In [1]:

```
import numpy as np
import pandas as pd

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import LinearSVC, SVC
from sklearn.neural_network import MLPClassifier
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
```

In [2]:

```
data = pd.read_csv("diabetes_data_upload.csv")
```

In [3]:

```
data
```

Out[3]:

	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	visual blurring	Itching
0	40	Male	No	Yes	No	Yes	No	No	No	Yes
1	58	Male	No	No	No	Yes	No	No	Yes	No
2	41	Male	Yes	No	No	Yes	Yes	No	No	Yes
3	45	Male	No	No	Yes	Yes	Yes	Yes	No	Yes
4	60	Male	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
...	...	...	...	...	...	...	...	...	...	...
515	39	Female	Yes	Yes	Yes	No	Yes	No	No	Yes
516	48	Female	Yes	Yes	Yes	Yes	Yes	No	No	Yes
517	58	Female	Yes	Yes	Yes	Yes	Yes	No	Yes	No
518	32	Female	No	No	No	Yes	No	No	Yes	Yes
519	42	Male	No	No	No	No	No	No	No	No

520 rows × 17 columns

In [4]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 520 entries, 0 to 519
Data columns (total 17 columns):
 #   Column              Non-Null Count  Dtype 
---  ------              --------------  ----- 
 0   Age                 520 non-null    int64 
 1   Gender              520 non-null    object
 2   Polyuria            520 non-null    object
 3   Polydipsia          520 non-null    object
 4   sudden weight loss  520 non-null    object
 5   weakness            520 non-null    object
 6   Polyphagia          520 non-null    object
 7   Genital thrush      520 non-null    object
 8   visual blurring     520 non-null    object
 9   Itching             520 non-null    object
 10  Irritability        520 non-null    object
 11  delayed healing     520 non-null    object
 12  partial paresis     520 non-null    object
 13  muscle stiffness    520 non-null    object
 14  Alopecia            520 non-null    object
 15  Obesity             520 non-null    object
 16  class               520 non-null    object
dtypes: int64(1), object(16)
memory usage: 69.2+ KB
```

Preprocessing

In [5]:

```
{column: len(data[column].unique()) for column in data.columns}
```

Out[5]:

```
{'Age': 51,
 'Gender': 2,
 'Polyuria': 2,
 'Polydipsia': 2,
 'sudden weight loss': 2,
 'weakness': 2,
 'Polyphagia': 2,
 'Genital thrush': 2,
 'visual blurring': 2,
 'Itching': 2,
 'Irritability': 2,
 'delayed healing': 2,
 'partial paresis': 2,
 'muscle stiffness': 2,
 'Alopecia': 2,
 'Obesity': 2,
 'class': 2}
```

In [18]:

```
def preprocess_inputs(df):
    df = df.copy()

    # Binary-encode Gender column
    df['Gender'] = df['Gender'].replace({'Female': 0, 'Male': 1})

    # Binary-encode the symptoms column
    for column in df.columns.drop(['Age', 'Gender', 'class']):
        df[column] = df[column].replace({'No': 0, 'Yes': 1})

    # Split df into X and y
    y = df['class']
    X = df.drop('class', axis=1)

    # Train-test split
    X_train, X_test, y_train, y_test = train_test_split(X, y, train_size=0.7, shuffl

    # Scale X
    scaler = StandardScaler()
    scaler.fit(X_train)
    X_train = pd.DataFrame(scaler.transform(X_train), index=X_train.index, columns=X
    X_test = pd.DataFrame(scaler.transform(X_test), index=X_test.index, columns=X_te

    return X_train, X_test, y_train, y_test
```

In [19]:

```
X_train, X_test, y_train, y_test = preprocess_inputs(data)
```

In [8]:

```
X_train
```

Out[8]:

	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	vis blurri

	Age	Gender	Polyuria	Polydipsia	sudden weight loss	weakness	Polyphagia	Genital thrush	vis blurri
122	-0.658902	0.740902	-0.994521	1.129159	-0.846747	0.841974	1.104315	-0.560428	-0.870
168	-0.913060	0.740902	-0.994521	-0.885615	-0.846747	0.841974	-0.905539	-0.560428	-0.870
23	0.018852	0.740902	-0.994521	1.129159	1.180990	0.841974	-0.905539	-0.560428	1.148
13	1.120204	0.740902	1.005510	1.129159	1.180990	0.841974	1.104315	1.784351	1.148
61	-1.082499	-1.349706	1.005510	1.129159	1.180990	0.841974	1.104315	-0.560428	1.148
...	...	...	...	...	...	...	...	...	
129	0.018852	0.740902	1.005510	1.129159	1.180990	0.841974	-0.905539	-0.560428	-0.870
144	1.713239	0.740902	1.005510	1.129159	-0.846747	-1.187685	1.104315	-0.560428	1.148
72	1.459081	-1.349706	-0.994521	-0.885615	-0.846747	-1.187685	-0.905539	1.784351	-0.870
235	-1.844973	0.740902	-0.994521	-0.885615	-0.846747	-1.187685	-0.905539	-0.560428	-0.870
37	1.289643	0.740902	1.005510	1.129159	1.180990	0.841974	1.104315	-0.560428	1.148

364 rows × 16 columns

In [9]:

```
y_train
```

Out[9]:

```
122    Positive
168    Positive
23     Positive
13     Positive
61     Positive
         ...   
129    Positive
144    Positive
72     Positive
235    Negative
37     Positive
Name: class, Length: 364, dtype: object
```

Training

In [10]:

```
models = {
    "                   Logistic Regression": LogisticRegression(),
    "                   K-Nearest Neighbors": KNeighborsClassifier(),
    "                         Decision Tree": DecisionTreeClassifier(),
    "Support Vector Machine (Linear Kernel)": LinearSVC(),
    "   Support Vector Machine (RBF Kernel)": SVC(),
    "                        Neural Network": MLPClassifier(),
    "                         Random Forest": RandomForestClassifier(),
    "                     Gradient Boosting": GradientBoostingClassifier()
}

for name, model in models.items():
    model.fit(X_train, y_train)
    print(name + " trained.")
```

```
                   Logistic Regression trained.
                   K-Nearest Neighbors trained.
                         Decision Tree trained.
```

```
Support Vector Machine (Linear Kernel) trained.
   Support Vector Machine (RBF Kernel) trained.
```

```
C:\Users\gtgau\anaconda3\lib\site-packages\sklearn\neural_network\_multilayer_percep
tron.py:582: ConvergenceWarning: Stochastic Optimizer: Maximum iterations (200) reac
hed and the optimization hasn't converged yet.
  warnings.warn(
                        Neural Network trained.
                         Random Forest trained.
                     Gradient Boosting trained.
```

Results

```
In [11]:
for name, model in models.items():
    print(name + ": {:.2f}%".format(model.score(X_test, y_test) * 100))
```

```
                   Logistic Regression: 92.31%
                   K-Nearest Neighbors: 90.38%
                         Decision Tree: 96.15%
Support Vector Machine (Linear Kernel): 92.31%
   Support Vector Machine (RBF Kernel): 95.51%
                        Neural Network: 96.79%
                         Random Forest: 98.08%
                     Gradient Boosting: 98.08%
```

```
In [12]:
for name, model in models.items():
    print(model.predict(X_test) )
```

```
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Negative' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Negative' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Negative' 'Negative' 'Positive' 'Negative' 'Positive' 'Negative'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Negative' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Negative' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
['Negative' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Negative' 'Negative' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Negative' 'Positive' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Positive' 'Negative' 'Positive'
 'Positive' 'Positive' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Negative' 'Positive' 'Positive' 'Positive' 'Positive'
 'Positive' 'Positive' 'Negative' 'Positive' 'Positive' 'Positive'
 'Negative' 'Positive' 'Positive' 'Positive' 'Negative' 'Positive'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Negative' 'Positive' 'Positive' 'Positive' 'Positive' 'Negative'
 'Positive' 'Negative' 'Positive' 'Positive' 'Negative' 'Negative'
 'Positive' 'Positive' 'Negative' 'Negative' 'Positive' 'Positive'
 'Positive' 'Negative' 'Negative' 'Negative' 'Positive' 'Positive']
```

In [13]:

```
model=RandomForestClassifier()
```

In [14]:

```
model.fit(X_train, y_train)
```

Out[14]:

```
RandomForestClassifier()
```

In [15]:

```
model.score(X_test, y_test)
```

Out[15]:

```
0.9807692307692307
```

In [16]:

```
model.predict(X_test)
```

Out[16]:

```
array(['Negative', 'Negative', 'Negative', 'Negative', 'Positive',
       'Positive', 'Positive', 'Positive', 'Negative', 'Positive',
       'Negative', 'Negative', 'Positive', 'Positive', 'Negative',
       'Positive', 'Negative', 'Negative', 'Positive', 'Negative',
       'Positive', 'Positive', 'Positive', 'Negative', 'Positive',
       'Negative', 'Positive', 'Positive', 'Negative', 'Positive',
       'Positive', 'Positive', 'Positive', 'Positive', 'Negative',
       'Positive', 'Positive', 'Negative', 'Negative', 'Positive',
       'Negative', 'Negative', 'Positive', 'Positive', 'Positive',
       'Positive', 'Positive', 'Negative', 'Positive', 'Positive',
       'Negative', 'Positive', 'Positive', 'Positive', 'Positive',
       'Positive', 'Negative', 'Negative', 'Negative', 'Negative',
       'Positive', 'Negative', 'Positive', 'Positive', 'Negative',
       'Negative', 'Positive', 'Positive', 'Positive', 'Positive',
       'Positive', 'Positive', 'Negative', 'Positive', 'Negative',
       'Positive', 'Positive', 'Positive', 'Positive', 'Positive',
       'Positive', 'Positive', 'Positive', 'Negative', 'Positive',
       'Negative', 'Negative', 'Positive', 'Positive', 'Positive',
       'Positive', 'Negative', 'Negative', 'Positive', 'Negative',
       'Positive', 'Positive', 'Positive', 'Positive', 'Positive',
       'Positive', 'Positive', 'Positive', 'Positive', 'Negative',
       'Positive', 'Positive', 'Positive', 'Negative', 'Negative',
       'Positive', 'Positive', 'Positive', 'Positive', 'Positive',
       'Positive', 'Negative', 'Positive', 'Positive', 'Positive',
       'Negative', 'Positive', 'Positive', 'Positive', 'Negative',
       'Positive', 'Positive', 'Negative', 'Positive', 'Positive',
       'Negative', 'Negative', 'Negative', 'Positive', 'Positive',
       'Positive', 'Positive', 'Negative', 'Positive', 'Negative',
       'Positive', 'Positive', 'Negative', 'Negative', 'Positive',
       'Positive', 'Negative', 'Negative', 'Positive', 'Positive',
       'Positive', 'Negative', 'Negative', 'Negative', 'Positive',
       'Positive'], dtype=object)
```

In [17]:

```
y_test
```

Out[17]:

```
273    Negative
272    Negative
329    Negative
480    Negative
173    Positive
         ...
335    Negative
407    Negative
330    Negative
257    Positive
95     Positive
Name: class, Length: 156, dtype: object
```

In [ ]: