

```
import pandas as pd

df = pd.read_csv('/content/survey_lung_cancer.csv')

print("First 5 rows of the dataset:")
display(df.head())

print("\nColumn names and their data types:")
display(df.info())
```

First 5 rows of the dataset:

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	CHRONIC DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL CONSUMING	COUGHING	SHORT OF BR
0	M	69	1	2	2	1	1	2	1	2	2	2	
1	M	74	2	1	1	1	2	2	2	1	1	1	
2	F	59	1	1	1	2	1	2	1	2	1	2	
3	M	63	2	2	2	1	1	1	1	1	2	1	
4	F	63	1	2	1	1	1	1	1	2	1	2	

Column names and their data types:

```
<class 'pandas.core.frame.DataFrame'>
```

```
RangeIndex: 309 entries, 0 to 308
```

```
Data columns (total 16 columns):
```

#	Column	Non-Null Count	Dtype
0	GENDER	309 non-null	object
1	AGE	309 non-null	int64
2	SMOKING	309 non-null	int64
3	YELLOW_FINGERS	309 non-null	int64
4	ANXIETY	309 non-null	int64
5	PEER_PRESSURE	309 non-null	int64
6	CHRONIC DISEASE	309 non-null	int64
7	FATIGUE	309 non-null	int64
8	ALLERGY	309 non-null	int64
9	WHEEZING	309 non-null	int64
10	ALCOHOL CONSUMING	309 non-null	int64
11	COUGHING	309 non-null	int64
12	SHORTNESS OF BREATH	309 non-null	int64
13	SWALLOWING DIFFICULTY	309 non-null	int64
14	CHEST PAIN	309 non-null	int64
15	LUNG_CANCER	309 non-null	object

```
dtypes: int64(14), object(2)
```

```
memory usage: 38.8+ KB
```

```
None
```

```
from sklearn.model_selection import train_test_split
```

```
X = df.drop('LUNG_CANCER', axis=1)
```

```
y = df['LUNG_CANCER']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

```
print("Shape of X_train:", X_train.shape)
```

```
print("Shape of X_test:", X_test.shape)
```

```
print("Shape of y_train:", y_train.shape)
```

```
print("Shape of y_test:", y_test.shape)
```

```
Shape of X_train: (247, 15)
```

```
Shape of X_test: (62, 15)
```

```
Shape of y_train: (247,)
```

```
Shape of y_test: (62,)
```

```
from sklearn.preprocessing import StandardScaler
```

```
scaler = StandardScaler()
```

```
X_train_scaled = scaler.fit_transform(X_train)
```

```
X_test_scaled = scaler.transform(X_test)
```

```
print("Shape of X_train_scaled:", X_train_scaled.shape)
```

```
print("Shape of X_test_scaled:", X_test_scaled.shape)
```

```
Shape of X_train_scaled: (247, 15)
```

```
Shape of X_test_scaled: (62, 15)
```

```
from sklearn.linear_model import LogisticRegression
```

```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

```
model = LogisticRegression()
```

```

model.fit(X_train_scaled, y_train)

y_pred = model.predict(X_test_scaled)

accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)

print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")

```

```

Accuracy: 0.9677
Precision: 0.9833
Recall: 0.9833
F1-score: 0.9833

```

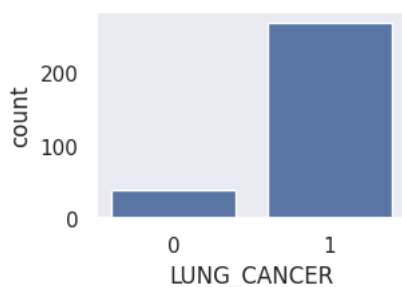
```

import matplotlib.pyplot as plt
import seaborn as sns

plt.figure(figsize=(3, 2))
sns.countplot(data=df, x='LUNG_CANCER')

```

<Axes: xlabel='LUNG\_CANCER', ylabel='count'>



encoding

x and y me break

train test split

standard scaler

model *train*

```

from sklearn.preprocessing import LabelEncoder

label_encoder = LabelEncoder()
df['LUNG_CANCER'] = label_encoder.fit_transform(df['LUNG_CANCER'])
df['GENDER'] = label_encoder.fit_transform(df['GENDER'])

print(df.head())

