lungcancer

June 28, 2024

[ ]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score
from matplotlib import pyplot as plt
```

[ ]:

```
data_true=pd.read_csv("/content/drive/MyDrive/lung cancer/cancer patient data␣
  ↪sets.csv")
df = data_true
```

[ ]:

```
df.head()
```

[ ]:

```
   index Patient Id  Age  Gender  Air Pollution  Alcohol use  Dust Allergy  \
0      0         P1   33       1              2            4             5
1      1        P10   17       1              3            1             5
2      2       P100   35       1              4            5             6
3      3      P1000   37       1              7            7             7
4      4       P101   46       1              6            8             7

   OccuPational Hazards  Genetic Risk  chronic Lung Disease  ...  Fatigue  \
0                     4             3                     2  ...        3
1                     3             4                     2  ...        1
2                     5             5                     4  ...        8
3                     7             6                     7  ...        4
4                     7             7                     6  ...        3

   Weight Loss  Shortness of Breath  Wheezing  Swallowing Difficulty  \
0            4                    2         2                      3
1            3                    7         8                      6
2            7                    9         2                      1
3            2                    3         1                      4
4            2                    4         1                      4

   Clubbing of Finger Nails  Frequent Cold  Dry Cough  Snoring   Level
0                         1              2          3        4     Low
1                         2              1          7        2  Medium
```

```
2                        4             6          7        2    High
3                        5             6          7        5    High
4                        2             4          2        3    High

[5 rows x 26 columns]
```

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

```
[ ]:             index          Age       Gender  Air Pollution  Alcohol use  \
     count  1000.000000  1000.000000  1000.000000      1000.0000  1000.000000
     mean    499.500000    37.174000     1.402000         3.8400     4.563000
     std     288.819436    12.005493     0.490547         2.0304     2.620477
     min       0.000000    14.000000     1.000000         1.0000     1.000000
     25%     249.750000    27.750000     1.000000         2.0000     2.000000
     50%     499.500000    36.000000     1.000000         3.0000     5.000000
     75%     749.250000    45.000000     2.000000         6.0000     7.000000
     max     999.000000    73.000000     2.000000         8.0000     8.000000

            Dust Allergy  OccuPational Hazards  Genetic Risk  chronic Lung Disease  \
     count   1000.000000           1000.000000   1000.000000           1000.000000
     mean       5.165000              4.840000      4.580000              4.380000
     std        1.980833              2.107805      2.126999              1.848518
     min        1.000000              1.000000      1.000000              1.000000
     25%        4.000000              3.000000      2.000000              3.000000
     50%        6.000000              5.000000      5.000000              4.000000
     75%        7.000000              7.000000      7.000000              6.000000
     max        8.000000              8.000000      7.000000              7.000000

            Balanced Diet  …  Coughing of Blood      Fatigue  Weight Loss  \
     count    1000.000000  …        1000.000000  1000.000000  1000.000000
     mean        4.491000  …           4.859000     3.856000     3.855000
     std         2.135528  …           2.427965     2.244616     2.206546
     min         1.000000  …           1.000000     1.000000     1.000000
     25%         2.000000  …           3.000000     2.000000     2.000000
     50%         4.000000  …           4.000000     3.000000     3.000000
     75%         7.000000  …           7.000000     5.000000     6.000000
     max         7.000000  …           9.000000     9.000000     8.000000

            Shortness of Breath     Wheezing  Swallowing Difficulty  \
     count          1000.000000  1000.000000            1000.000000
     mean              4.240000     3.777000               3.746000
     std               2.285087     2.041921               2.270383
     min               1.000000     1.000000               1.000000
     25%               2.000000     2.000000               2.000000
     50%               4.000000     4.000000               4.000000
     75%               6.000000     5.000000               5.000000
     max               9.000000     8.000000               8.000000
```

```
       Clubbing of Finger Nails  Frequent Cold    Dry Cough      Snoring
count               1000.000000    1000.000000  1000.000000  1000.000000
mean                   3.923000       3.536000     3.853000     2.926000
std                    2.388048       1.832502     2.039007     1.474686
min                    1.000000       1.000000     1.000000     1.000000
25%                    2.000000       2.000000     2.000000     2.000000
50%                    4.000000       3.000000     4.000000     3.000000
75%                    5.000000       5.000000     6.000000     4.000000
max                    9.000000       7.000000     7.000000     7.000000

[8 rows x 24 columns]
```

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

```
[ ]: index                       0
     Patient Id                  0
     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
```

