8
Alcohol Usage

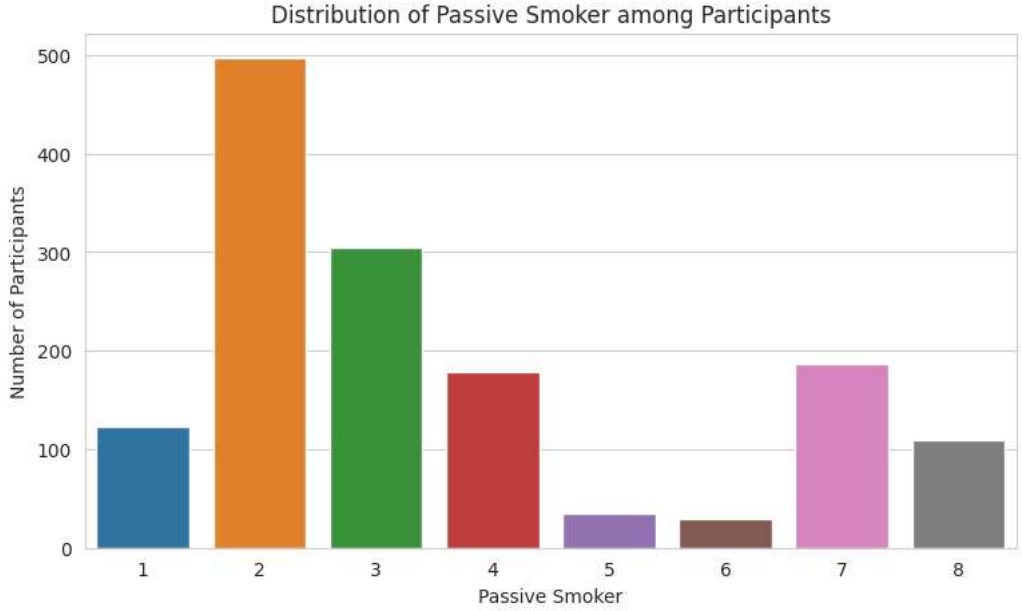

Distribution of Genetic Risk among Participants
Number of Participants
0
50
100
150
200
250
300
350
1
2
3
4
5
6
7
Genetic Risk

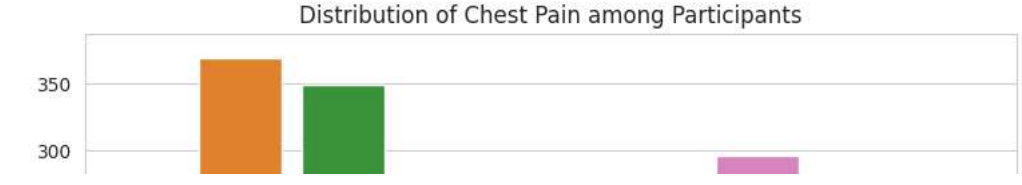

Distribution of Lung Disease among Participants
Number of Participants
0
50
100
150
200
250
300
350
1
2
3
4
5
6
7
Lung Disease