```

	GENDER	AGE	SMOKING	YELLOW_FINGERS	ANXIETY	PEER_PRESSURE	\
0	1	69	1	2	2	1	
1	1	74	2	1	1	1	
2	0	59	1	1	1	2	
3	1	63	2	2	2	1	
4	0	63	1	2	1	1	

	CHRONIC_DISEASE	FATIGUE	ALLERGY	WHEEZING	ALCOHOL_CONSUMING	COUGHING	\
0	1	2	1	2	2	2	
1	2	2	2	1	1	1	
2	1	2	1	2	1	2	
3	1	1	1	1	2	1	
4	1	1	1	2	1	2	

	SHORTNESS_OF_BREATH	SWALLOWING_DIFFICULTY	CHEST_PAIN	LUNG_CANCER
0	2	2	2	1
1	2	2	2	1
2	2	1	2	0
3	1	2	2	0
4	2	1	1	0

```

from sklearn.svm import SVC

model=SVC()
model.fit(X_train,y_train)

```

▼ SVC ⓘ ?  
SVC()

```
model.score(X_test,y_test)*100,model.score(X_train,y_train)*100
```

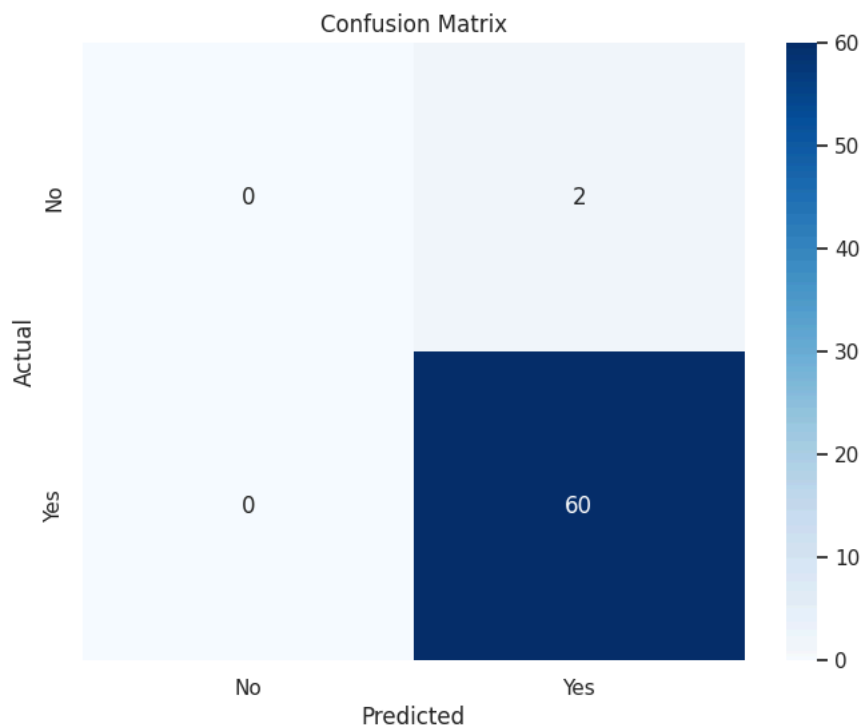
```
(96.7741935483871, 85.02024291497976)
```

```
y_pred=model.predict(X_test)
```

```
from sklearn.metrics import confusion_matrix
cm=confusion_matrix(y_test,y_pred)
cm
```

```
array([[ 0,  2],
       [ 0, 60]])
```

```
# Visualize the confusion matrix
plt.figure(figsize=(8, 6))
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['No', 'Yes'], yticklabels=['No', 'Yes'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```