```
[ ]:
```

```
data = data_true.drop(["index", "Patient Id", "Age", "Gender", "Level" ,"Dust␣
 ↪Allergy","chronic Lung Disease","Balanced Diet","Obesity","Chest␣
 ↪Pain","Coughing of Blood","Fatigue","Weight Loss","Snoring","Dry␣
 ↪Cough","Frequent Cold","Clubbing of Finger Nails","Swallowing␣
 ↪Difficulty","Wheezing","Shortness of Breath","Passive␣
 ↪Smoker","Smoking","OccuPational Hazards"], axis=1)
data.head(10)
```

```
[ ]:    Air Pollution  Alcohol use  Genetic Risk
     0              2            4             3
     1              3            1             4
     2              4            5             5
     3              7            7             6
     4              6            8             7
     5              4            5             5
     6              2            4             3
     7              3            1             2
     8              4            5             6
     9              2            3             4
```

```
[ ]: x = df[['Air Pollution','Alcohol use']]
     y = df['Genetic Risk']
```

```
[ ]: k = 3
     knn = KNeighborsClassifier(n_neighbors=k)
     knn.fit(x,y)
```

```
[ ]: KNeighborsClassifier(n_neighbors=3)
```

```
[ ]: # New data point to predict on
     new_data = np.array([[6,8]])
     prediction = knn.predict(new_data)

     # Check the prediction and print the result
     if prediction[0] == 0:  # Access the prediction string from the array
         print("low Risk")
     else:
         print("high Risk")
```

```
high Risk
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but KNeighborsClassifier was fitted with feature
names
  warnings.warn(
```

[ ]:

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
```

[ ]:

```
from sklearn.linear_model import LinearRegression
LR = LinearRegression()
```

[ ]:

```
from sklearn.model_selection import train_test_split as ttp
from sklearn.metrics import classification_report
```

[ ]:

```
# Assuming 'y' is your target variable and it contains strings like 'yes' and␣
 ↪'no'
# Convert 'yes' to 1 and 'no' to 0
y_numeric = y.replace({'yes': 1, 'no': 0})

# Now fit the model with the numeric target variable
LR.fit(x, y_numeric)
```

[ ]:

```
LinearRegression()
```

[ ]:

```
new_data = np.array([[6, 35]])
prediction = LR.predict(new_data)[0]
# Check the prediction and print the result
if prediction == 0:   # Removed indexing as 'prediction' is a scalar
    print("low Risk")
else:
    print("high Risk")
```

```
high Risk
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:439: UserWarning: X does
not have valid feature names, but LinearRegression was fitted with feature names
  warnings.warn(
```

[ ]:

```
# Install necessary libraries if not already installed
!pip install -q pandas scikit-learn matplotlib

# Import necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import classification_report, confusion_matrix
import matplotlib.pyplot as plt
```

```
[ ]: # Load data (replace 'lung_cancer_data.csv' with your actual data file)
     data = pd.read_csv('/content/drive/MyDrive/lung cancer/cancer patient data sets.
      ↪csv')

     # Display first few rows to understand the data
     print(data.head())

     # Define independent variables (features) and dependent variable (target)
     X = data[['Air Pollution', 'Alcohol use']].values  # Features
     y = data['Genetic Risk'].values  # Target variable

     # Split data into training and testing sets
     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
      ↪random_state=42)

     # Standardize features (optional but recommended for logistic regression)
     scaler = StandardScaler()
     X_train = scaler.fit_transform(X_train)
     X_test = scaler.transform(X_test)
```

```
   index Patient Id  Age  Gender  Air Pollution  Alcohol use  Dust Allergy  \
0      0         P1   33       1              2            4             5
1      1        P10   17       1              3            1             5
2      2       P100   35       1              4            5             6
3      3      P1000   37       1              7            7             7
4      4       P101   46       1              6            8             7

   OccuPational Hazards  Genetic Risk  chronic Lung Disease  ...  Fatigue  \
0                     4             3                     2  ...        3
1                     3             4                     2  ...        1
2                     5             5                     4  ...        8
3                     7             6                     7  ...        4
4                     7             7                     6  ...        3

   Weight Loss  Shortness of Breath  Wheezing  Swallowing Difficulty  \
0            4                    2         2                      3
1            3                    7         8                      6
2            7                    9         2                      1
3            2                    3         1                      4
4            2                    4         1                      4

   Clubbing of Finger Nails  Frequent Cold  Dry Cough  Snoring   Level
0                         1              2          3        4     Low
1                         2              1          7        2  Medium
2                         4              6          7        2    High
3                         5              6          7        5    High
4                         2              4          2        3    High
```

```
[5 rows x 26 columns]
```