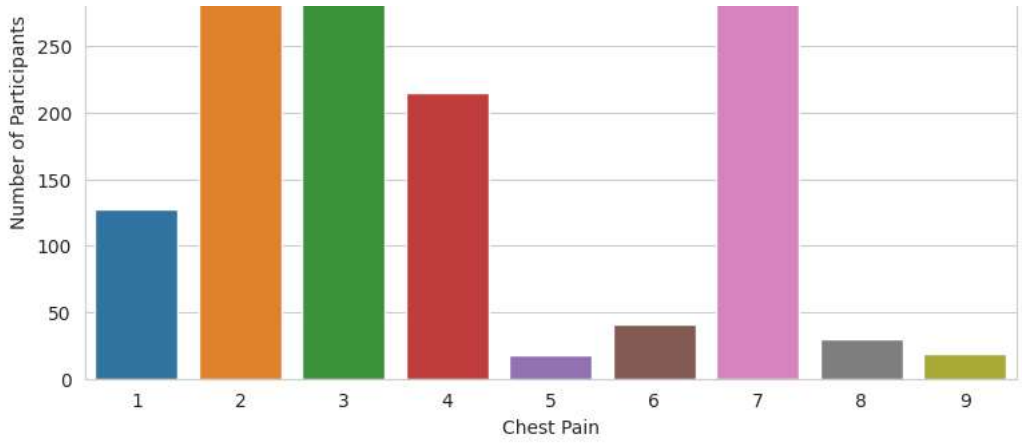

Distribution of Obesity among Participants
350

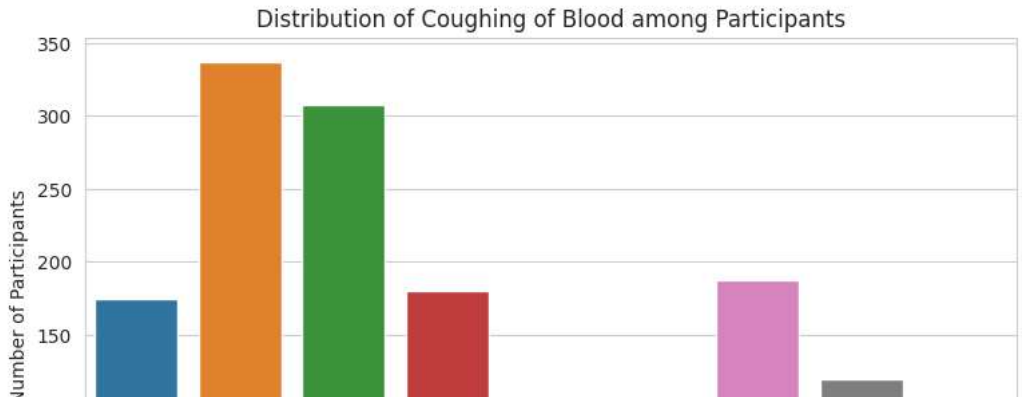

Number of Participants
0
50
100
150
200
250
300
1
2
3
4
5
6
7
Obesity


Distribution of Smoking among Participants
Number of Participants
0
50
100
150
200
250
300
350
400
1
2
3
4
5
6
7
8
Smoking


Distribution of Passive Smoker among Participants
Number of Participants
0
100
200
300
400
500
1
2
3
4
5
6
7
8
Passive Smoker


Distribution of Chest Pain among Participants
300
350





Pre-processing

In the obsesity dataset, the following variables were continuous: Age, Calorie_Intake, Sugar_Intake, Physical_Activity,Fast_Food_Frequency, Screen_Time, Height, Weight, BMI. These variables exhibited notable differences in magnitude. Therefore, to avoid potential biases towards variables with higher magnitudes, all continous variables were normalized before applying the KNN algorithm.

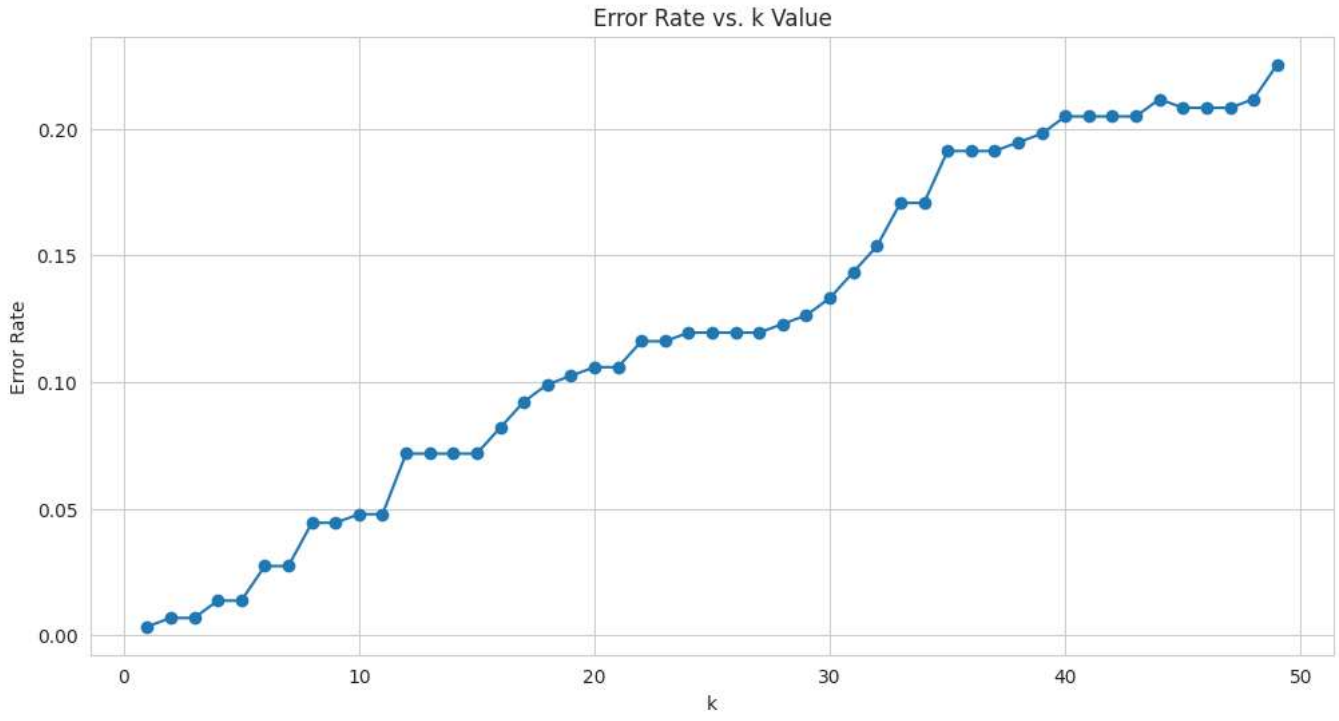
In the cancer dataset, most variables were categorical/ordinal with similar number of ordinal levels. The age and gender data is removed due to low correlation.

```
# Dropping Age and Gender
dfCancer.drop('Age', axis=1, inplace=True)
dfCancer.drop('Gender', axis=1, inplace=True)

# Setting X and y
X = dfCancer[['Air Pollution','Alcohol Usage','Genetic Risk','Lung Disease','Obesity','Smoking','Passive Smoker','Chest Pain','Coughing of Bl
y = dfCancer['Severity']

# Scaling
minMaxScaler = preprocessing.MinMaxScaler()
X = pd.DataFrame(minMaxScaler.fit_transform(X.values))
```

k-NN Prediction Model for Lung Cancer Dataset

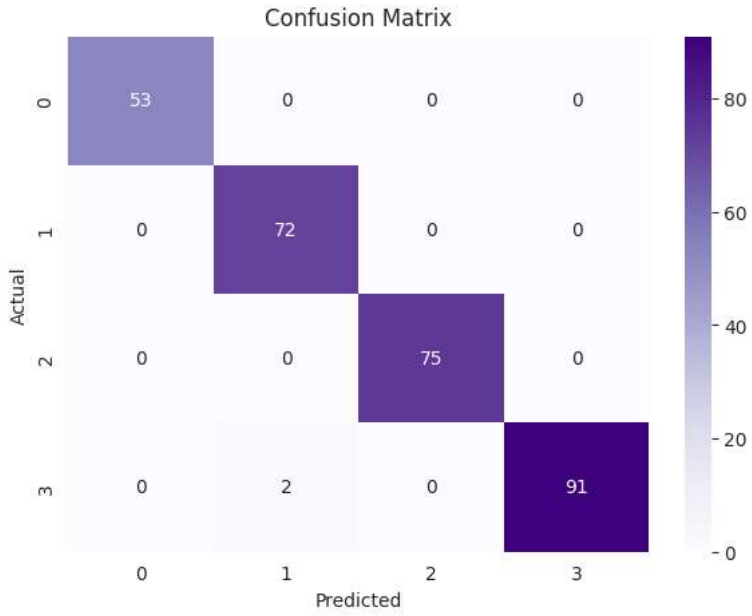
```
# Defining Functions:

# Plots a confusion matrix
def confusionPlot(classes, y_test, y_pred):
  # Plotting the confusion matrix
  plt.figure(figsize=(7,5))
  sns.heatmap(confusion_matrix(y_test, y_pred, labels=classes), cmap='Purples', annot=True, fmt='g', xticklabels=classes, yticklabels=classes
  plt.xlabel('Predicted')
  plt.ylabel('Actual')
  plt.title('Confusion Matrix')
  return plt

# Elbow Method from "KNN_elbow_point_JZ"
# Loop over k_values to train and test the KNN classifier, plots the errors
```

```
def elbowPointPlot(X, y, k_values, t_size=0.2, r_state=0):
  # Split dataset into training set and test set
  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=t_size, random_state=r_state)

  # Choose a range of k values to test.

  # A list to keep track of the error rates for each k value.
  errors = []

  for k in k_values:
    # Initialize a KNN classifier with current k value.
    knn = KNeighborsClassifier(n_neighbors=k)

    # Train the classifier on the training data.
    knn.fit(X_train, y_train)

    # Predict the labels for the test set.
    y_pred = knn.predict(X_test)

    # Calculate the error rate: 1 - accuracy.
    error = 1 - accuracy_score(y_test, y_pred)
    errors.append(error)

  # Plotting
  # This visualization helps in understanding the relationship between k value and the error rate.
  plt.figure(figsize=(12, 6))
  plt.plot(range(1,50), errors, marker='o')
  plt.title('Error Rate vs. k Value')
  plt.xlabel('k')
  plt.ylabel('Error Rate')
  plt.grid(True)
  plt.show()

  return np.array(errors)


errors = elbowPointPlot(X, y, range(1,50))
# Choose k=2
```