```
from sklearn.metrics import classification_report
print(classification_report(y_test,y_pred))
```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	2
1	0.97	1.00	0.98	60
accuracy			0.97	62
macro avg	0.48	0.50	0.49	62
weighted avg	0.94	0.97	0.95	62

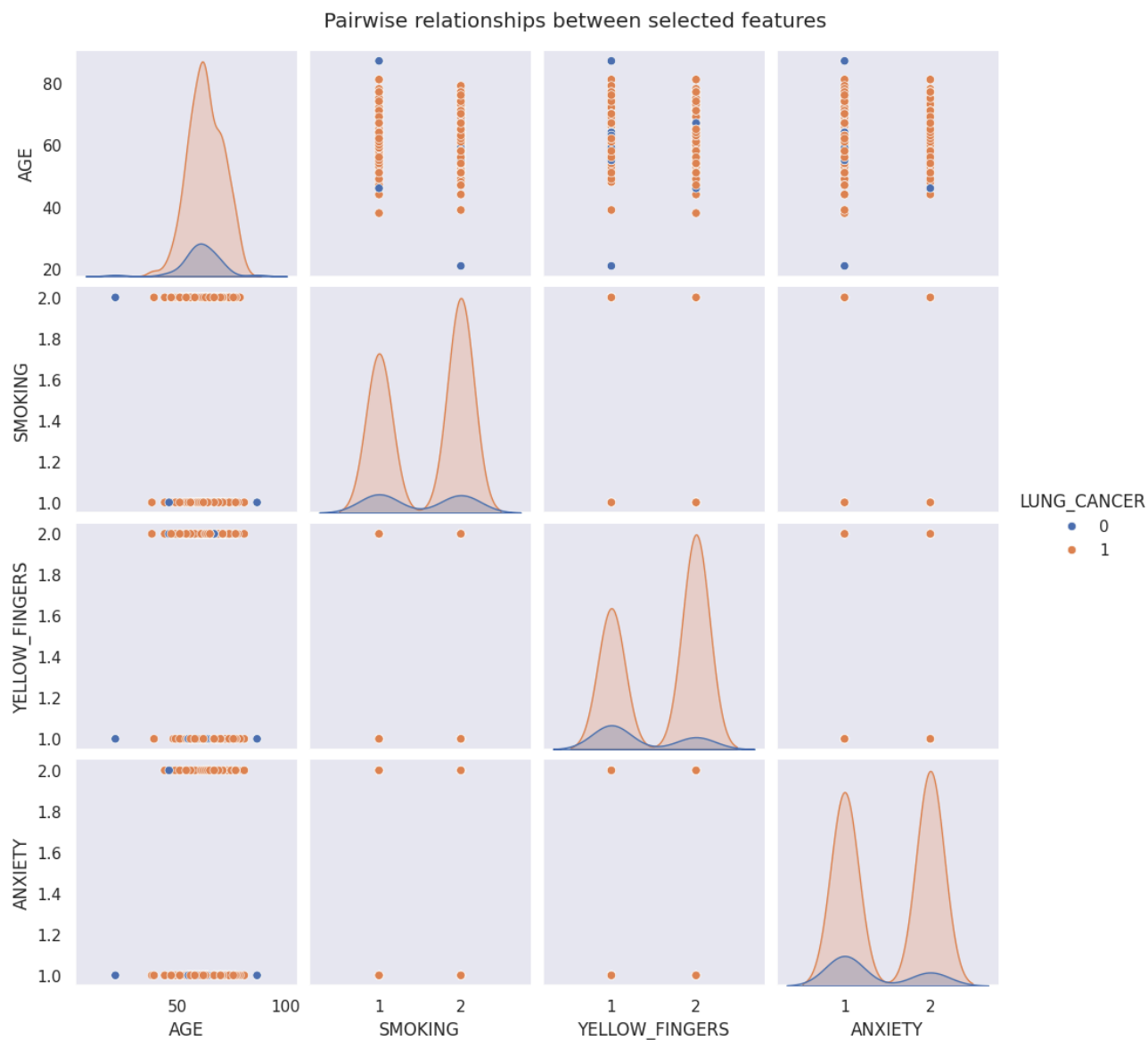
```
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defir
_warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defir
_warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
/usr/local/lib/python3.12/dist-packages/sklearn/metrics/_classification.py:1565: UndefinedMetricWarning: Precision is ill-defir
_warn_prf(average, modifier, f"{metric.capitalize()} is", len(result))
```