[ ]:

```
# Initialize logistic regression model
model = LogisticRegression()

# Train the model
model.fit(X_train, y_train)
```

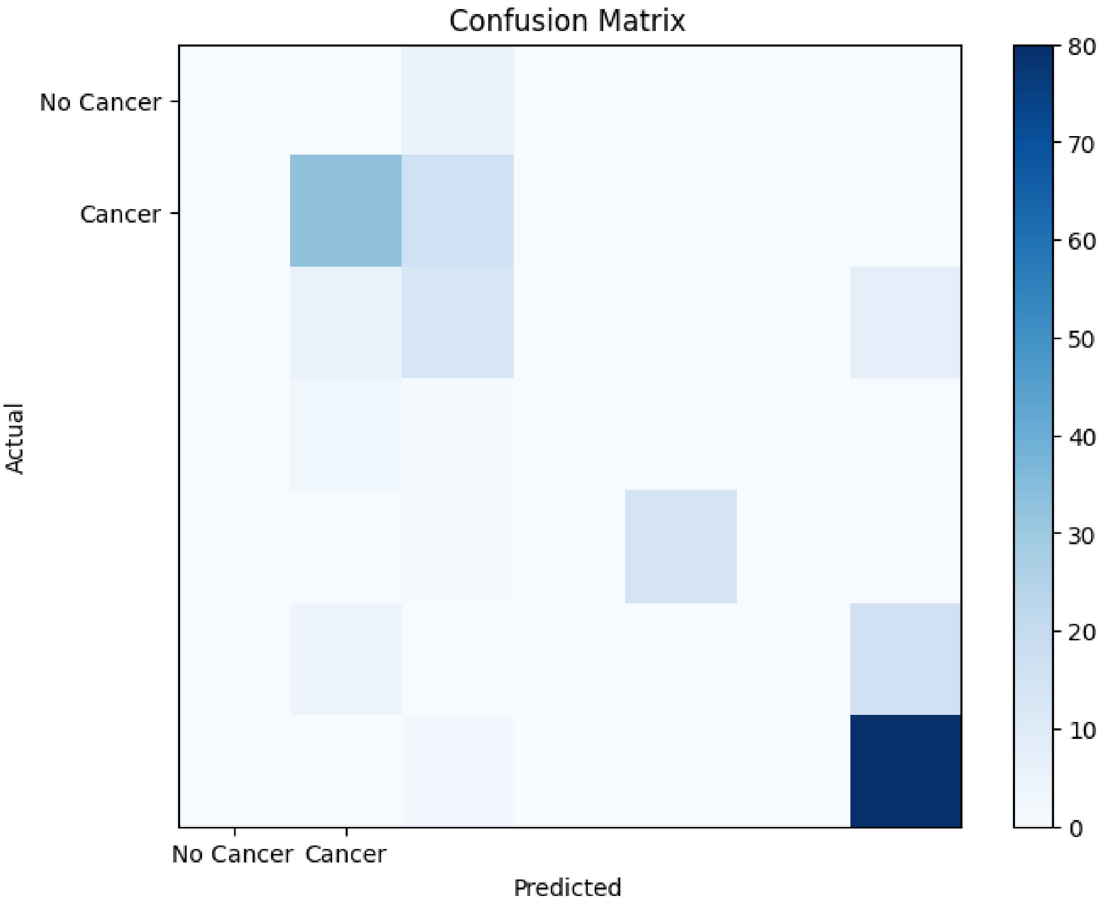
[ ]:

```
LogisticRegression()
```

[ ]:

```
# Predict on test data
y_pred = model.predict(X_test)

# Print classification report
print(classification_report(y_test, y_pred))

# Plot confusion matrix
conf_mat = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
plt.imshow(conf_mat, cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
plt.xticks([0, 1], ['No Cancer', 'Cancer'])
plt.yticks([0, 1], ['No Cancer', 'Cancer'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         5
           2       0.73      0.67      0.70        49
           3       0.34      0.52      0.41        25
           4       0.00      0.00      0.00         4
           5       1.00      0.93      0.97        15
           6       0.00      0.00      0.00        20
           7       0.78      0.98      0.86        82

    accuracy                           0.70       200
   macro avg       0.41      0.44      0.42       200
weighted avg       0.62      0.70      0.65       200
```



```
[ ]: # Install necessary libraries if not already installed
     !pip install -q pandas scikit-learn matplotlib

     # Import necessary libraries
     import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.tree import DecisionTreeClassifier, plot_tree
     from sklearn.metrics import classification_report, confusion_matrix
     import matplotlib.pyplot as plt
```