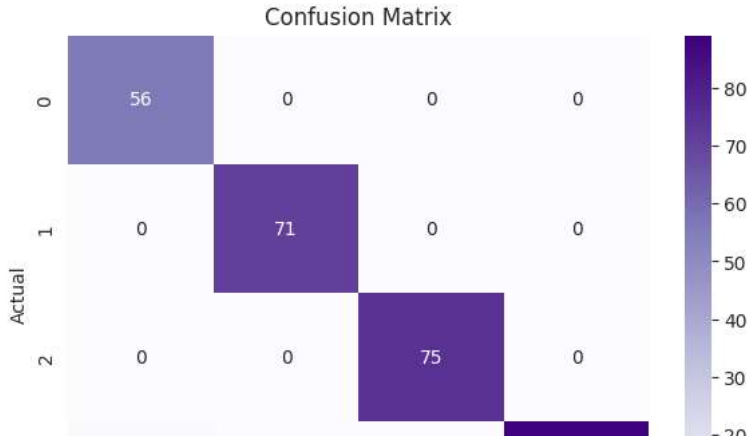



```
# Get unique classes
#random.seed(1234)
classes = dfCancer['Severity'].unique()

# Fit knn with k=2:
knn=KNeighborsClassifier(n_neighbors=2)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
knn.fit(X_train, y_train)
```

```
y_pred = knn.predict(X_test)

# Plotting the confusion matrix
confusionPlot(classes, y_test, y_pred).show()

for i in range(3):
    # Fit knn with k=2 for various random states:
    knn=KNeighborsClassifier(n_neighbors=2)
    randN = random.randint(0, 100)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=randN)
    knn.fit(X_train, y_train)
    y_pred = knn.predict(X_test)

    # Plotting the confusion matrix
    print("Random State = ",randN,":", sep="")
    confusionPlot(classes, y_test, y_pred).show()
```



