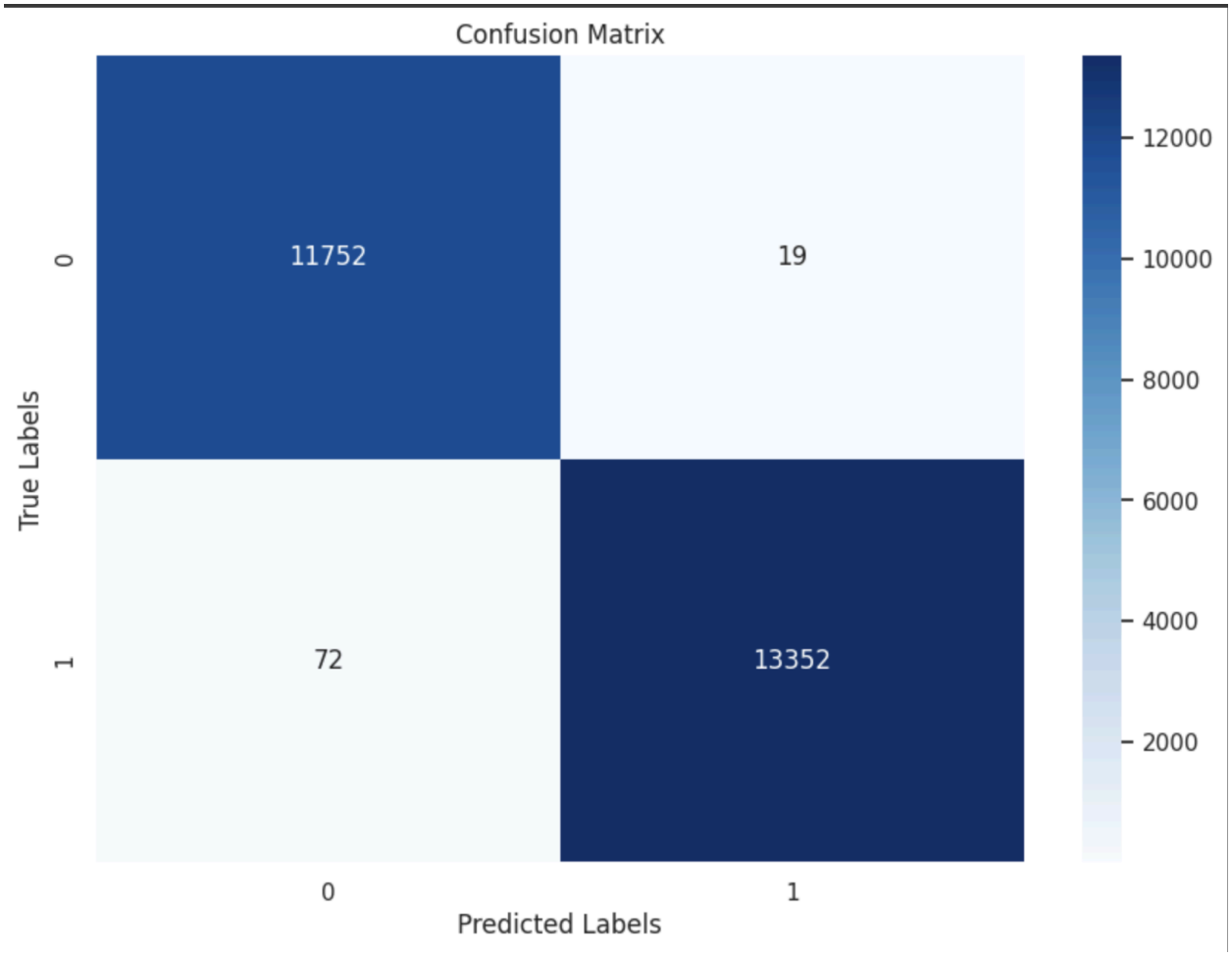
Aeden Jameson
EEP-596 Homework 2

4c.

```
Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00     11771
           1       1.00      0.99      1.00     13424

    accuracy                           1.00     25195
   macro avg       1.00      1.00      1.00     25195
weighted avg       1.00      1.00      1.00     25195
```