[ ]:

```
# Load data (replace 'lung_cancer_data.csv' with your actual data file)
data = pd.read_csv('/content/drive/MyDrive/lung cancer/cancer patient data sets.
 ↪csv')

# Display first few rows to understand the data
print(data.head())

# Define independent variables (features) and dependent variable (target)
X = data[['Air Pollution', 'Alcohol use']].values  # Features
y = data['Genetic Risk'].values  # Target variable

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

```
   index Patient Id  Age  Gender  Air Pollution  Alcohol use  Dust Allergy  \
0      0         P1   33       1              2            4             5
1      1        P10   17       1              3            1             5
2      2       P100   35       1              4            5             6
3      3      P1000   37       1              7            7             7
4      4       P101   46       1              6            8             7

   OccuPational Hazards  Genetic Risk  chronic Lung Disease  …  Fatigue  \
0                     4             3                     2  …        3
1                     3             4                     2  …        1
2                     5             5                     4  …        8
3                     7             6                     7  …        4
4                     7             7                     6  …        3

   Weight Loss  Shortness of Breath  Wheezing  Swallowing Difficulty  \
0            4                    2         2                      3
1            3                    7         8                      6
2            7                    9         2                      1
3            2                    3         1                      4
4            2                    4         1                      4

   Clubbing of Finger Nails  Frequent Cold  Dry Cough  Snoring   Level
0                         1              2          3        4     Low
1                         2              1          7        2  Medium
2                         4              6          7        2    High
3                         5              6          7        5    High
4                         2              4          2        3    High

[5 rows x 26 columns]
```

[ ]:

```
# Initialize decision tree classifier
clf = DecisionTreeClassifier(random_state=42)

# Train the model
clf.fit(X_train, y_train)
```

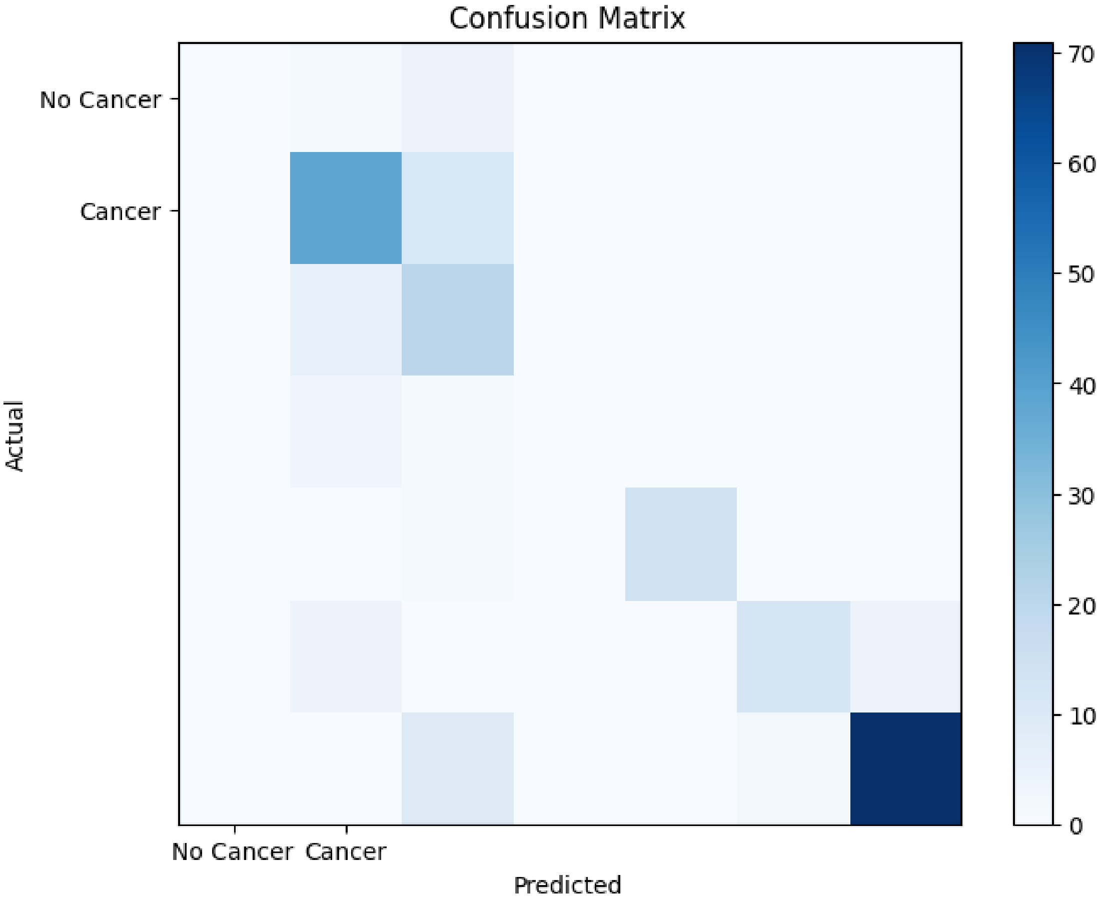
[ ]:

```
DecisionTreeClassifier(random_state=42)
```

[ ]:

```
# Predict on test data
y_pred = clf.predict(X_test)

# Print classification report
print(classification_report(y_test, y_pred))

# Plot confusion matrix
conf_mat = confusion_matrix(y_test, y_pred)
plt.figure(figsize=(8, 6))
plt.imshow(conf_mat, cmap=plt.cm.Blues)
plt.title('Confusion Matrix')
plt.colorbar()
plt.xticks([0, 1], ['No Cancer', 'Cancer'])
plt.yticks([0, 1], ['No Cancer', 'Cancer'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         5
           2       0.75      0.78      0.76        49
           3       0.43      0.80      0.56        25
           4       0.00      0.00      0.00         4
           5       1.00      0.93      0.97        15
           6       0.86      0.60      0.71        20
           7       0.95      0.87      0.90        82

    accuracy                           0.78       200
   macro avg       0.57      0.57      0.56       200
weighted avg       0.79      0.78      0.77       200
```



```
[ ]: # Install necessary libraries if not already installed
     !pip install -q pandas scikit-learn matplotlib

     # Import necessary libraries
     import pandas as pd
     from sklearn.model_selection import train_test_split
     from sklearn.ensemble import RandomForestClassifier
     from sklearn.metrics import classification_report
     import matplotlib.pyplot as plt
```

[ ]:

```python
# Load data (replace 'lung_cancer_data.csv' with your actual data file)
data = pd.read_csv('/content/drive/MyDrive/lung cancer/cancer patient data sets.
  ↪csv')

# Display first few rows to understand the data
print(data.head())

# Define independent variables (features) and dependent variable (target)
X = data[['Air Pollution', 'Alcohol use']].values  # Features
y = data['Genetic Risk'].values  # Target variable

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
  ↪random_state=42)
```

```
   index Patient Id  Age  Gender  Air Pollution  Alcohol use  Dust Allergy  \
0      0         P1   33       1              2            4             5
1      1        P10   17       1              3            1             5
2      2       P100   35       1              4            5             6
3      3      P1000   37       1              7            7             7
4      4       P101   46       1              6            8             7

   OccuPational Hazards  Genetic Risk  chronic Lung Disease  …  Fatigue  \
0                     4             3                     2  …        3
1                     3             4                     2  …        1
2                     5             5                     4  …        8
3                     7             6                     7  …        4
4                     7             7                     6  …        3

   Weight Loss  Shortness of Breath  Wheezing  Swallowing Difficulty  \
0            4                    2         2                      3
1            3                    7         8                      6
2            7                    9         2                      1
3            2                    3         1                      4
4            2                    4         1                      4

   Clubbing of Finger Nails  Frequent Cold  Dry Cough  Snoring   Level
0                         1              2          3        4     Low
1                         2              1          7        2  Medium
2                         4              6          7        2    High
3                         5              6          7        5    High
4                         2              4          2        3    High

[5 rows x 26 columns]
```

[ ]:

```python
# Initialize Random Forest classifier
rfc = RandomForestClassifier(n_estimators=100, random_state=42)
```

```

# Train the model
rfc.fit(X_train, y_train)
```

[ ]: RandomForestClassifier(random_state=42)

[ ]:
```
# Predict on test data
y_pred = rfc.predict(X_test)

# Print classification report
print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         5
           2       0.75      0.78      0.76        49
           3       0.41      0.52      0.46        25
           4       0.00      0.00      0.00         4
           5       1.00      0.93      0.97        15
           6       0.86      0.60      0.71        20
           7       0.88      0.95      0.91        82

    accuracy                           0.78       200
   macro avg       0.55      0.54      0.54       200
weighted avg       0.75      0.78      0.76       200


/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.10/dist-packages/sklearn/metrics/_classification.py:1344:
UndefinedMetricWarning: Precision and F-score are ill-defined and being set to
0.0 in labels with no predicted samples. Use `zero_division` parameter to
control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```