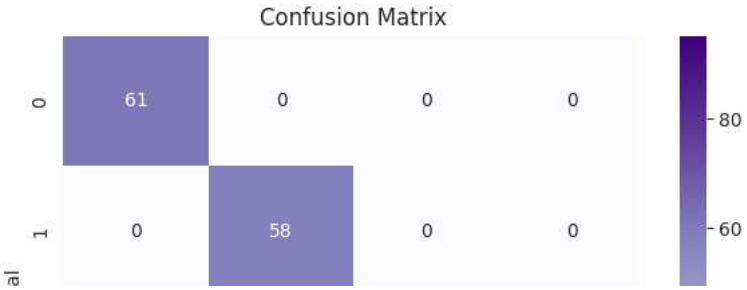
Random State = 100:




```
# Fit knn with k=2:
knn=KNeighborsClassifier(n_neighbors=2)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=100)
knn.fit(X_train, y_train)
y_pred = knn.predict(X_test)

#random.seed(1234)
print(confusion_matrix(y_test, y_pred))
print(classification_report(y_test, y_pred))
#print(classification_report(y_test, y_pred))
```

```
[[56  0  0  0]
 [ 0 71  0  0]
 [ 0  0 75  0]
 [ 2  0  0 89]]
              precision    recall  f1-score   support

           0       0.97      1.00      0.98        56
           1       1.00      1.00      1.00        71
           2       1.00      1.00      1.00        75
           3       1.00      0.98      0.99        91

    accuracy                           0.99       293
   macro avg       0.99      0.99      0.99       293
weighted avg       0.99      0.99      0.99       293
```

Random State = 36:


Confusion Matrix
61
0
0
0
0
58
0
0
80
60
al

Datathon 1: Analysis of Lung Cancer and BMI Risk Factors

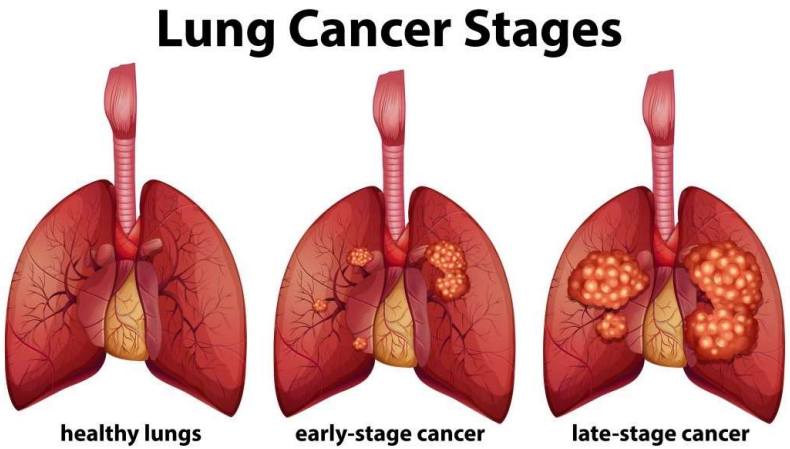
Priyonto Saha, Kinna Zhao, Yacine Marouf

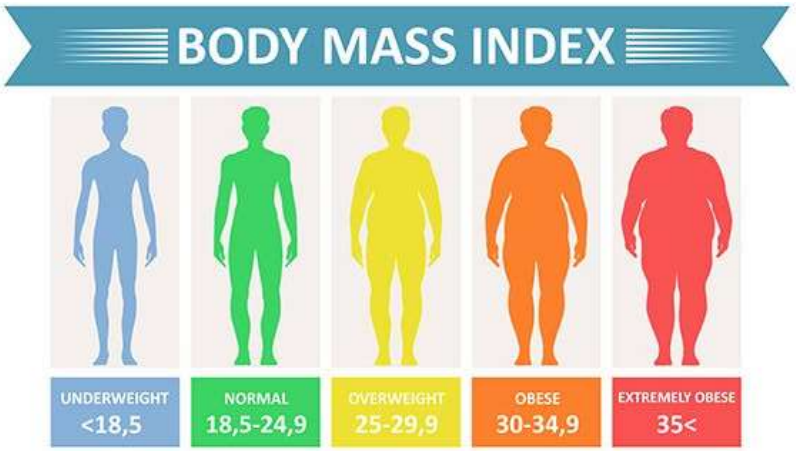
Objectives

1) Analyse and visualise the descriptive statistics of a lung cancer dataset from Ethiopia and a BMI dataset from Canada.

