Random Forest For Audio Input

In [3]:

```
# Importing our Utilities functions:
from utils import CautDataloaderRegular
import os
import pandas as pd
import numpy as np
import math
import matplotlib.pyplot as plt


# Modelling
import pandas as pd
import tensorflow as tf
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense, Dropout
from keras.layers import LSTM

from sklearn.metrics import accuracy_score, confusion_matrix, precision_score, recall_scor
from sklearn.model_selection import RandomizedSearchCV, train_test_split
from scipy.stats import randint

def RFR_grid_search(csv_path,
                    data_dir,
                    feature_type="MFCC",
                    threshold=0.5):

    X_y_data = CautDataloaderRegular.get_X_y_TrainTest_Audio(csv_path=csv_path,
                                                             data_dir=data_dir,
                                                             feature_type=feature_type,
                                                             input_length_in_seconds=3,
                                                             class_to_num_dict={"truth": (
                                                             verbose=False)

    X_train, y_train = X_y_data[0].reshape(-1, X_y_data[0].shape[1] * X_y_data[0].shape[2]
    X_test, y_test = X_y_data[2].reshape(-1, X_y_data[2].shape[1] * X_y_data[2].shape[2]),

    params = {
        'n_estimators': randint(50,500),
        'max_depth': randint(1,20)
    }
    rfc = RandomForestClassifier()
    rand_search = RandomizedSearchCV(rfc,
                                param_distributions = params,
                                n_iter=5,
                                cv=5)
    rand_search.fit(X_train, y_train)
    best_rf = rand_search.best_estimator_
    return best_rf.get_params()['max_depth'], best_rf.get_params()['n_estimators']

def RFR_Model_Audio(csv_path,
                    data_dir,
                    feature_type="MFCC",
                    threshold=0.5,
                    train_verbose=True):

    X_y_data = CautDataloaderRegular.get_X_y_TrainTest_Audio(csv_path=csv_path,
                                                             data_dir=data_dir,
                                                             feature_type=feature_type,
                                                             input_length_in_seconds=3,
                                                             class_to_num_dict={"truth": (
                                                             verbose=train_verbose)
```

```
    X_train, y_train = X_y_data[0].reshape(-1, X_y_data[0].shape[1] * X_y_data[0].shape[2]
    X_test, y_test = X_y_data[2].reshape(-1, X_y_data[2].shape[1] * X_y_data[2].shape[2]),

    print(X_train.shape, y_train.shape, X_test.shape, y_test.shape)

    depth, estimators = RFR_grid_search(csv_path,
                                        data_dir,
                                        feature_type="MFCC",
                                        threshold=0.5)

    # Setup model:
    #fitting and evaluating
    rf = RandomForestClassifier(n_estimators=estimators, max_depth=depth)

    # fit the model:
    rf.fit(X_train, y_train)

     # predict on test data:
    y_pred = rf.predict(X_test)

    # get metrics (accuracy, precision, recall)
    accuracy = accuracy_score(y_test, y_pred)
    precision = precision_score(y_test, y_pred)
    recall = recall_score(y_test, y_pred)
    print("Accuracy:", accuracy)
    print("Precision", precision)
    print("Recall", recall)

    # show confusion matrix
    CautDataloaderRegular.plot_confusion_matrix(y_test, y_pred)
```

In [4]:

```
RFR_Model_Audio(csv_path="C:\\Work\\606Capstone\\Video_chunks\\CSV\\",
                data_dir="C:\\Work\\606Capstone\\Video_chunks\\audio_features\\",
                feature_type="MFCC",
                threshold=0.5,
                train_verbose=True)
```

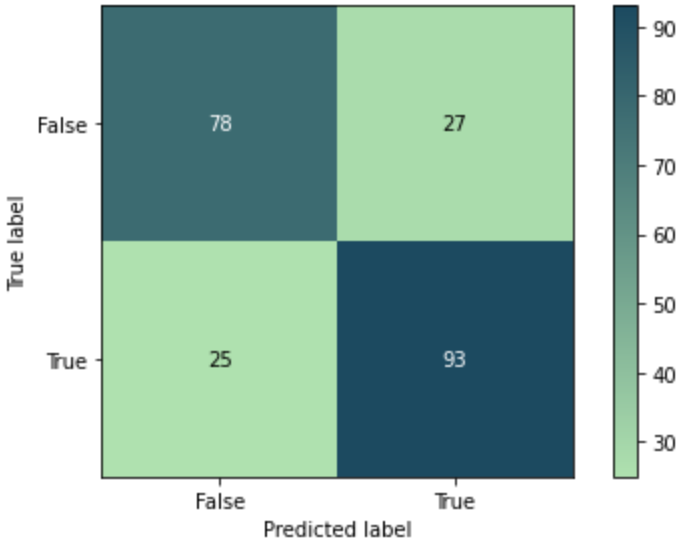
```
data_dir updated to: C:\Work\606Capstone\Video_chunks\audio_features\MFCC_audio_features

Selected csv_path: C:\Work\606Capstone\Video_chunks\CSV\
Processed 100 / 520
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 0
Processed 200 / 520
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 0
Processed 300 / 520
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 1
Processed 400 / 520
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 0
Processed 500 / 520
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 1
Processed 100 / 223
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 1
```

```
Processed 200 / 223
  - Audio sample shape & label:
    - X_data: (130, 20)
    - y_data: 1
----------------------------
Gathered data shapes:
X_train.shape: (520, 130, 20)
y_train.shape: (520,)
X_test.shape: (223, 130, 20)
y_test.shape: (223,)
(520, 2600) (520,) (223, 2600) (223,)
data_dir updated to: C:\Work\606Capstone\Video_chunks\audio_features\MFCC_audio_features

Selected csv_path: C:\Work\606Capstone\Video_chunks\CSV\


Accuracy: 0.7668161434977578
Precision 0.775
Recall 0.788135593220339
```




```
----------------------------------------------------------------

Metrics Rates:
     - True Positive            : 93
     - False Positive           : 27
     - True Negative            : 78
     - False Negative           : 25
     - True Positive Rate       :  0.788135593220339
     - True Negative Rate       :  0.7428571428571429
     - Positive Predictive Value:  0.775
     - Negative predictive value:  0.7572815533980582
     - False Positive Rate      :  0.2571428571428571
     - False Negative Rate      :  0.211864406779661
     - False Discovery Rate     :  0.225
```