```
# Get categorical columns
categorical_cols = df.select_dtypes(include=['object', 'category']).columns
```

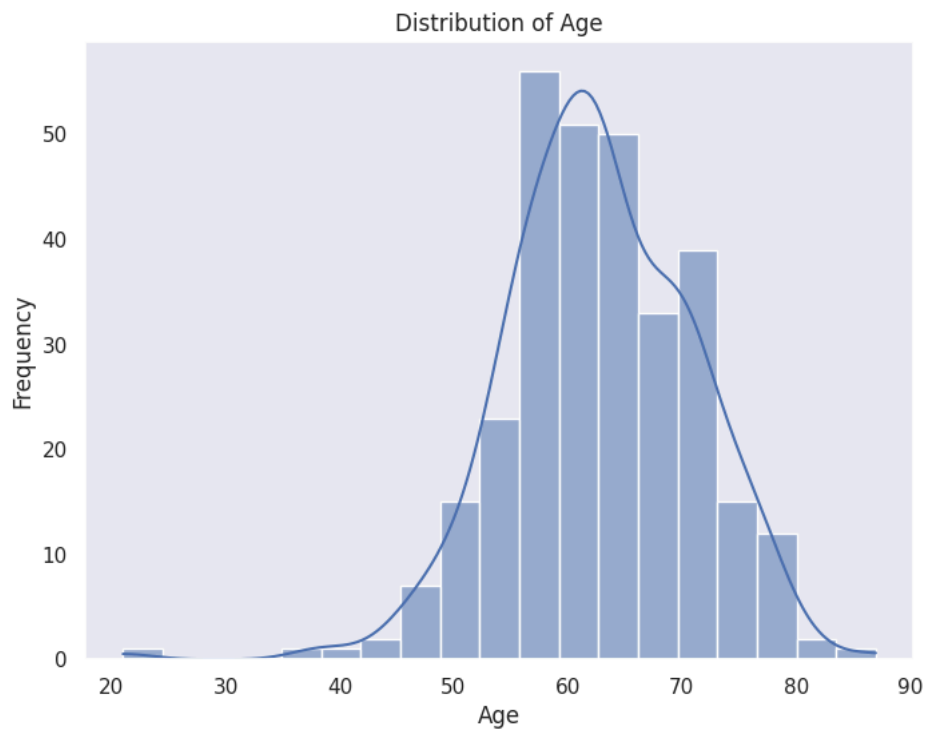
```
# Create bar plots for categorical features
```

```
for col in categorical_cols:
    print(f"\nAnalyzing column: {col}")
    plt.figure(figsize=(8, 4))
    sns.countplot(data=df, x=col, order=df[col].value_counts().index)
    plt.title(f'Count of {col}')
    plt.xlabel(col)
    plt.ylabel('Count')
    plt.show()
```

```
sns.pairplot(df[['AGE', 'SMOKING', 'YELLOW_FINGERS', 'ANXIETY', 'LUNG_CANCER']], hue='LUNG_CANCER')
plt.suptitle('Pairwise relationships between selected features', y=1.02)
plt.show()
```

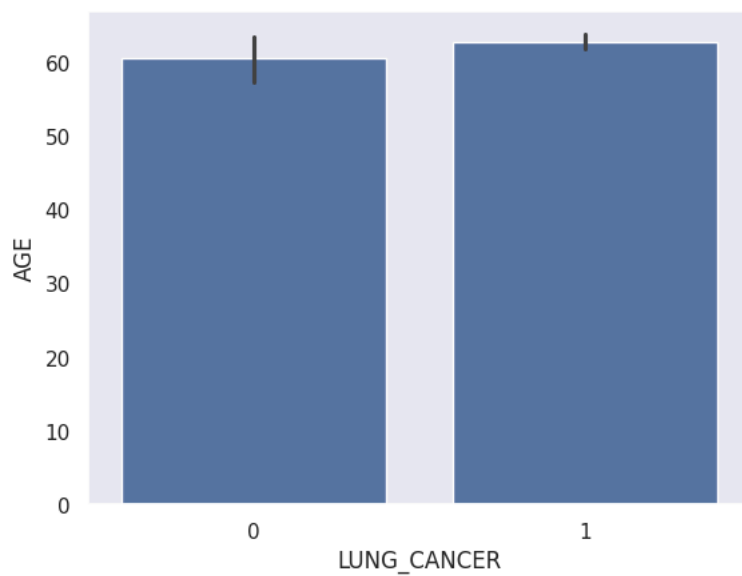