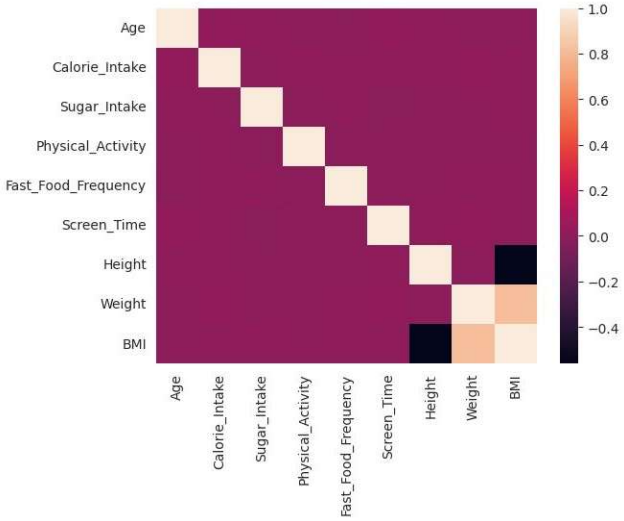
2) Predict the possible severity of cancer and rate of obesity through the use of linear regressions and K-nearest-neighbours on multiple risk factors.

3) If possible, determine any possible environmental, behavioural, and physical similarities between varying lung cancer patients and people with varying BMIs through the use of K means unsupervised learning.

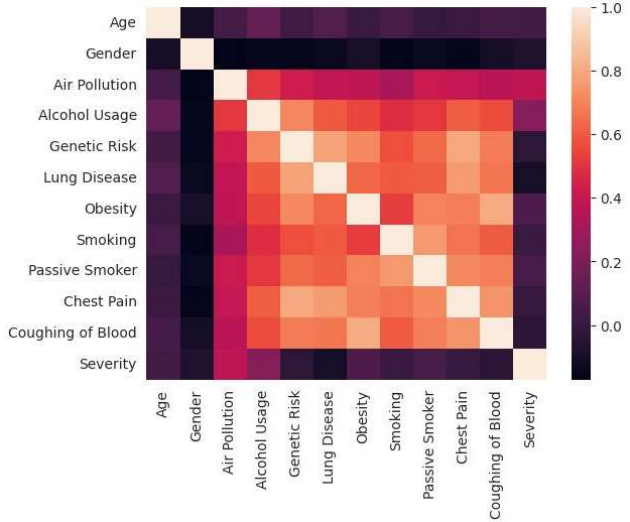




Visualisation and preprocessing



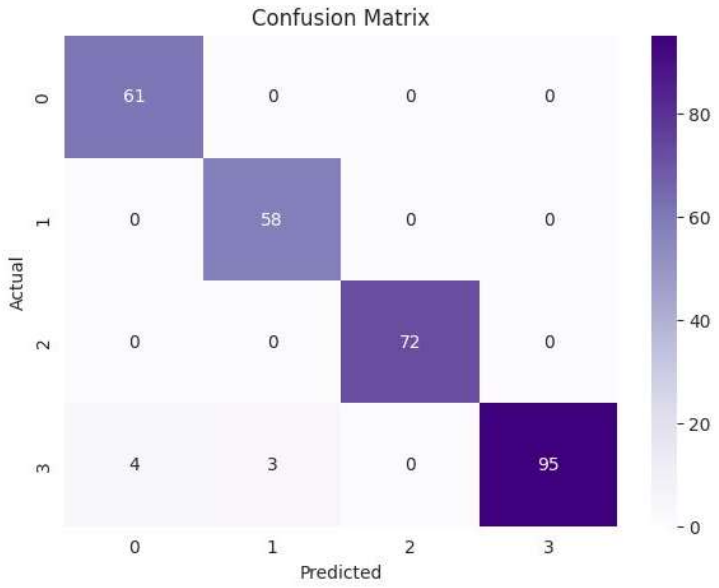
BMI dataset



Lung cancer dataset

Results and analysis

0 is Low Severity, 1 is Medium Severity,
2 is High Severity, 3 is Healthy.



KNN on Cancer dataset with K = 2.

class	precision	recall	f1-score	support
0	0.97	1.00	0.98	56
1	1.00	1.00	1.00	71
2	1.00	1.00	1.00	75
3	1.00	0.98	0.99	91
accuracy			0.99	293
Macro avg	0.99	0.99	0.99	293
Micro avg	0.99	0.99	0.99	293