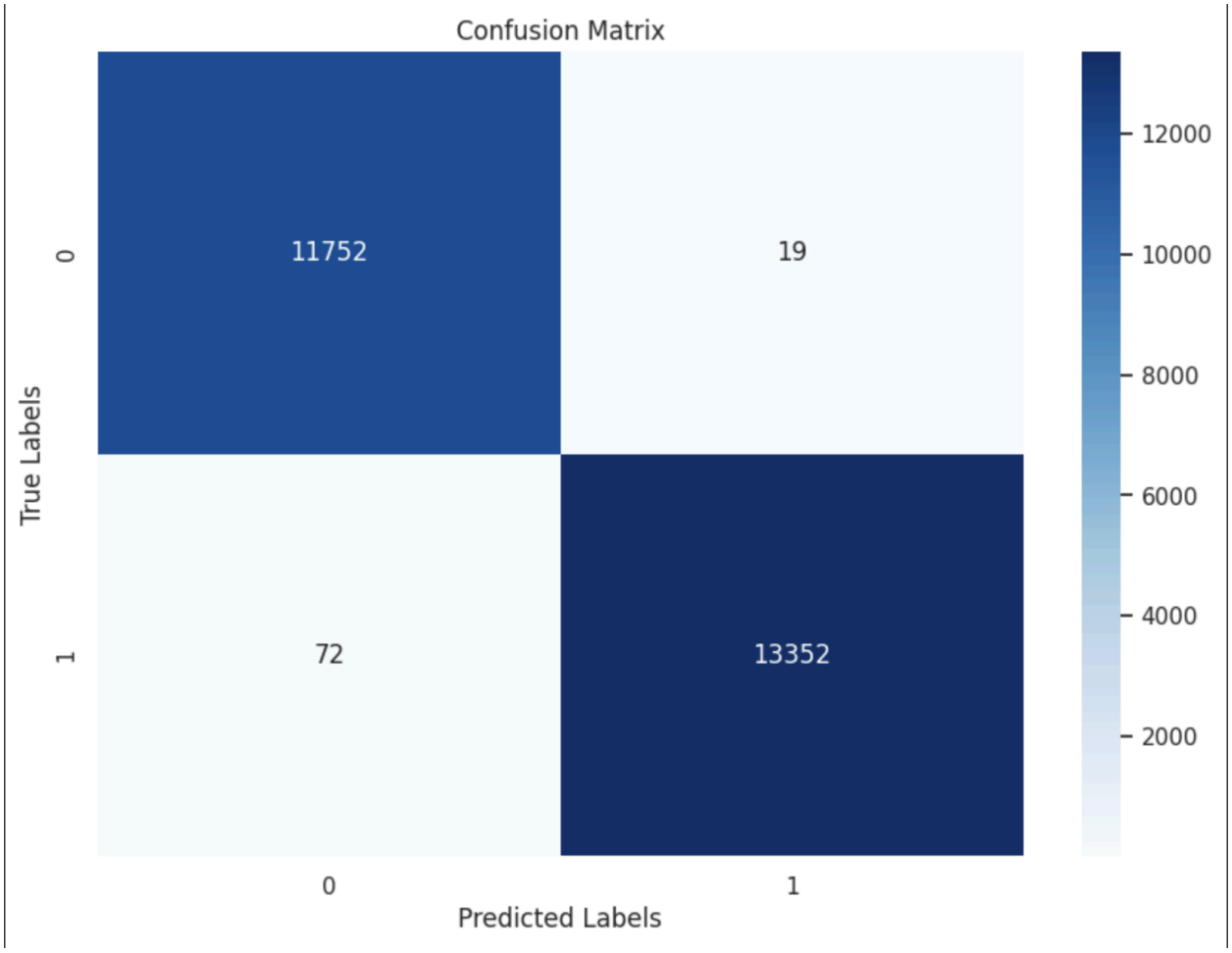


5b.

```
TFLite Classification Report:
              precision    recall  f1-score   support

           0       0.99      1.00      1.00     11771
           1       1.00      0.99      1.00     13424

    accuracy                           1.00     25195
   macro avg       1.00      1.00      1.00     25195
weighted avg       1.00      1.00      1.00     25195
```



6c.

```
Output    Serial Monitor ×
Message (Enter to send message to 'Arduino Nano 33 BLE' on '/dev/cu.usbmodem14101')
Sample #0, Predicted Class: normal, Actual Class: normal
Sample #1, Predicted Class: normal, Actual Class: normal
Sample #2, Predicted Class: attack, Actual Class: attack
Sample #3, Predicted Class: attack, Actual Class: attack
Sample #4, Predicted Class: normal, Actual Class: normal
```

6d.

```
Output    Serial Monitor ×
Message (Enter to send message to 'Arduino Nano 33 BLE' on '/dev/cu.usbmodem14101')
Sample #0, Predicted Class: normal, Actual Class: normal
Sample #1, Predicted Class: attack, Actual Class: attack
Sample #2, Predicted Class: normal, Actual Class: normal
Sample #3, Predicted Class: normal, Actual Class: normal
Sample #4, Predicted Class: normal, Actual Class: normal
Sample #5, Predicted Class: normal, Actual Class: normal
Sample #6, Predicted Class: attack, Actual Class: attack
Sample #7, Predicted Class: attack, Actual Class: attack
Sample #8, Predicted Class: attack, Actual Class: attack
Sample #9, Predicted Class: attack, Actual Class: attack
```

7a. Presuming we are using the Arduino as is then at least two plausible mechanisms for streaming network data to the Arduino Nano would be,

- Bluetooth. For more see, https://docs.arduino.cc/tutorials/nano-33-ble-sense/ble-device-to-device/
- Leverage the microphone and encode the data as sound according to an agreed upon scheme.

7b. If we have high-dimensional nonlinear network data, Kernel PCA is most suitable to use of the three options because,

- PCA assumes a linear relationship between features, which makes it inapplicable given the assumption of nonlinear data.
- t-SNE is typically used for exploratory analysis and cannot be directly used with new data because t-SNE is a non-parametric method. Thus there isn't a function that maps data from a new input to the t-sne map.
- Kernel PCA is purpose built for handling non-linear data and entails a map function for reducing new data based off of the trained transformer. Assuming one exports the KernelPCA model to Arduino. The new parameter that the Arduino program would need to be aware of is the dimension of the reduced vector that is the output of something like KernelPCA.transform().