```
plt.figure(figsize=(8, 6))
sns.histplot(data=df, x='AGE', kde=True)
plt.title('Distribution of Age')
plt.xlabel('Age')
plt.ylabel('Frequency')
plt.show()
```

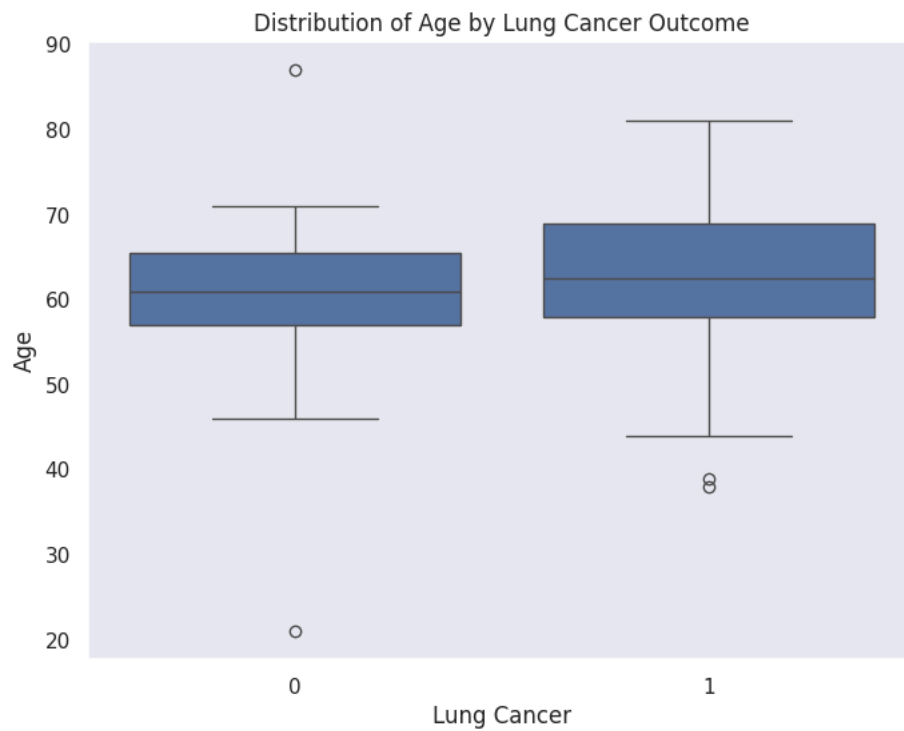


```
sns.barplot(x='LUNG_CANCER', y='AGE', data=df)
```

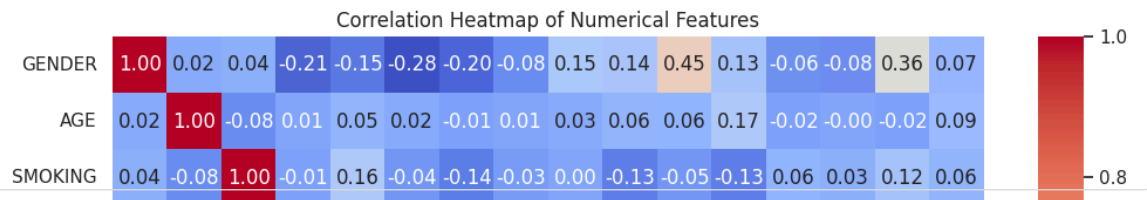
<Axes: xlabel='LUNG\_CANCER', ylabel='AGE'>



```
plt.figure(figsize=(8, 6))
sns.boxplot(data=df, x='LUNG_CANCER', y='AGE')
plt.title('Distribution of Age by Lung Cancer Outcome')
plt.xlabel('Lung Cancer')
plt.ylabel('Age')
plt.show()
```



```
plt.figure(figsize=(12, 10))
sns.heatmap(df.corr(), annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Heatmap of Numerical Features')
plt.show()
```



```
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
```

```
y_pred = model.predict(X_test)
```

```
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred)
recall = recall_score(y_test, y_pred)
f1 = f1_score(y_test, y_pred)
```

```
print(f"Accuracy: {accuracy:.4f}")
print(f"Precision: {precision:.4f}")
print(f"Recall: {recall:.4f}")
print(f"F1-score: {f1:.4f}")
```

```
WHEEZING 0.14 0.06 -0.13 -0.08 -0.19 -0.07 -0.05 0.14 0.17 1.00 0.27 0.37 0.04 0.07 0.15 0.25
```