How NOT to Create an Um Detector Using Create ML

Yono Mittlefehldt — @yonomitt — 2022

360|iDev 2022

Live Demo

Live Demo

Listening to live audio

Running a Core ML model

Analyzing for “um”s

Making predictions every 0.5s

Live Demo

Listening to live audio

Running a Core ML model

Analyzing for “um”s

Making predictions every 0.5s

…

Counting my shame

MY EYES! THE GOGGLES DO NOTHING!

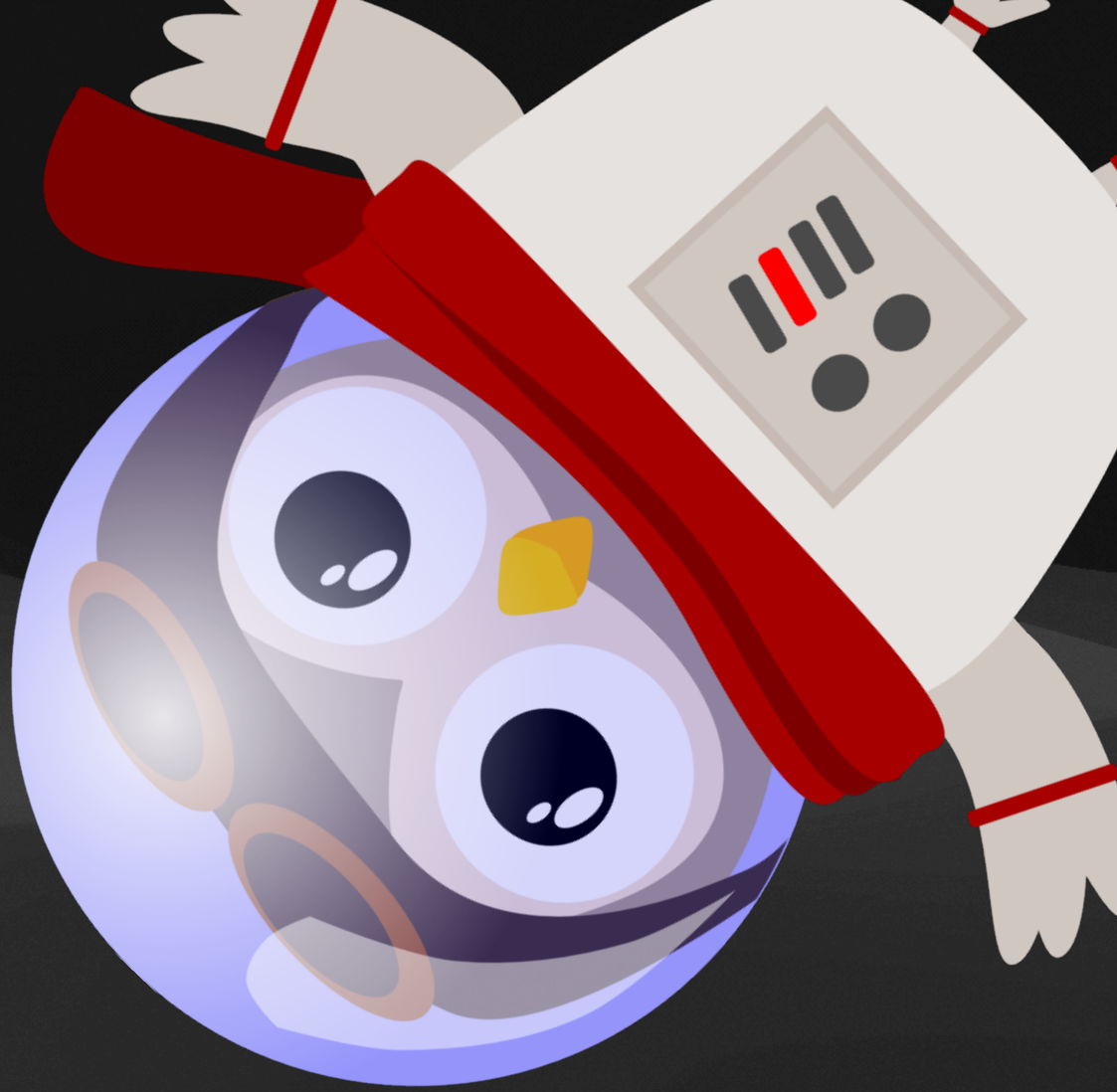
Who is Yono?

Co-creator of Gus on the Go

Who is Yono?

Co-creator of Gus on the Go

Freelance software developer
specializing in Computer Vision and iOS



Who is Yono?

Co-creator of Gus on the Go

Freelance software developer specializing in Computer Vision and iOS

Tutorial writer for raywenderlich.com

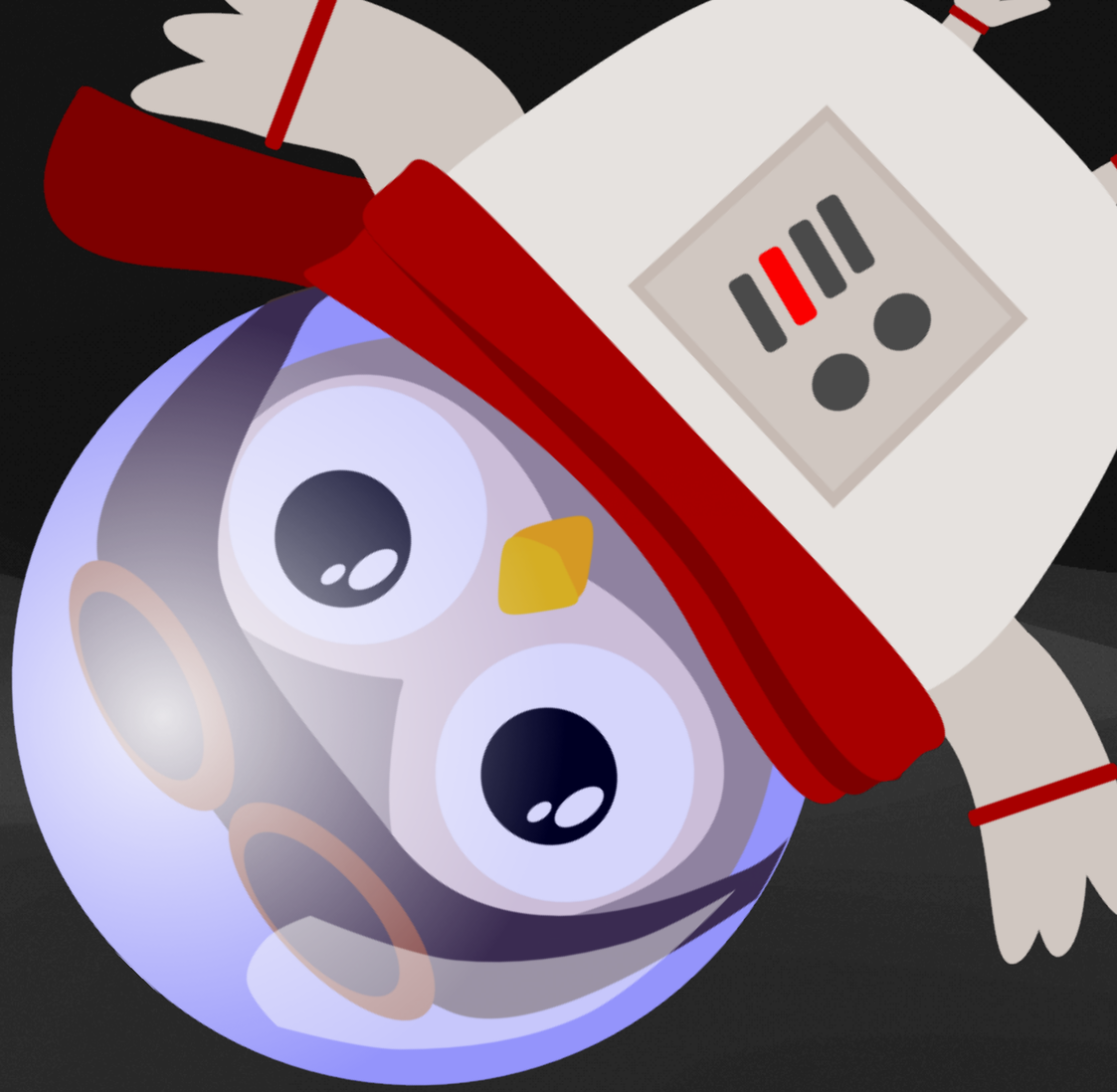
Who is Yono?

Co-creator of Gus on the Go

Freelance software developer specializing in Computer Vision and iOS

Tutorial writer for raywenderlich.com

A neighbor once called the police on me for running a meth lab out of my apartment



Story Time

2

2

2

2

2

2

2

2

2

2

2

2

2

2

Why would you want to do this?

360|iDev
2021

Seriously, why would you want to do this?

Who needs this?

Podcasters

Audio editors

Public speaking trainers

Who needs this?

Podcasters

Audio editors

Public speaking trainers

…

YouTubers?

This talk isn’t about code

It’s really about a thought process

How do we detect ums?

Attempt #1 - Speech Recognition

Speech Recognition

Apple’s Speech Framework

It’s built in

It’s easy to use

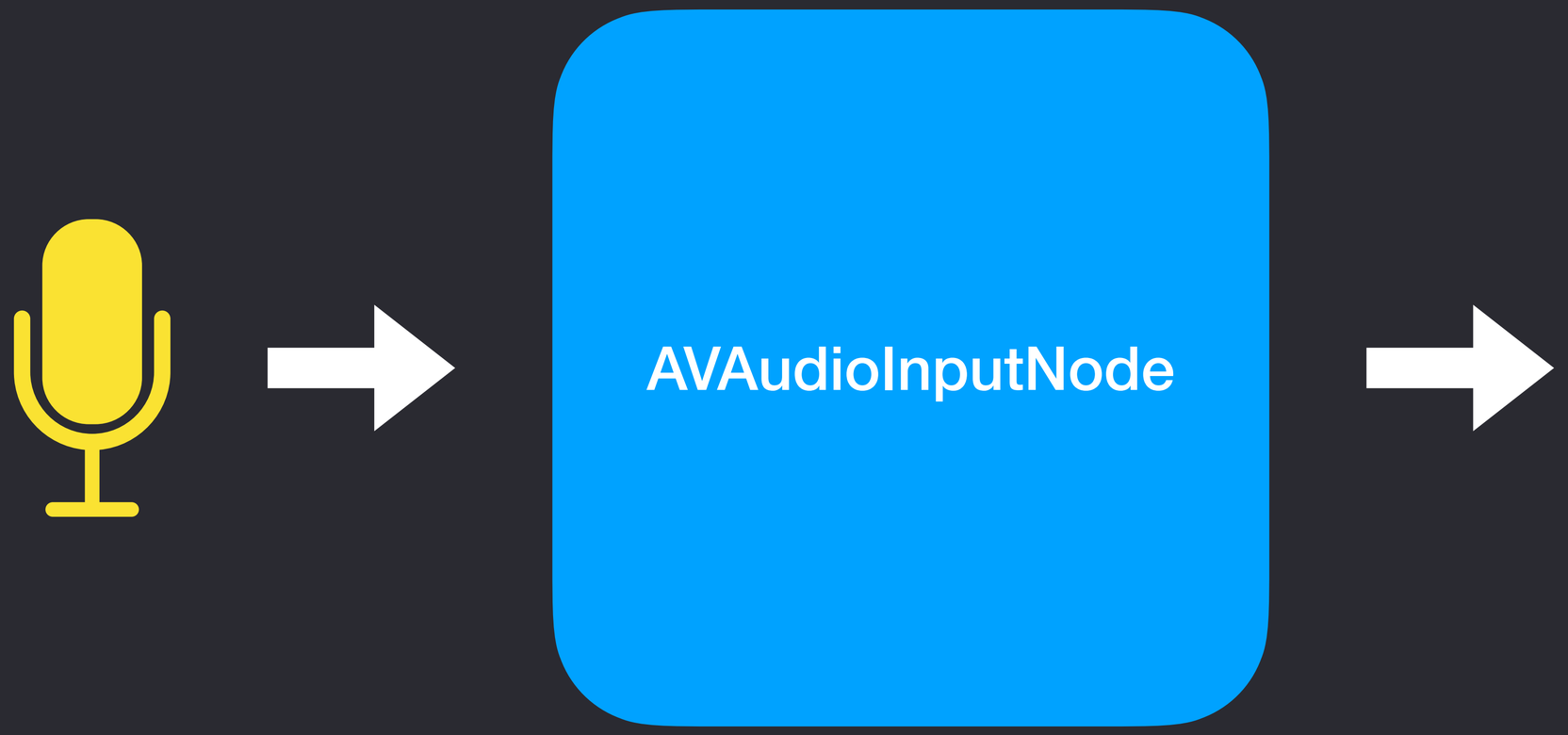
```swift
let audioEngine = AVAudioEngine()
```

```swift
// Configure the microphone input
let audioEngine = AVAudioEngine()
let inputNode = audioEngine.inputNode
```

```swift
// Configure the microphone input
let audioEngine = AVAudioEngine()
let inputNode = audioEngine.inputNode

// Get the format of the data
let format = inputNode.outputFormat(forBus: 0)
```

AVAudioInputNode

AVAudioInputNode


AVAudioInputNode

iPhone XS Max

```swift
// Configure the microphone input
let audioEngine = AVAudioEngine()
let inputNode = audioEngine.inputNode

// Get the format of the data
let format = inputNode.outputFormat(forBus: 0)
```

```swift
// Configure the microphone input
let audioEngine = AVAudioEngine()
let inputNode = audioEngine.inputNode

// Get the format of the data
let format = inputNode.outputFormat(forBus: 0)

// Install an audio tap to observe the output of the input node
inputNode.installTap(onBus: 0, bufferSize: 1024, format: format) { (buffer, _) in
    request.append(buffer)
}
```

```swift
// Create the Speech request
let request = SFSpeechAudioBufferRecognitionRequest()

// Configure the microphone input
let audioEngine = AVAudioEngine()
let inputNode = audioEngine.inputNode

// Get the format of the data
let format = inputNode.outputFormat(forBus: 0)

// Install an audio tap to observe the output of the input node
inputNode.installTap(onBus: 0, bufferSize: 1024, format: format) { (buffer, _) in
    request.append(buffer)
}
```

But it doesn’t work

How not to create
an arm detector
using create ML

How not to create
an M detector

Hacky Hack

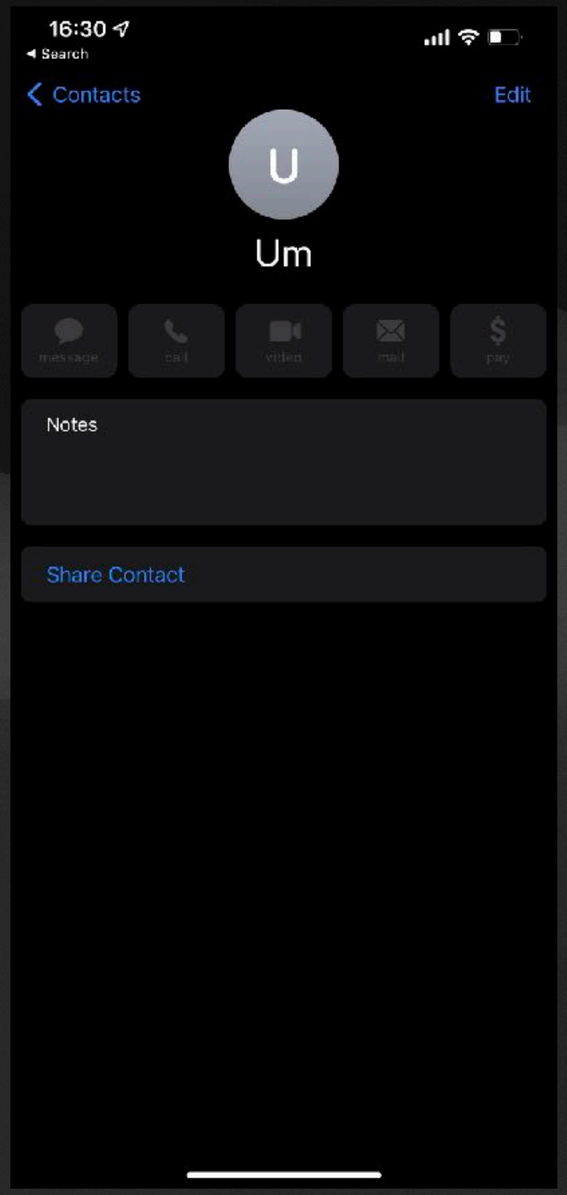
Um Contact

Allows dictation to detect “um”s

Sorta…

Only detected if “um” could be replaced by a name

“Uh” didn’t work



How do we detect ums?

Attempt #2 - Machine Learning

“Machine Learning is the science of getting computers to learn without being explicitly programmed.”

Sebastian Thrun

X
Y
A
B

Chat Bot User Interface
Temp Name
Temp Data

Machine Learning

What do we need?

Lots of math

Dataset

Train a model

Testing

Machine Learning

What do we need?

~~Lots of math~~

Dataset

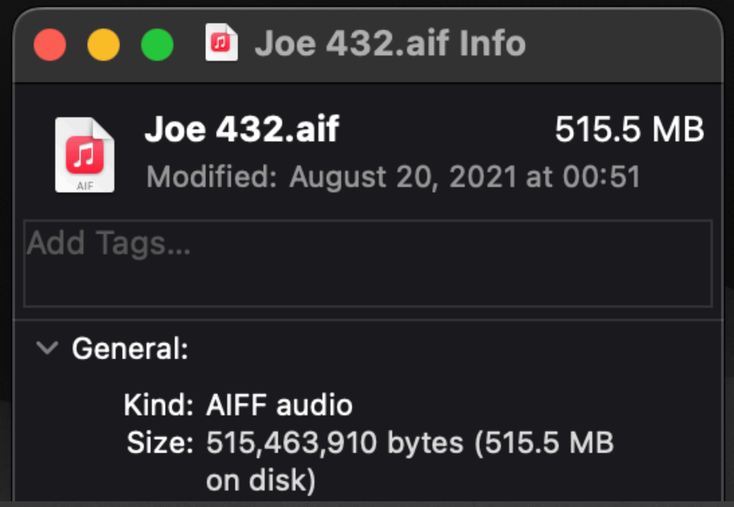
Train a model

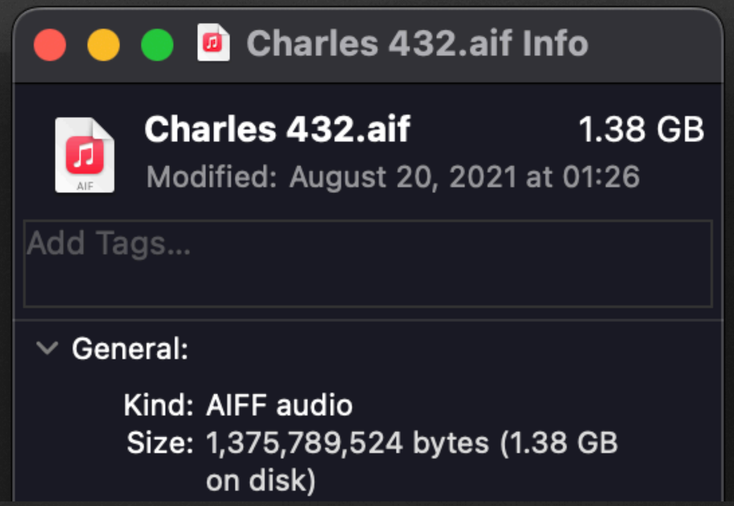
Testing

Dataset

Possibly the most important part

Raw audio files





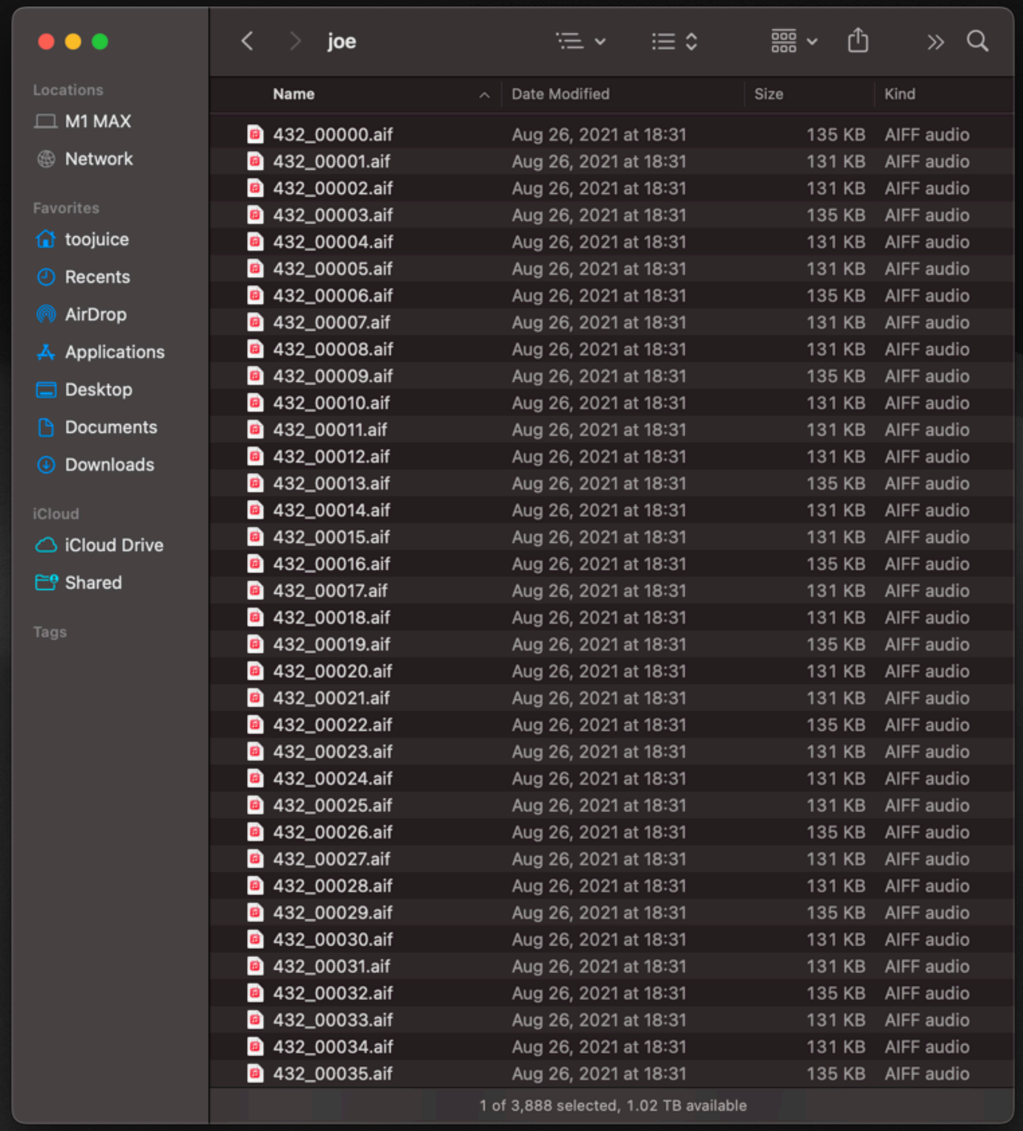
* Charles recorded 2 channel, 32 bit audio, Joe recorded 1 channel, 24 bit

Dataset

Possibly the most important part

Raw audio files

Cut them into 1 second segments



>

```
> ffmpeg
```

```
> ffmpeg -i Joe\ 432.aif
```

```
> ffmpeg -i Joe\ 432.aif -f segment
```

```
> ffmpeg -i Joe\ 432.aif -f segment -segment_time 1
```

```
> ffmpeg -i Joe\ 432.aif -f segment -segment_time 1 -c copy
```

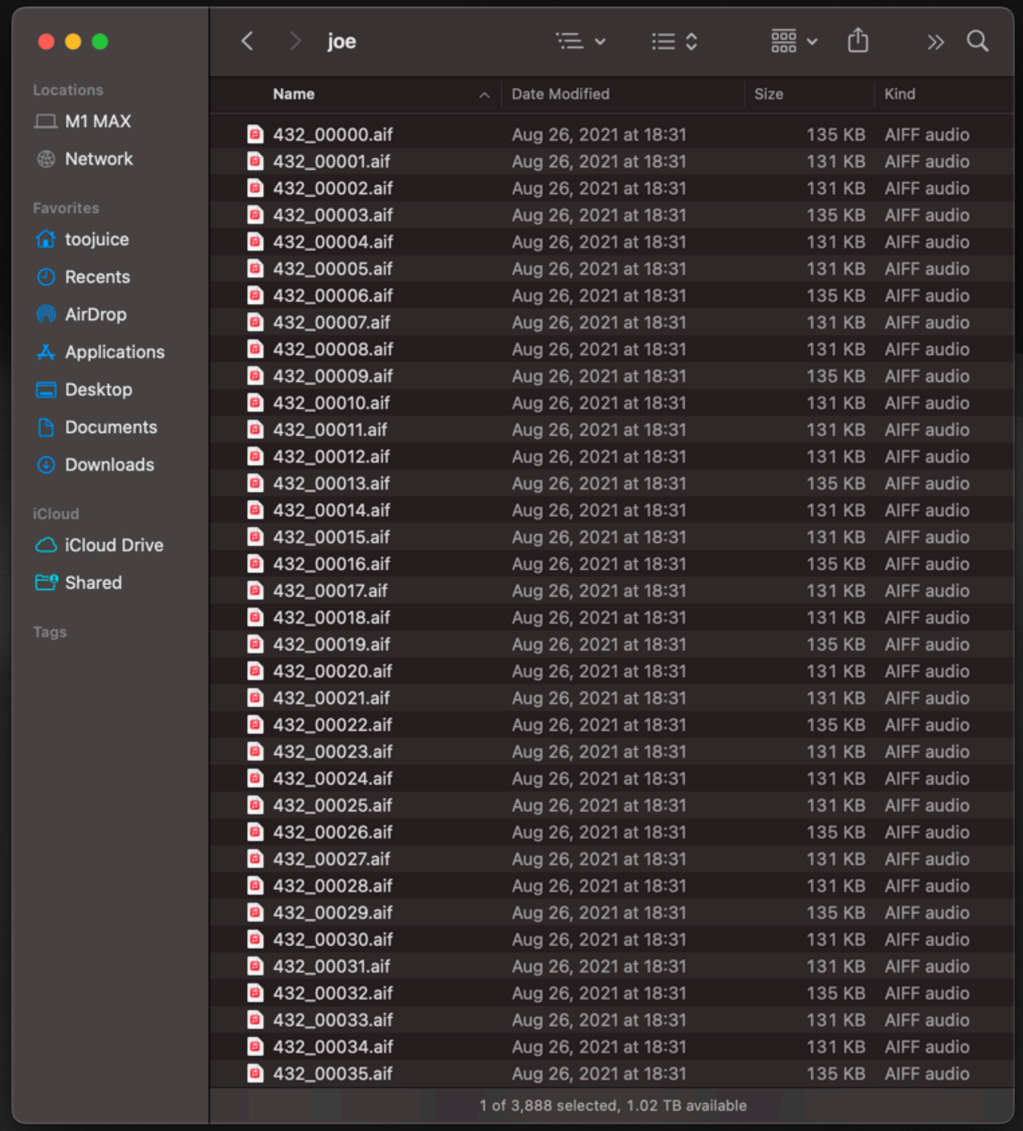
```
> ffmpeg -i Joe\ 432.aif -f segment -segment_time 1 -c copy joe/432_%05d.aif
```

Dataset

Possibly the most important part

Raw audio files

Cut them into 1 second segments



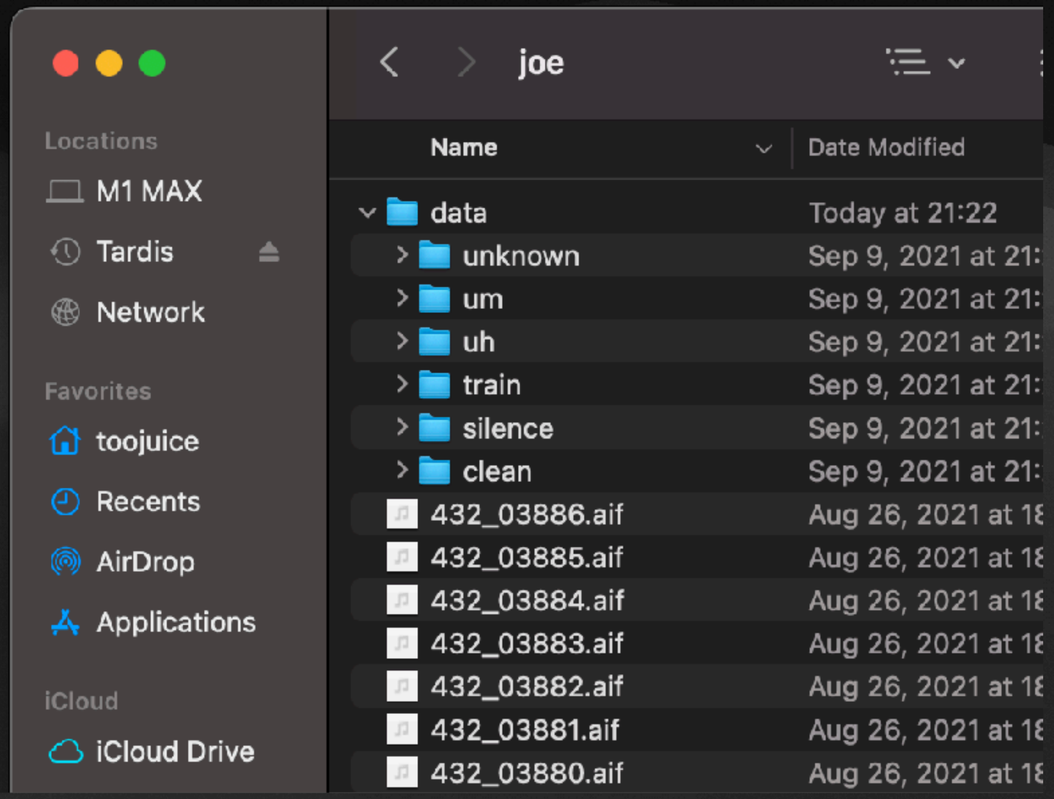
Dataset

Possibly the most important part

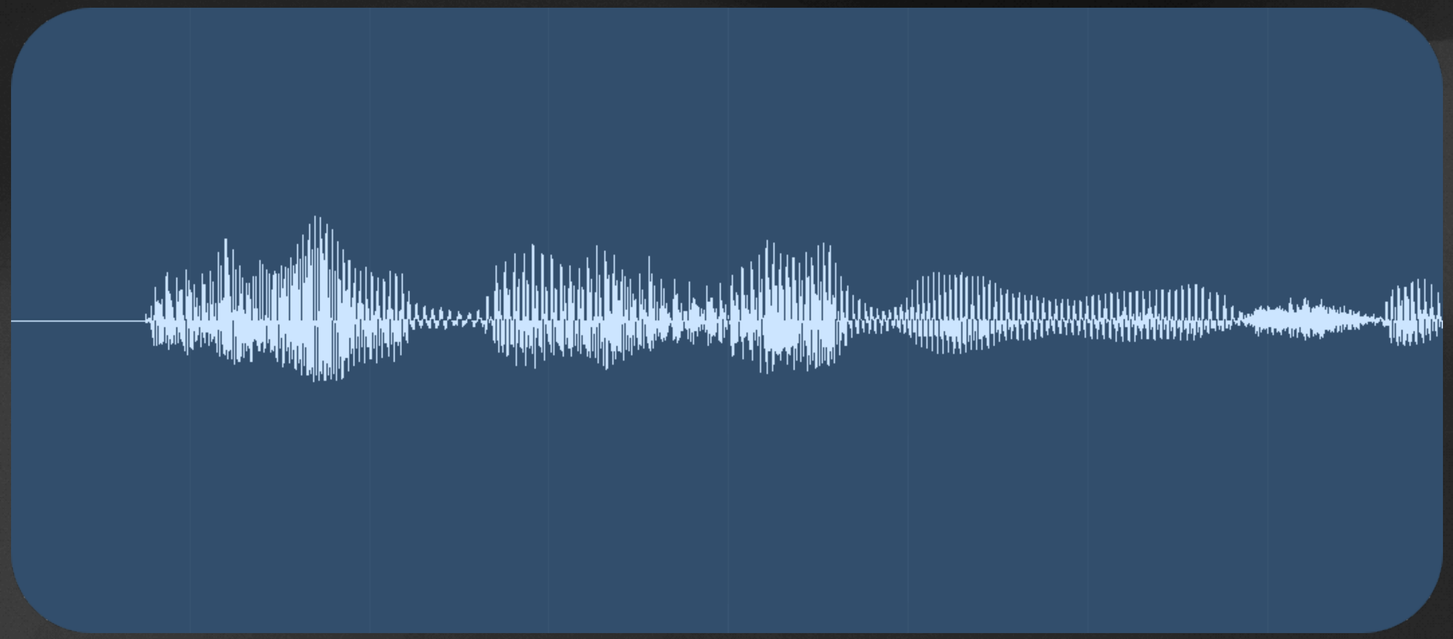
Raw audio files

Cut them into 1 second segments

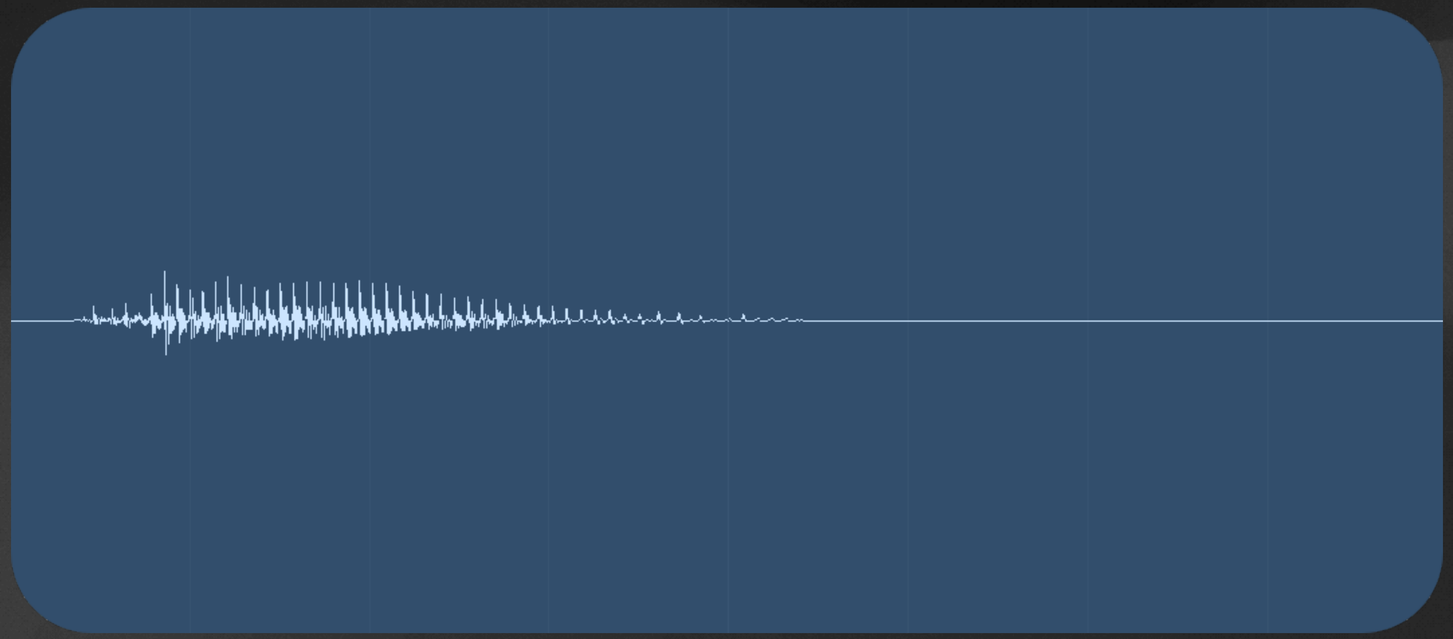
Sort them into categories



Clean



Um



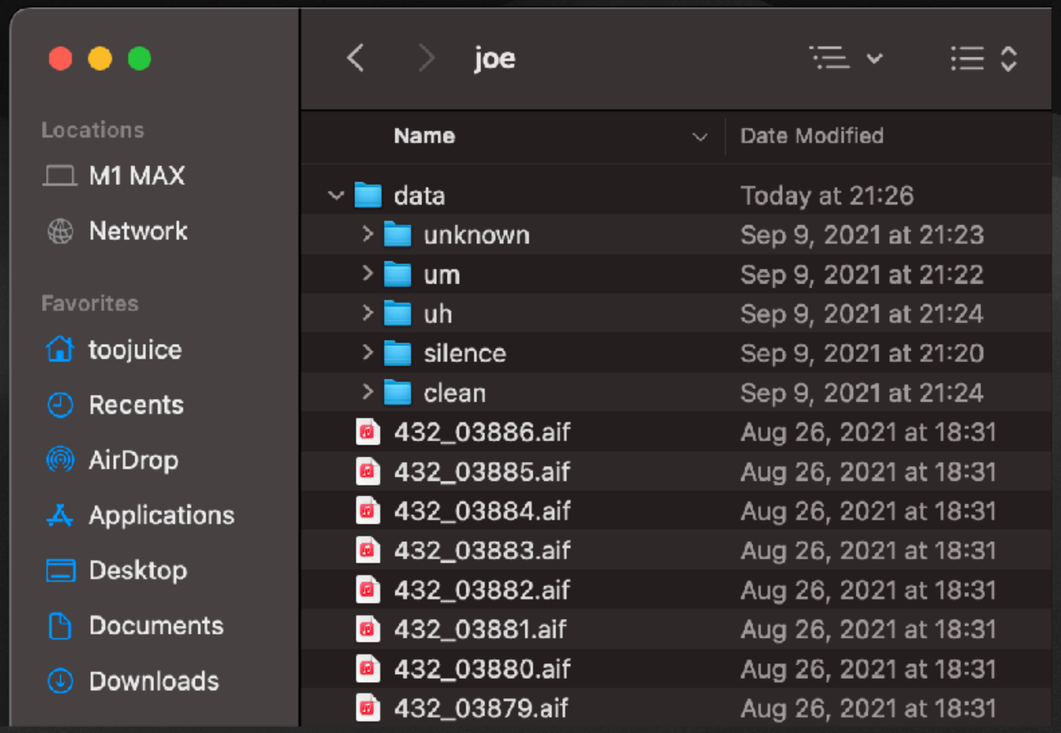
Dataset

Possibly the most important part

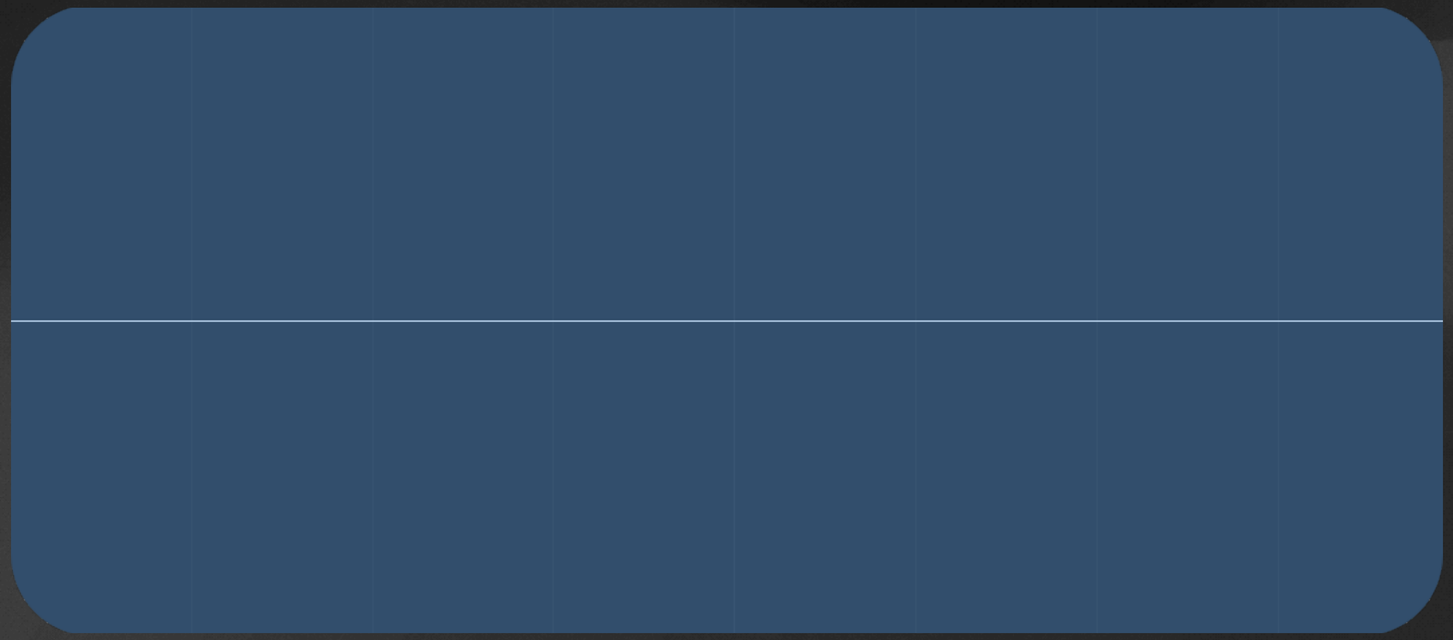
Raw audio files

Cut them into 1 second segments

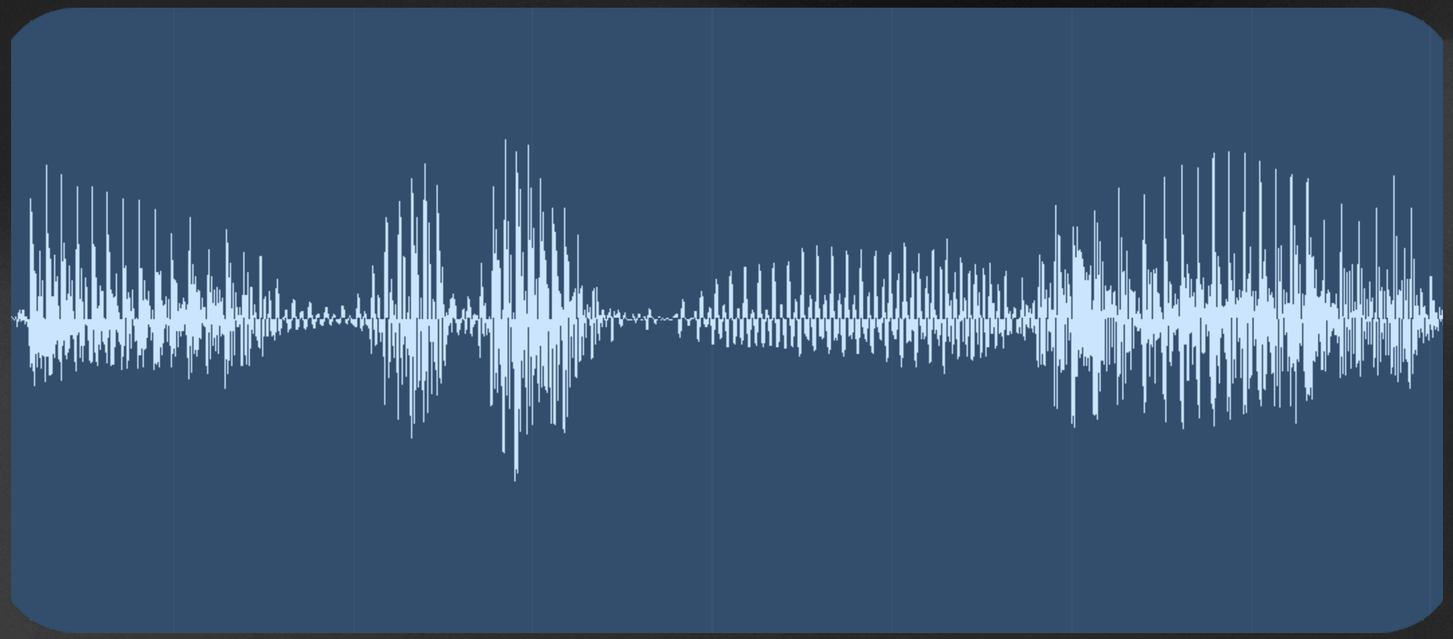
Sort them into categories



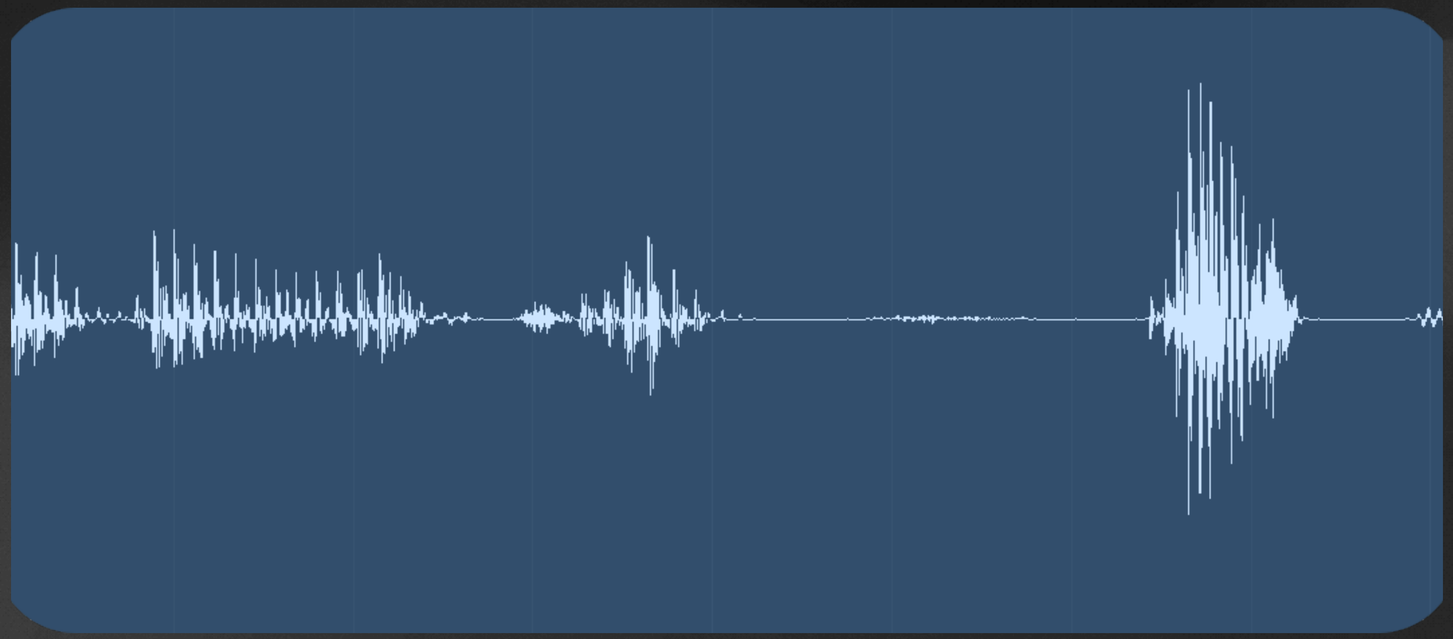
Silence



Uh



Unknown



“This seems tedious”

You, probably

Touch Bar

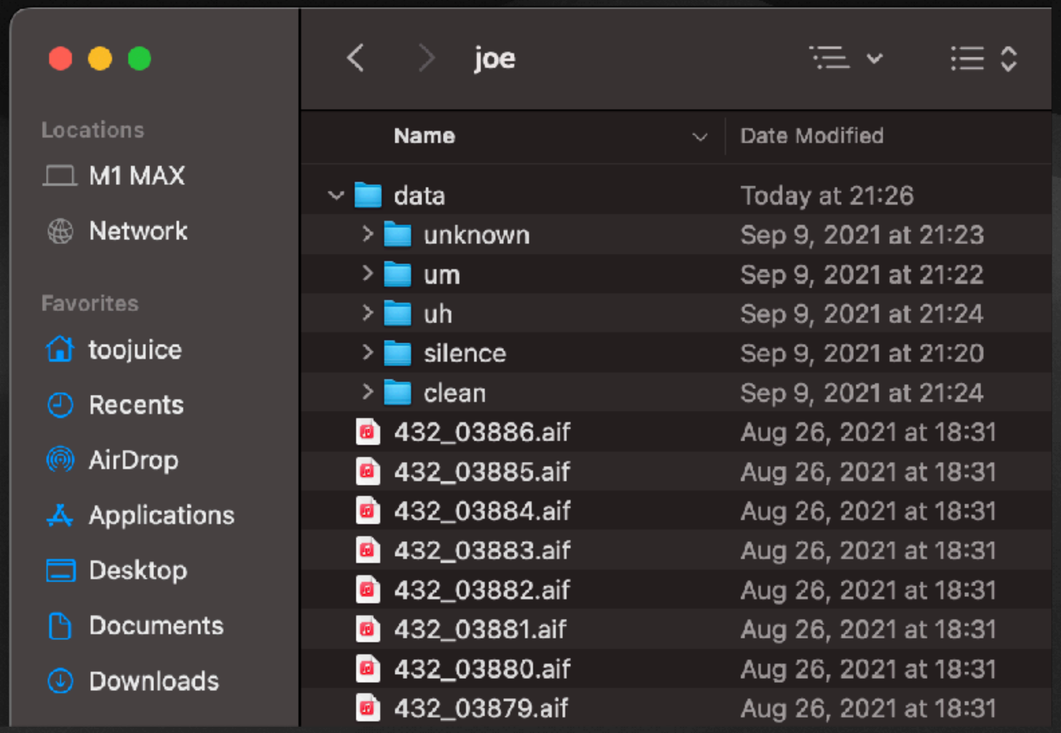
Dataset

Possibly the most important part

Raw audio files

Cut them into 1 second segments

Sort them into categories



Dataset

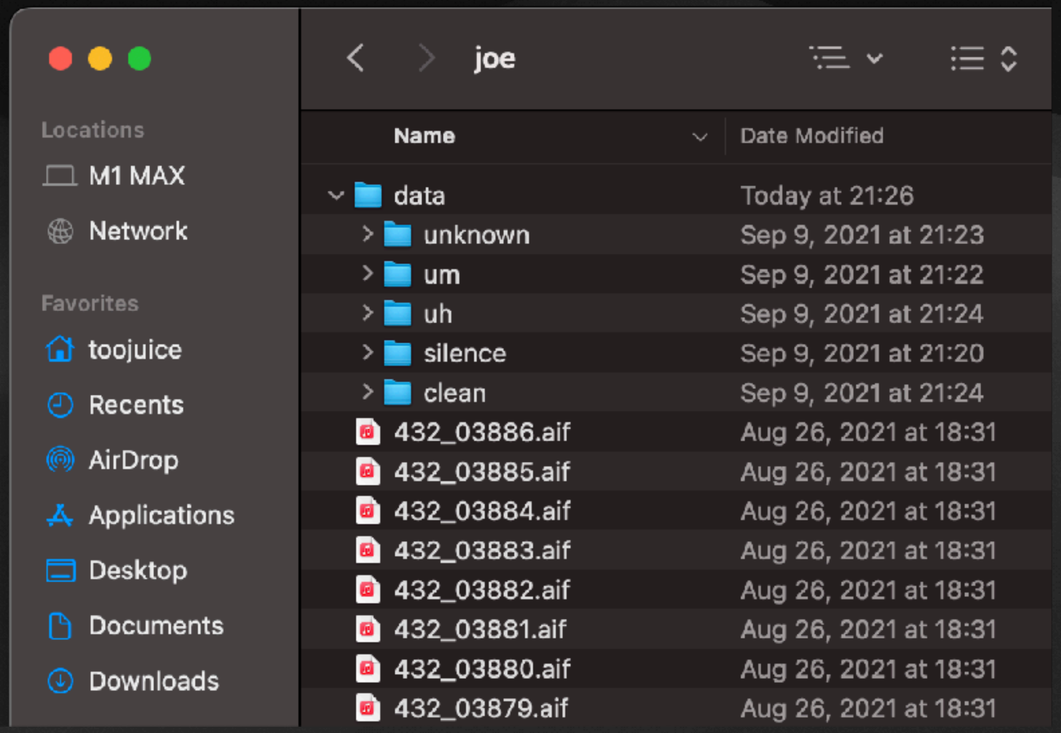
Possibly the most important part

Raw audio files

Cut them into 1 second segments

Sort them into categories

Leave some for testing



Machine Learning

What do we need?

~~Lots of math~~

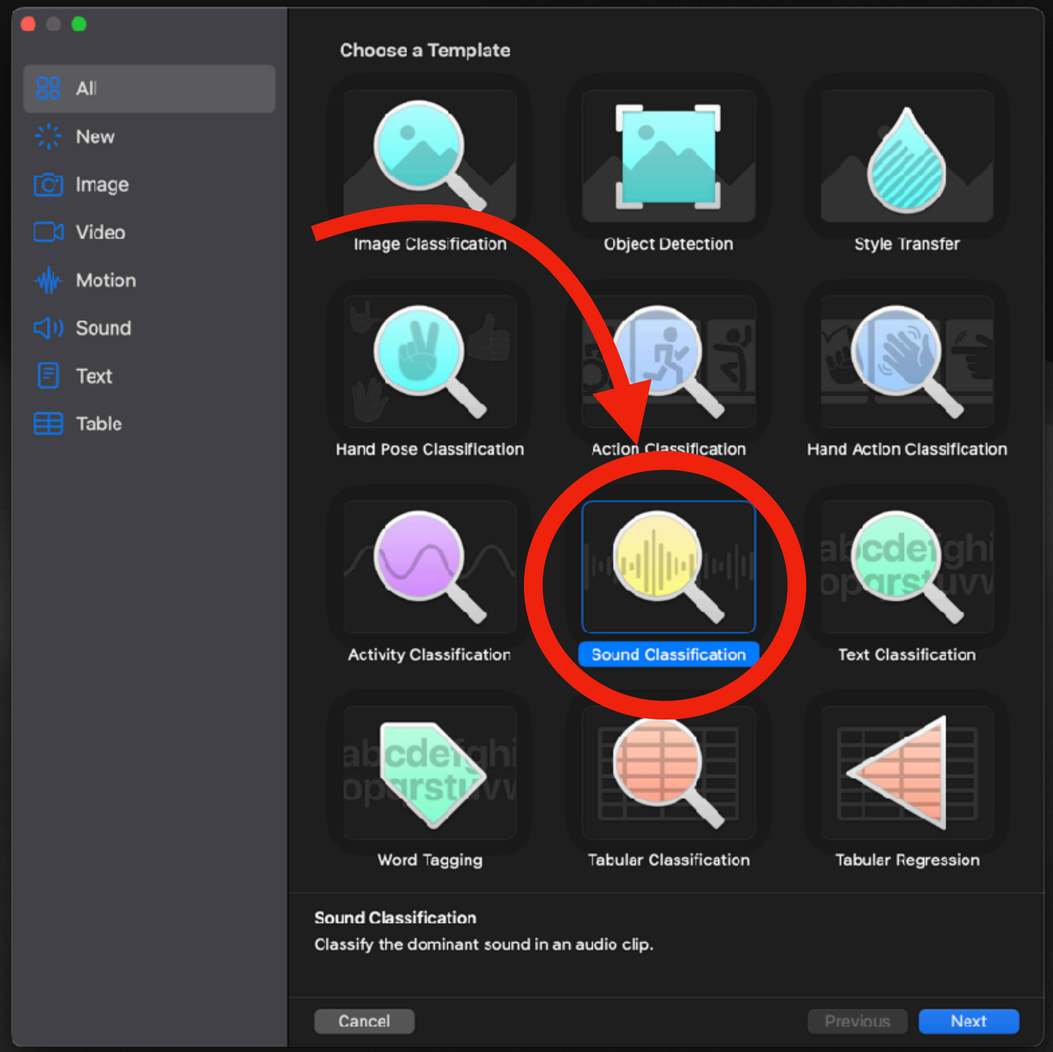
Dataset

Train a model

Testing

Create ML

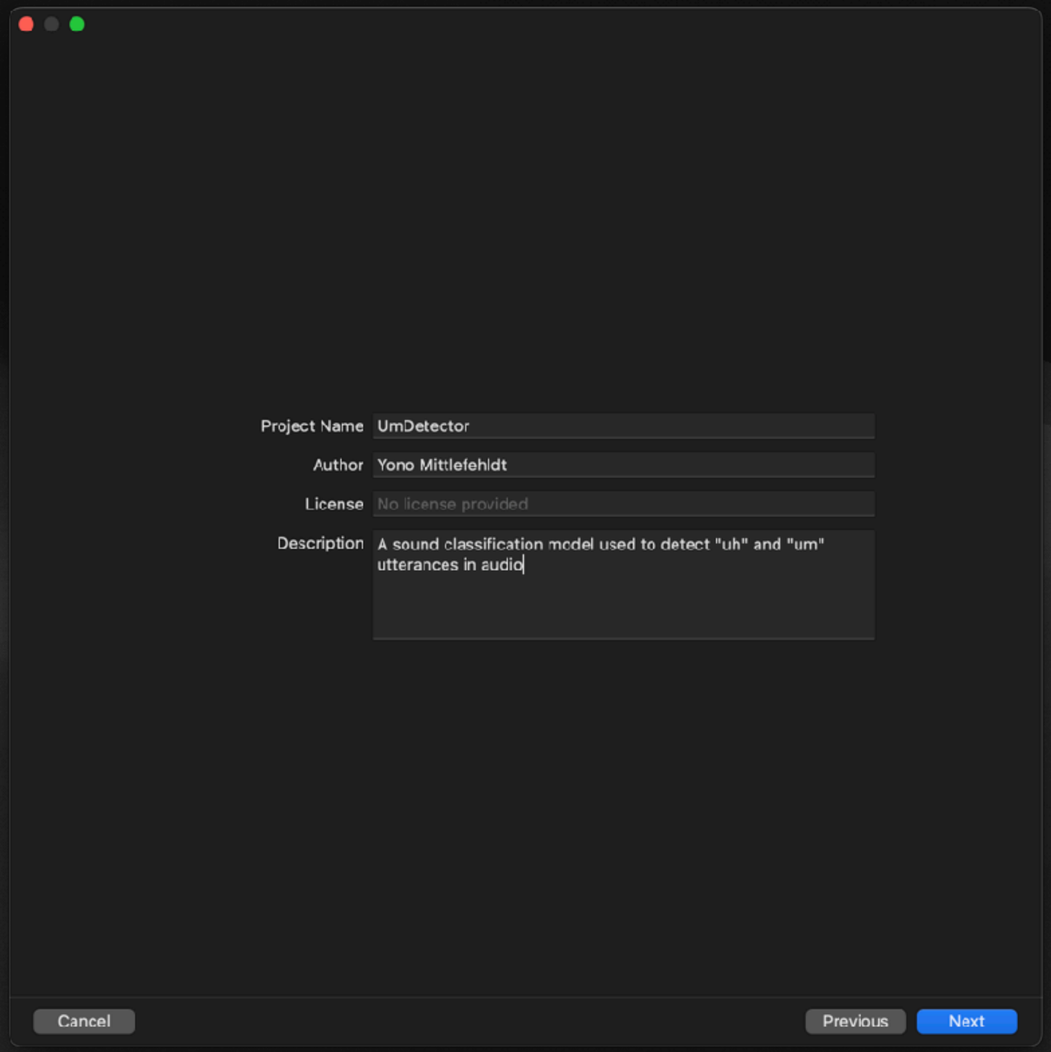
Create a sound classification project in Create ML



Create ML

Create a sound classification project in Create ML

Set project info

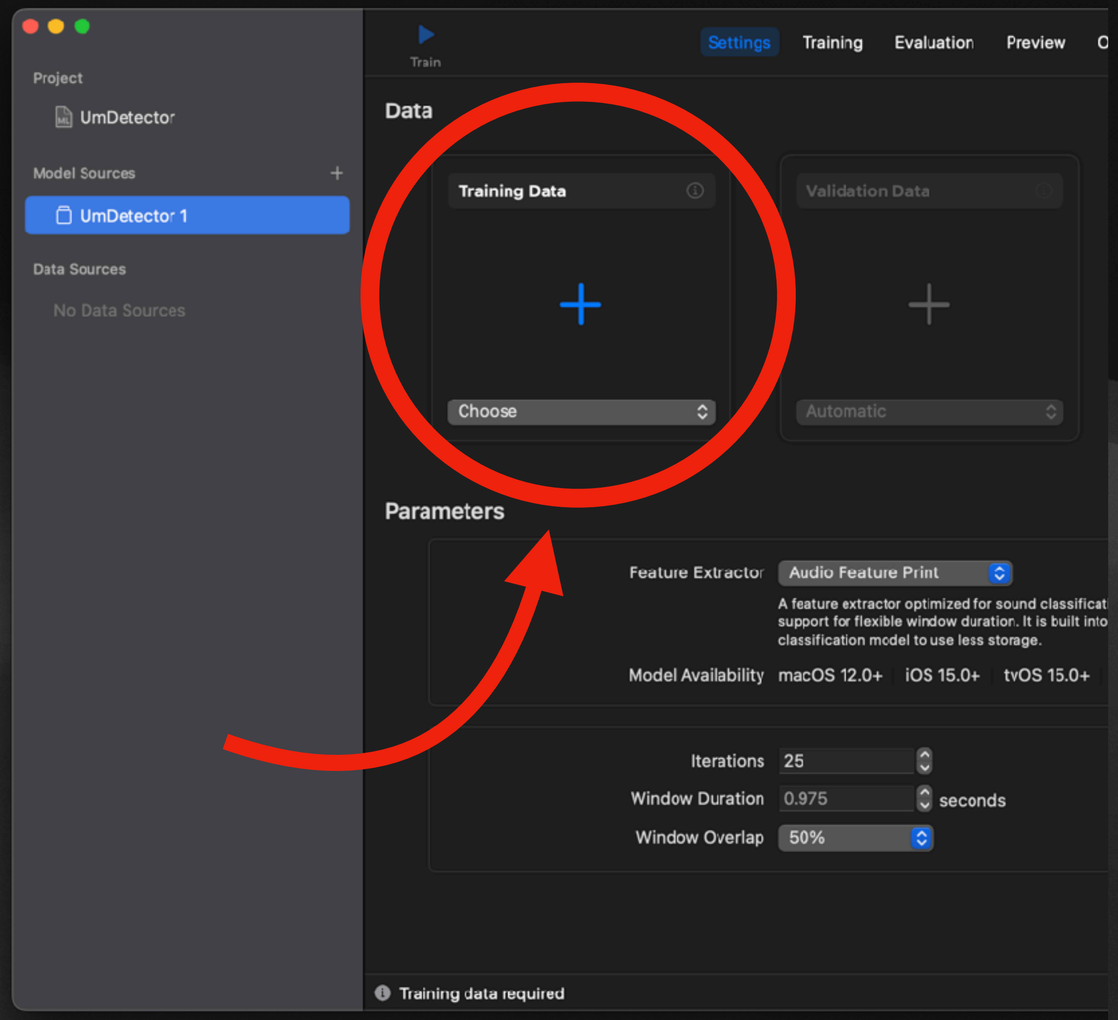


Create ML

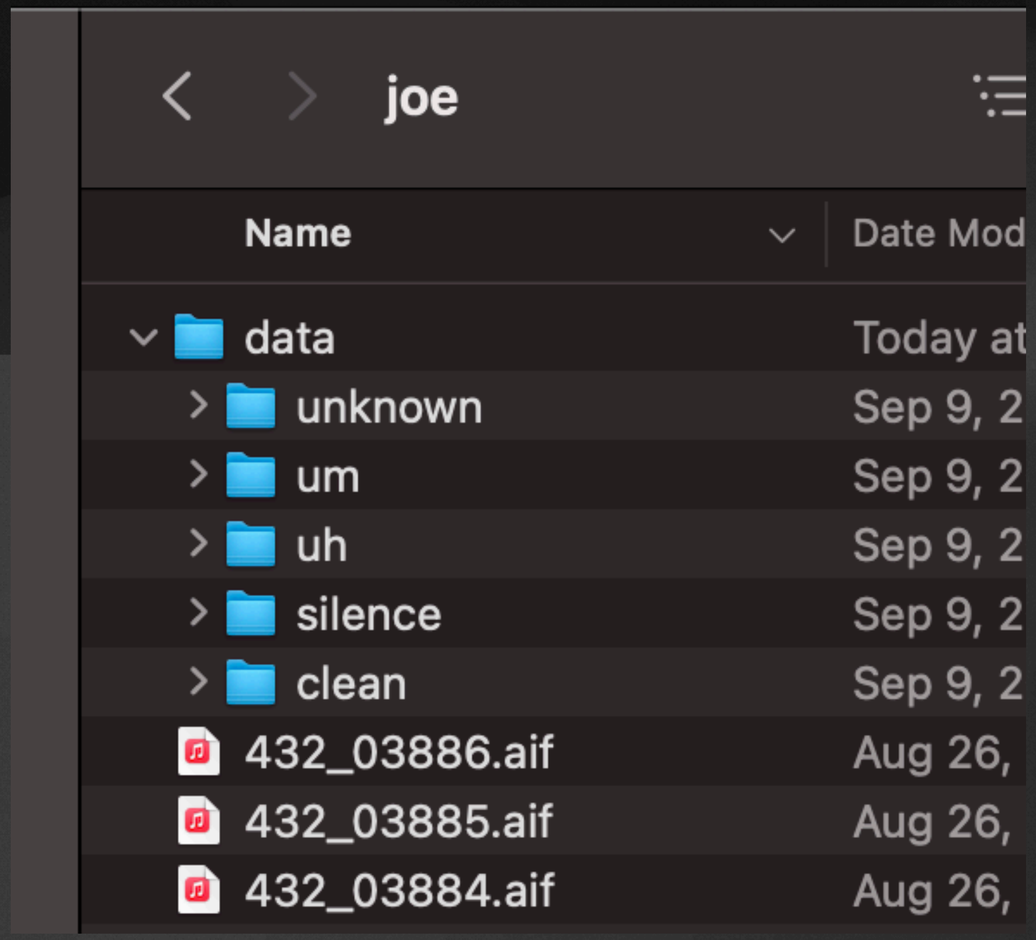
Create a sound classification project in Create ML

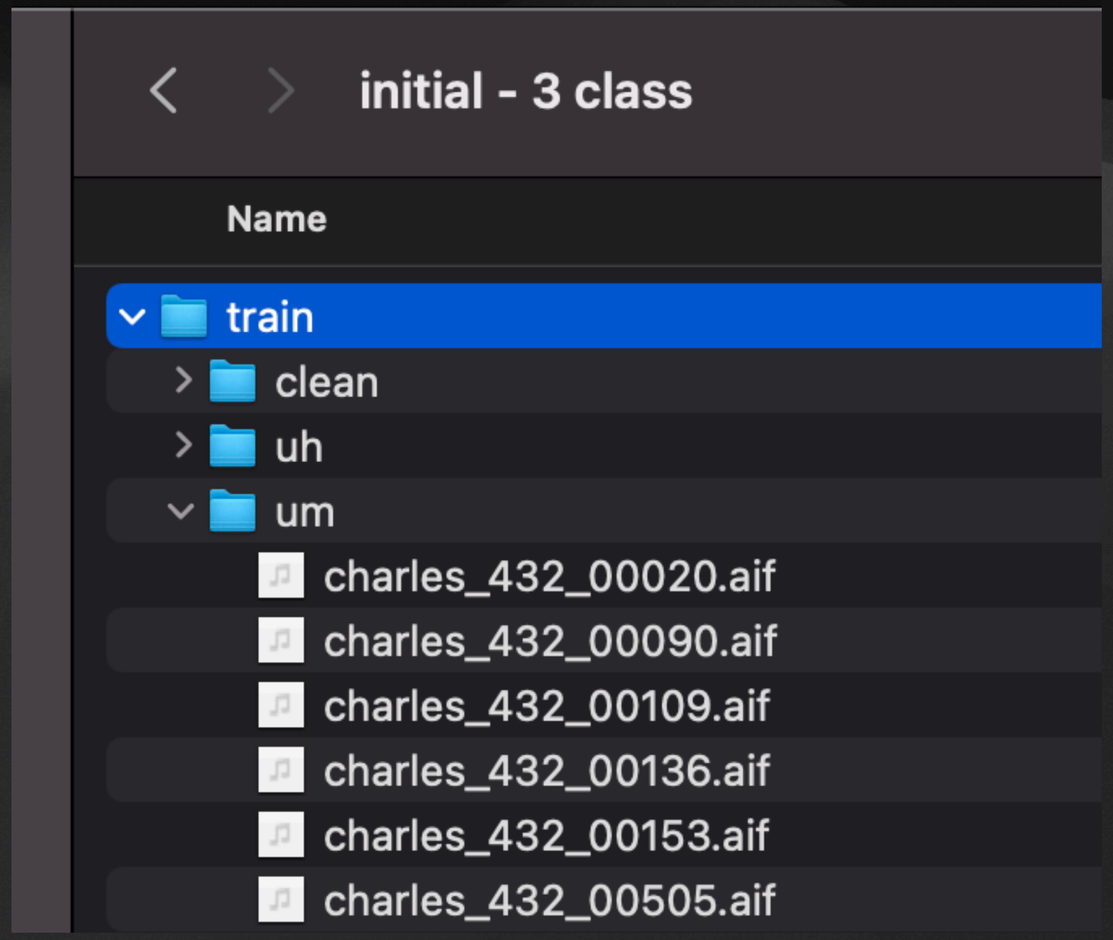
Set project info

Add the dataset



Dataset


joe
Name
Date Mod
data
Today at
unknown
Sep 9, 2
um
Sep 9, 2
uh
Sep 9, 2
silence
Sep 9, 2
clean
Sep 9, 2
432_03886.aif
Aug 26,
432_03885.aif
Aug 26,
432_03884.aif
Aug 26,


initial - 3 class
Name
train
clean
uh
um
charles_432_00020.aif
charles_432_00090.aif
charles_432_00109.aif
charles_432_00136.aif
charles_432_00153.aif
charles_432_00505.aif

Two vs Three Classes

Um vs Uh

Initially only two class

- Clean
- Um

Two vs Three Classes

Um vs Uh

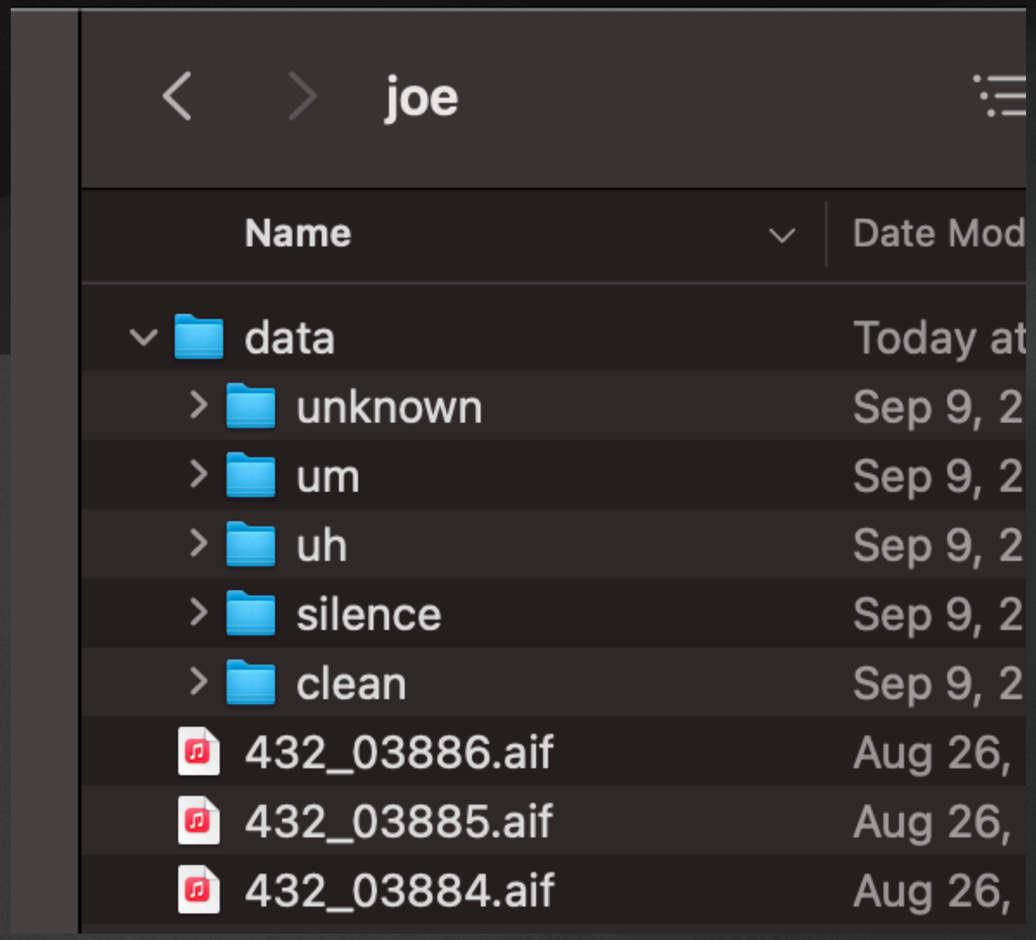
Initially only two class

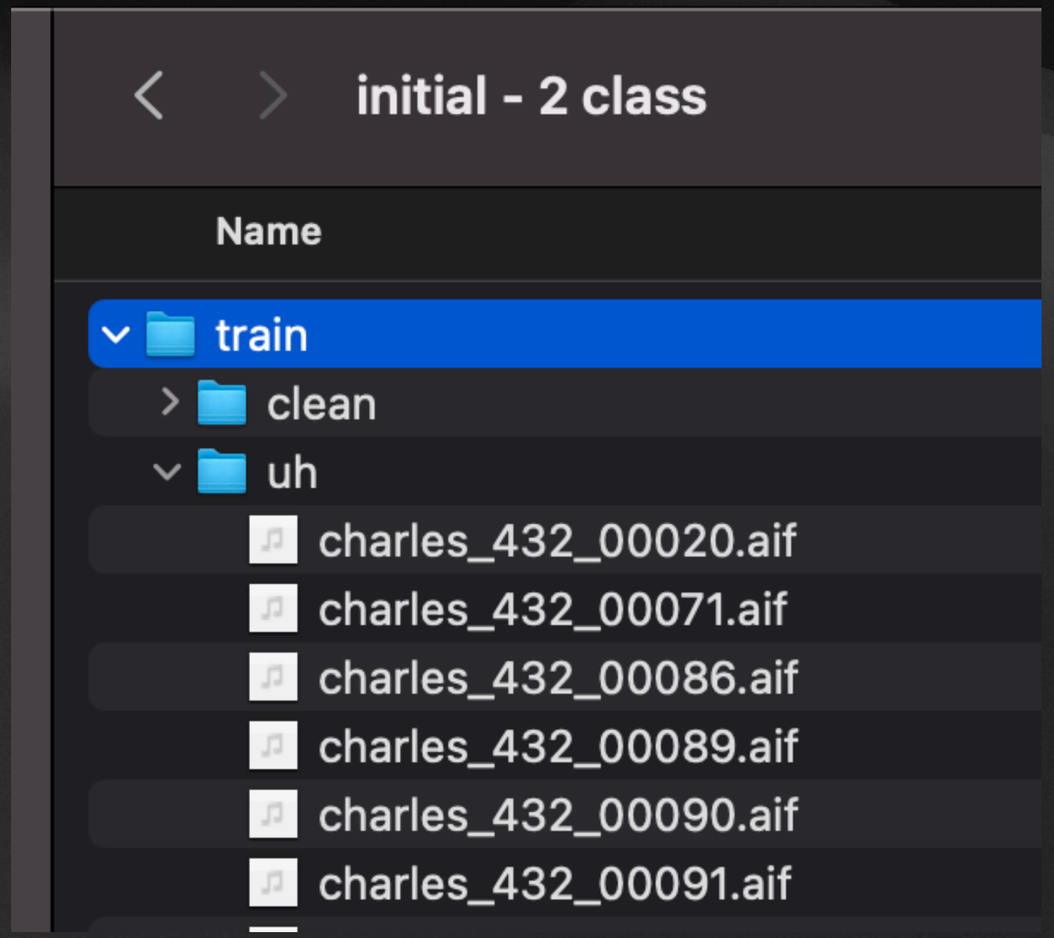
- Clean
- Um

Dataset creation unveiled 3rd class

- Uh

Dataset


joe
Name
Date Mod
data
Today at
unknown
Sep 9, 2
um
Sep 9, 2
uh
Sep 9, 2
silence
Sep 9, 2
clean
Sep 9, 2
432_03886.aif
Aug 26,
432_03885.aif
Aug 26,
432_03884.aif
Aug 26,

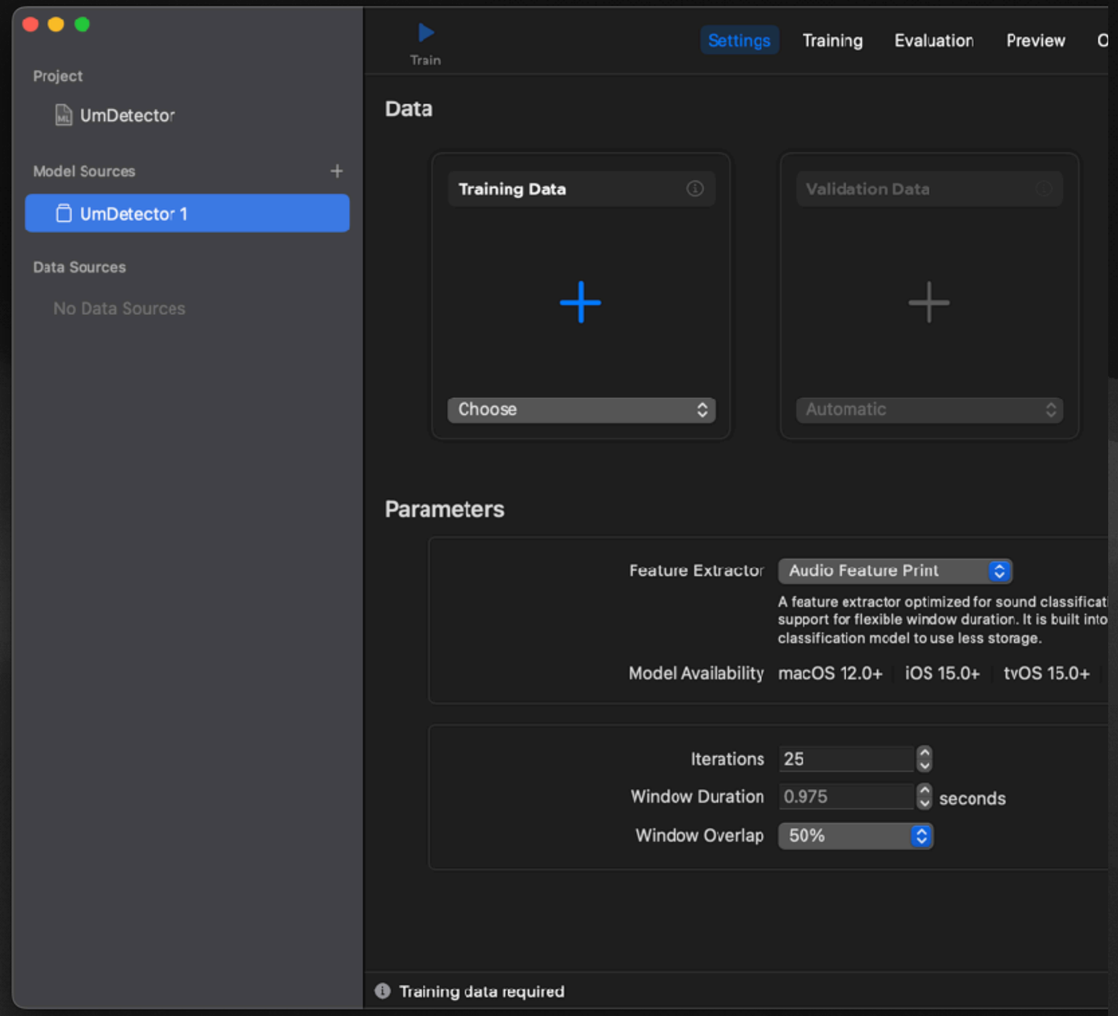

initial - 2 class
Name
train
clean
uh
charles_432_00020.aif
charles_432_00071.aif
charles_432_00086.aif
charles_432_00089.aif
charles_432_00090.aif
charles_432_00091.aif

Create ML

Create a sound classification project in Create ML

Set project info

Add the dataset

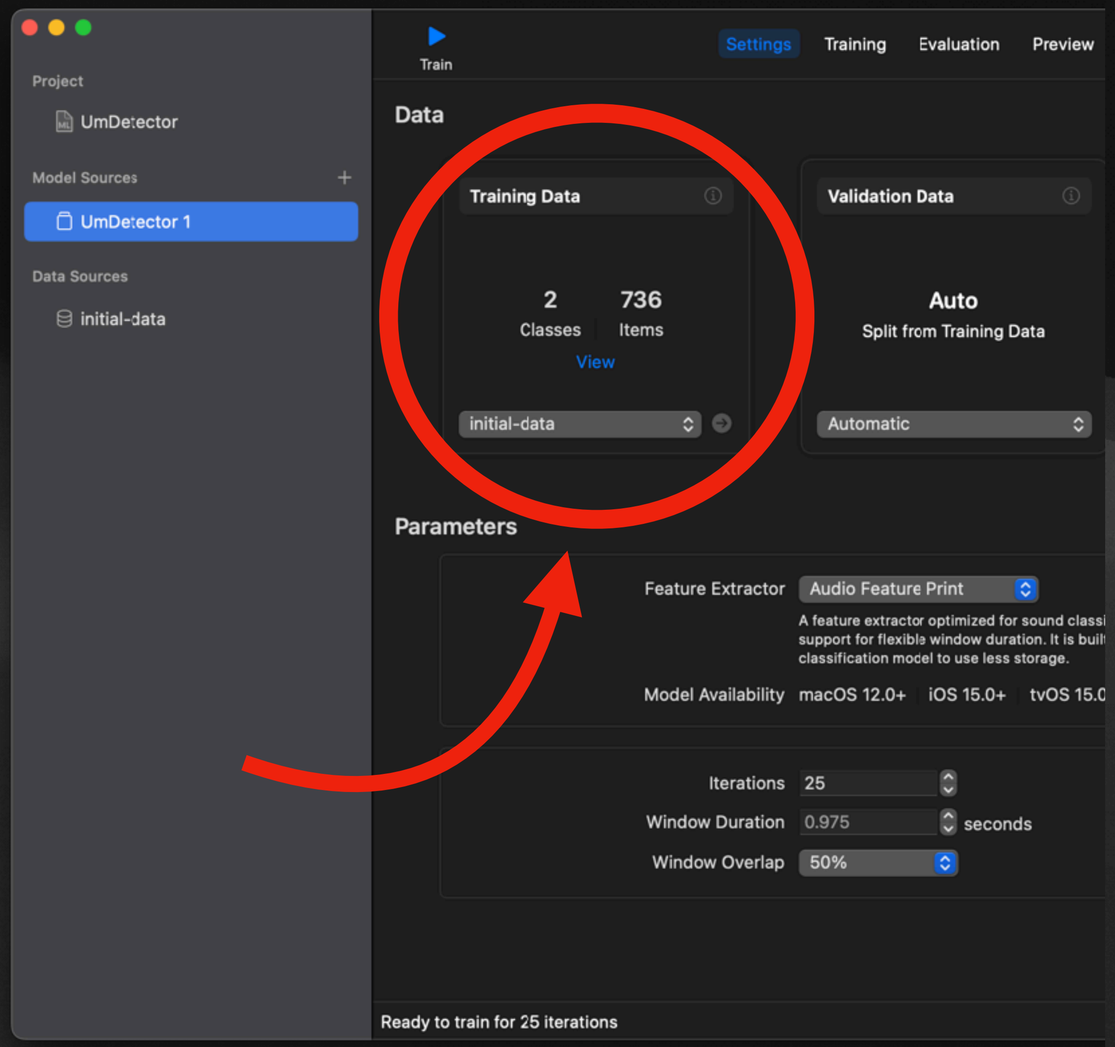


Create ML

Create a sound classification project in Create ML

Set project info

Add the dataset

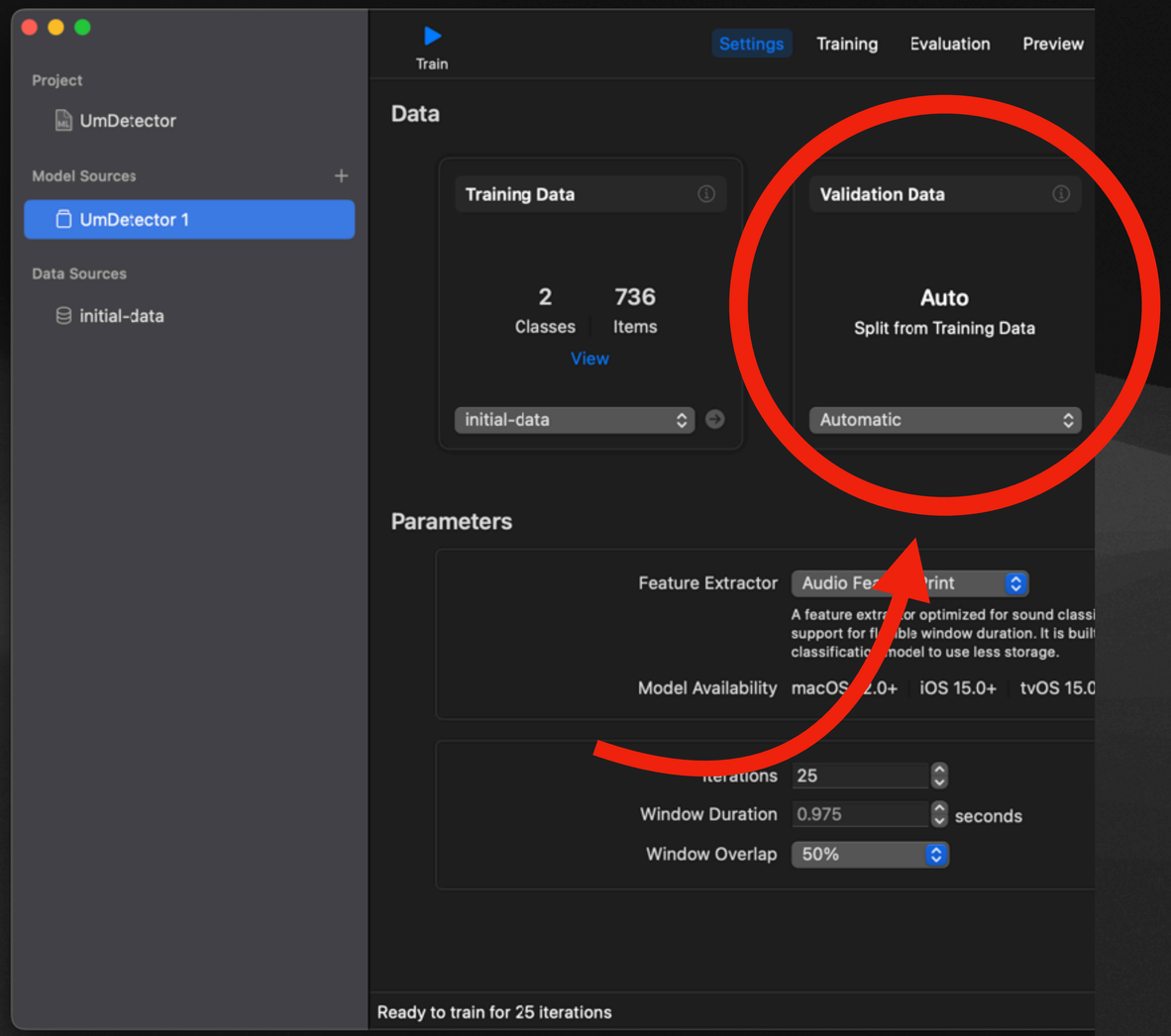


Create ML

Create a sound classification project in Create ML

Set project info

Add the dataset



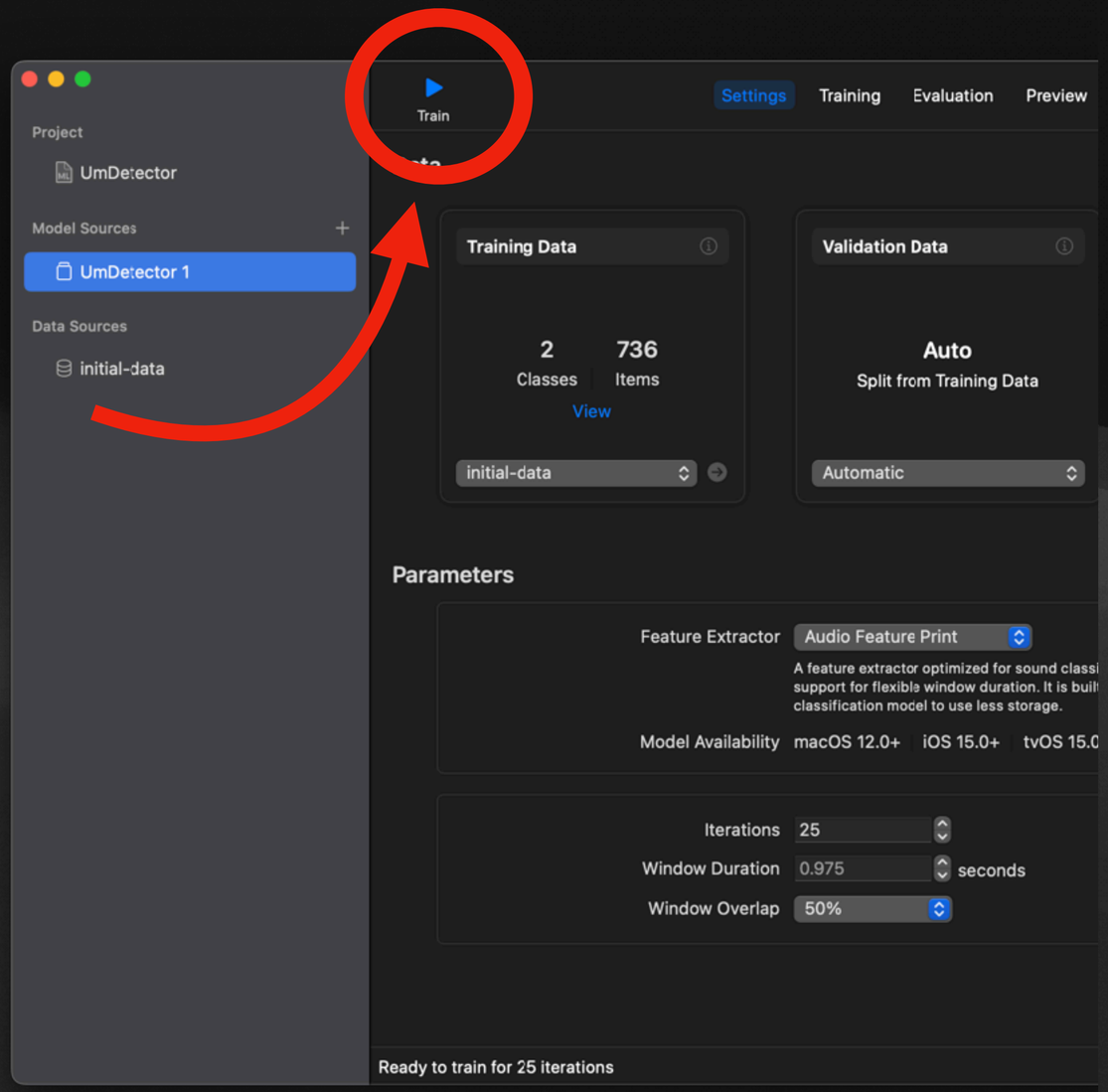
Create ML

Create a sound classification project in Create ML

Set project info

Add the dataset

Train



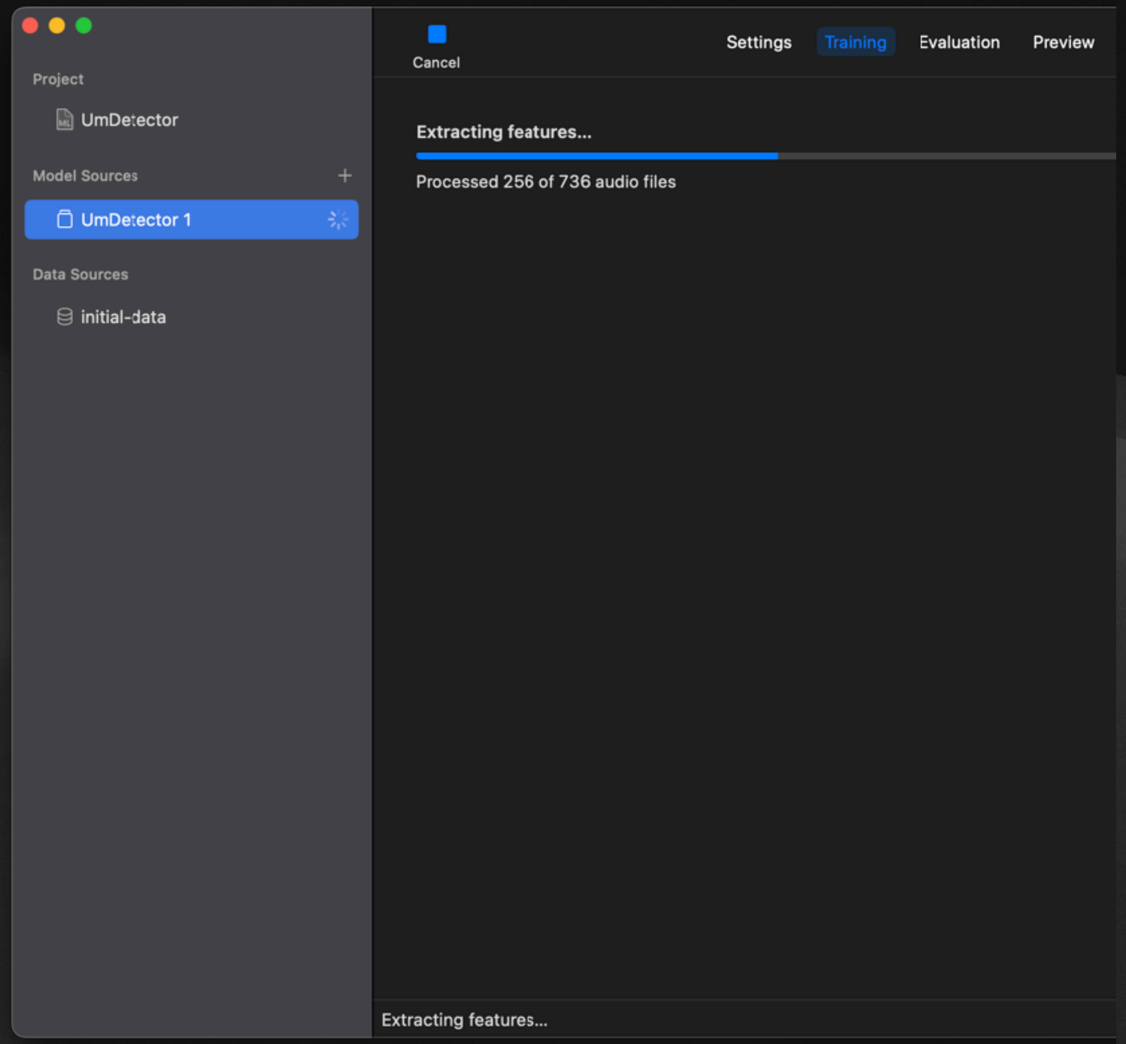
Create ML

Create a sound classification project in Create ML

Set project info

Add the dataset

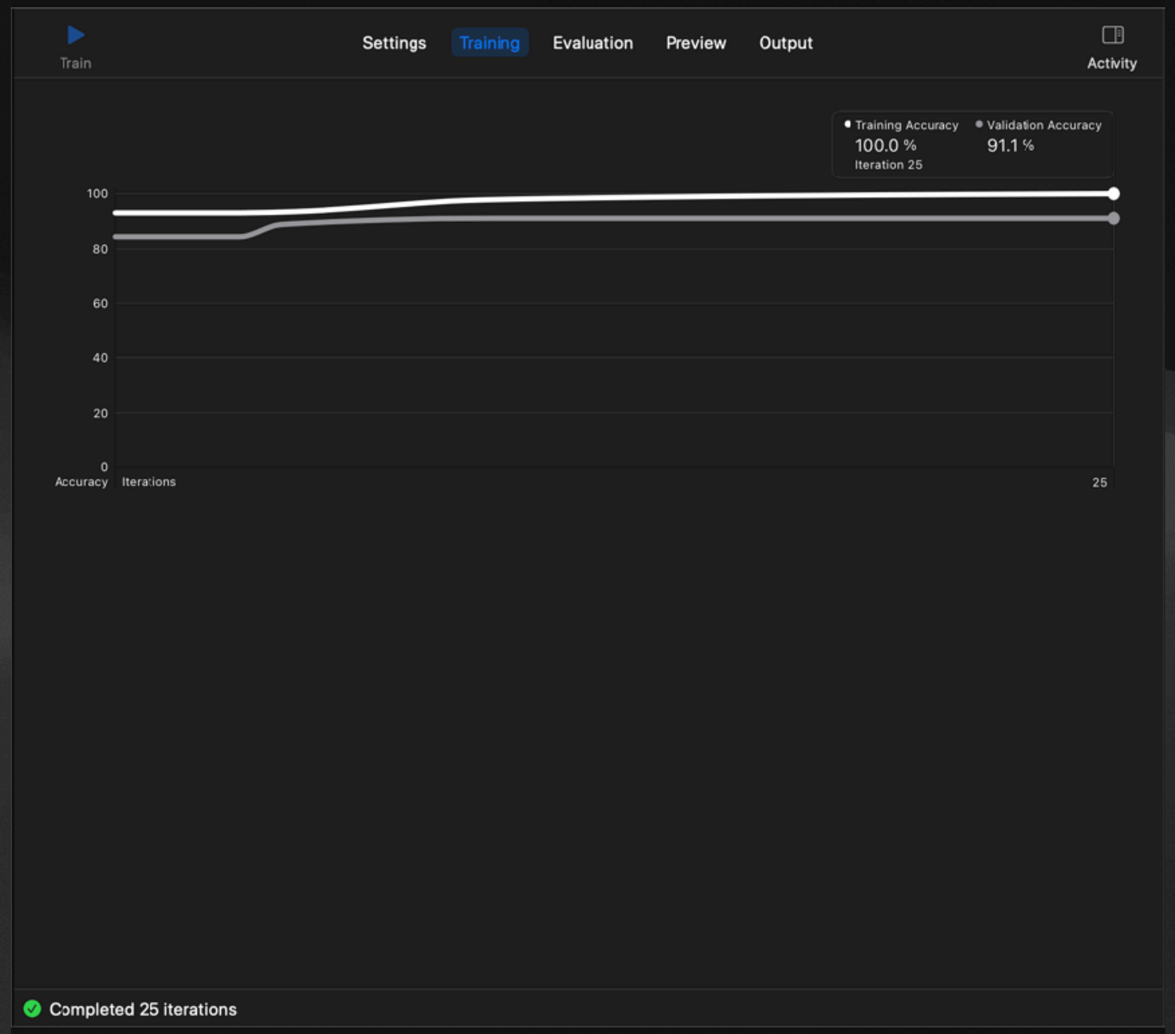
Train



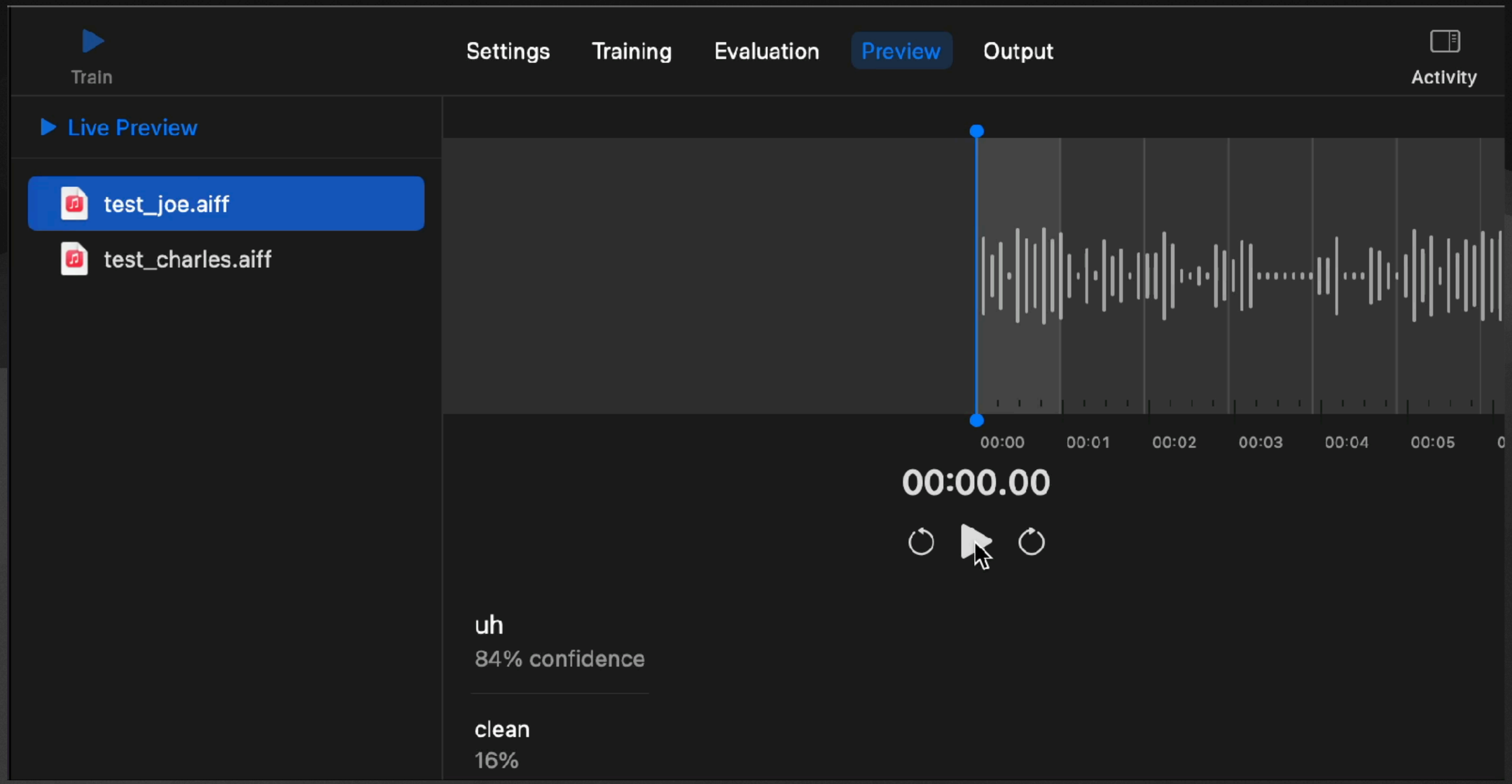
Accuracy

Training accuracy: 100%

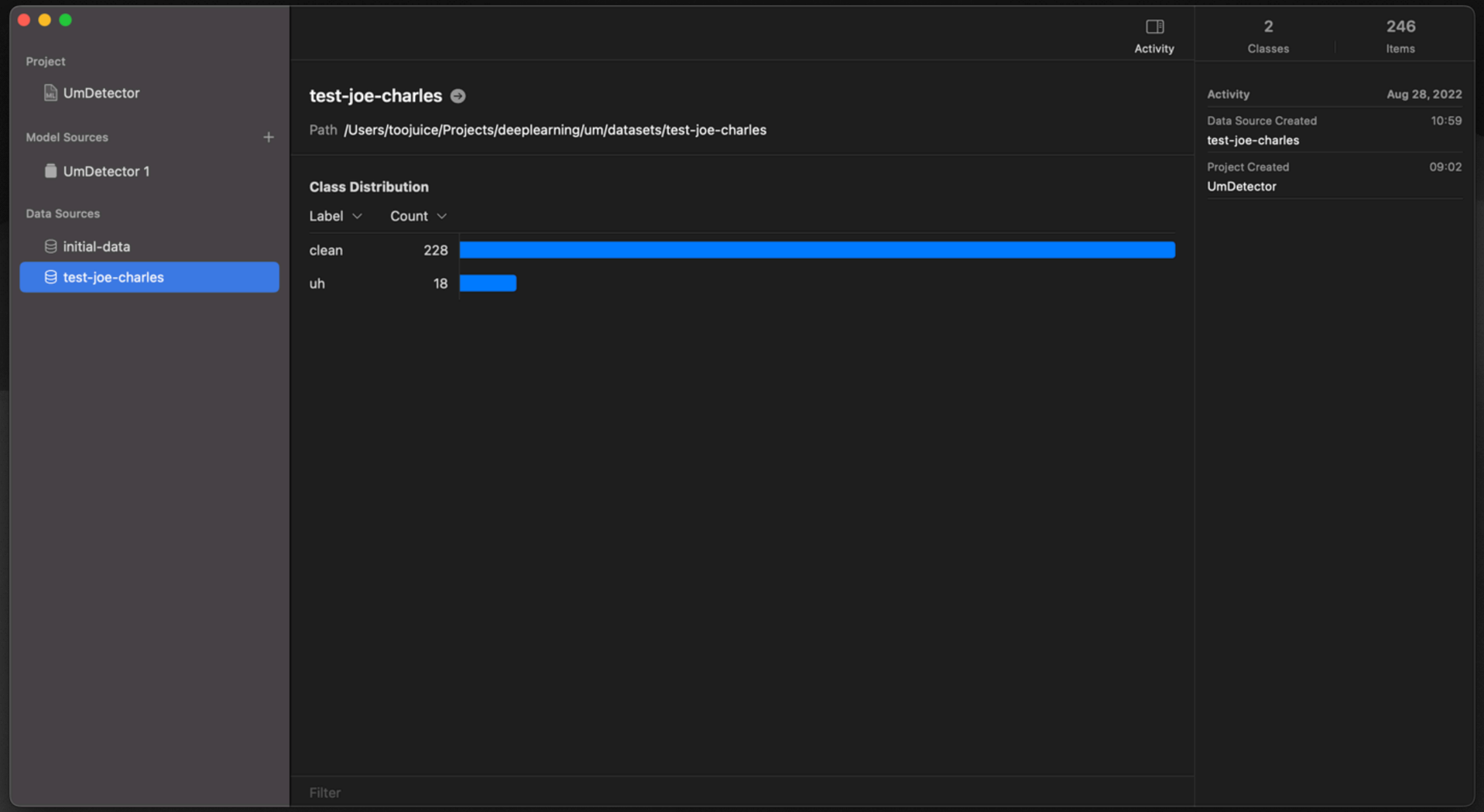
Validation accuracy: 91.1%



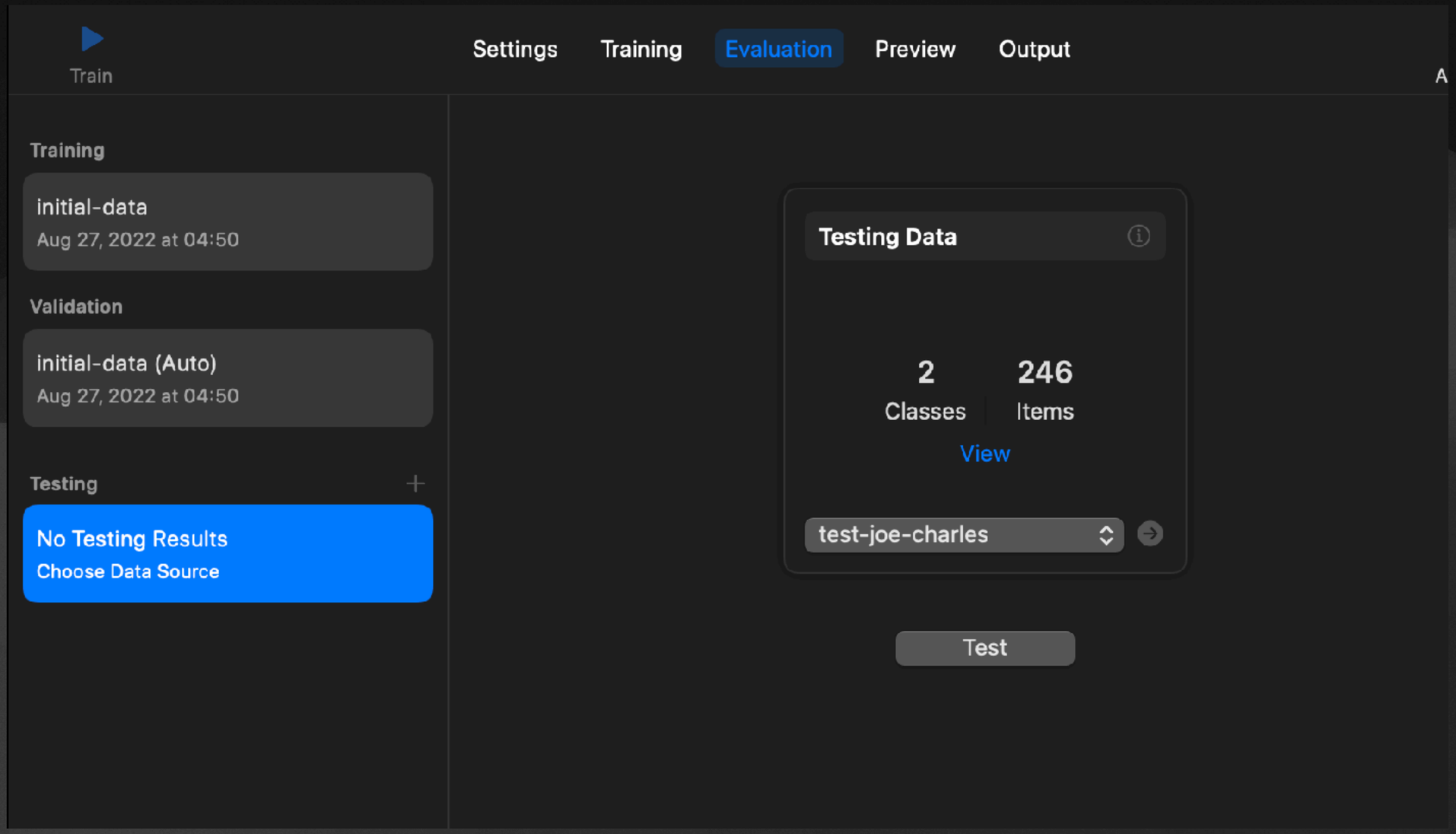
Testing - Subjective



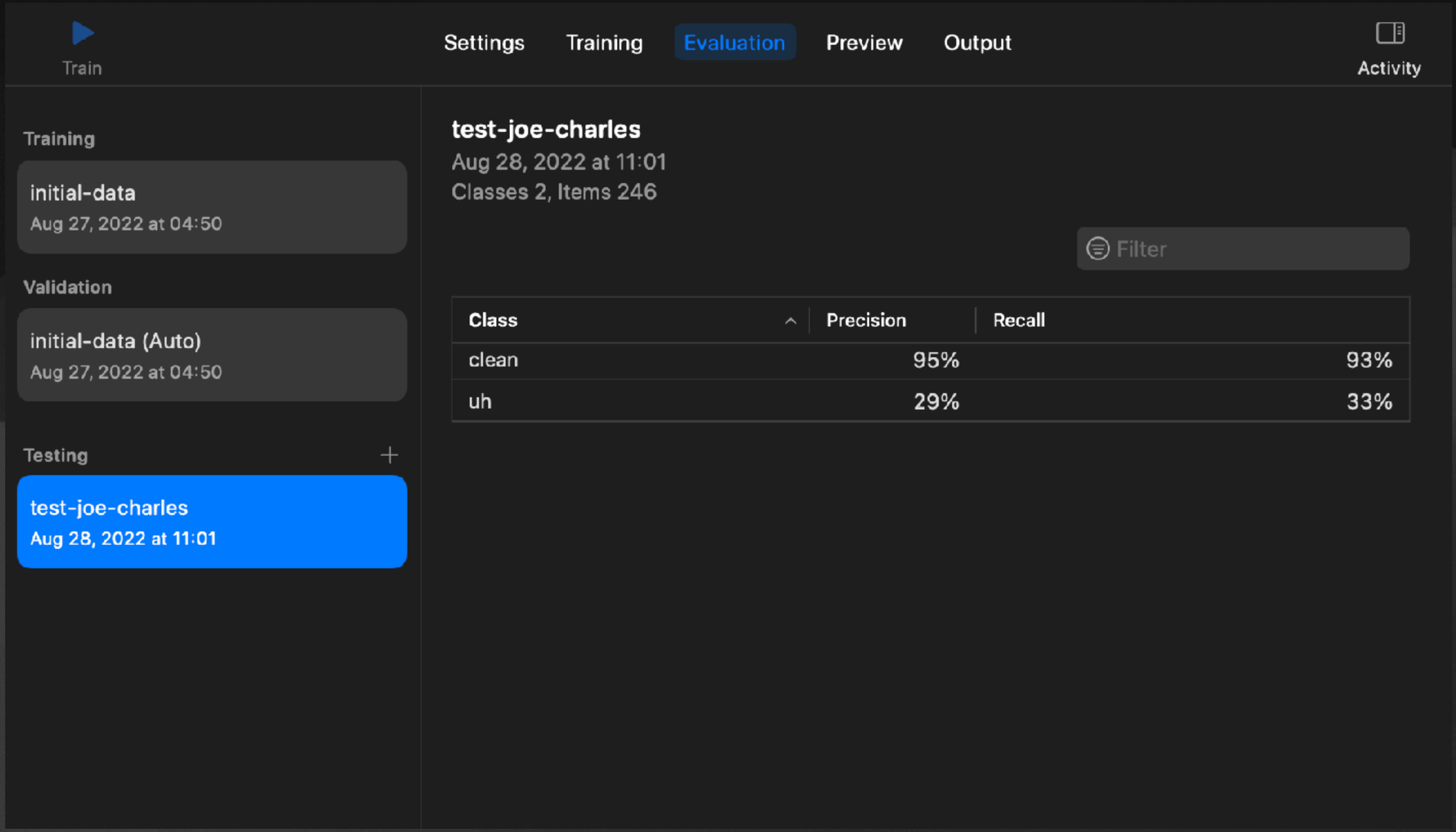
Testing - Objective



Testing - Objective



Testing - Objective



Testing - Objective

$$\frac{R_{clean} \cdot N_{clean} + R_{uh} \cdot N_{uh}}{N_{clean} + N_{uh}} = \text{Accuracy}$$

Testing - Objective

89%

IT WORKS!

sort of

Three Problems

Voice Diversity

Voice Diversity

Works well for audio recorded by Charles or Joe

Works well for me

Did not work for my wife

Did not work for my kids

Three Problems

Voice Diversity

Three Problems

Voice Diversity

Unbalanced data

Three Problems

Voice Diversity

Unbalanced data

Labeling data is manual and time-intensive

Attempt #3 - Still Machine Learning

More Data

Use our model to pre-label more data for us

Using a model in Swift script

SNRequest

SNResultsObserving

SNAudioFileAnalyzer

```swift
import CoreML
import SoundAnalysis
```

```swift
import CoreML
import SoundAnalysis

// Create the model configuration
let defaultConfig = MLModelConfiguration()
```

```swift
import CoreML
import SoundAnalysis

// Create the model configuration
let defaultConfig = MLModelConfiguration()

// Instantiate the Sound Classifier
let classifier = try! UmDetector(configuration: defaultConfig)
```

```swift
import CoreML
import SoundAnalysis

// Create the model configuration
let defaultConfig = MLModelConfiguration()

// Instantiate the Sound Classifier
let classifier = try! UmDetector(configuration: defaultConfig)

// Create the request using the Core ML model
let request = try! SNClassifySoundRequest(mlModel: classifier.model)
```

```
class UmObserver: NSObject, SNResultsObserving {

}
```

```swift
class UmObserver: NSObject, SNResultsObserving {

    // required method for SNResultsObserving protocol
    func request(_ request: SNRequest, didProduce result: SNResult) {









    }
}
```

```
class UmObserver: NSObject, SNResultsObserving {

    // required method for SNResultsObserving protocol
    func request(_ request: SNRequest, didProduce result: SNResult) {

        // get the SNClassification object out of the SNClassificationResult
        guard let result = result as? SNClassificationResult,
              let classification = result.classifications.first else {
            return
        }




    }
}
```

```
class UmObserver: NSObject, SNResultsObserving {

    // required method for SNResultsObserving protocol
    func request(_ request: SNRequest, didProduce result: SNResult) {

        // get the SNClassification object out of the SNClassificationResult
        guard let result = result as? SNClassificationResult,
              let classification = result.classifications.first else {
            return
        }

        // Do cool things with it
        print("Detected: \(classification.identifier)")
        print("    with: \(classification.confidence * 100.0)% confidence")
    }
}
```

```
class UmObserver: NSObject, SNResultsObserving {

    // required method for SNResultsObserving protocol
    func request(_ request: SNRequest, didProduce result: SNResult) {

        // get the SNClassification object out of the SNClassificationResult
        guard let result = result as? SNClassificationResult,
              let classification = result.classifications.first else {
            return
        }

        // Do cool things with it
        print("Detected: \(classification.identifier)")
        print("    with: \(classification.confidence * 100.0)% confidence")
    }
}

let resultsObserver = UmObserver()
```

```swift
// Create the analyzer
let audioAnalyzer = try! SNAudioFileAnalyzer(url: audioFileURL)
```

```swift
// Create the analyzer
let audioAnalyzer = try! SNAudioFileAnalyzer(url: audioFileURL)

// Add the request and the observer to the analyzer
try! audioAnalyzer.add(request, withObserver: resultsObserver)
```

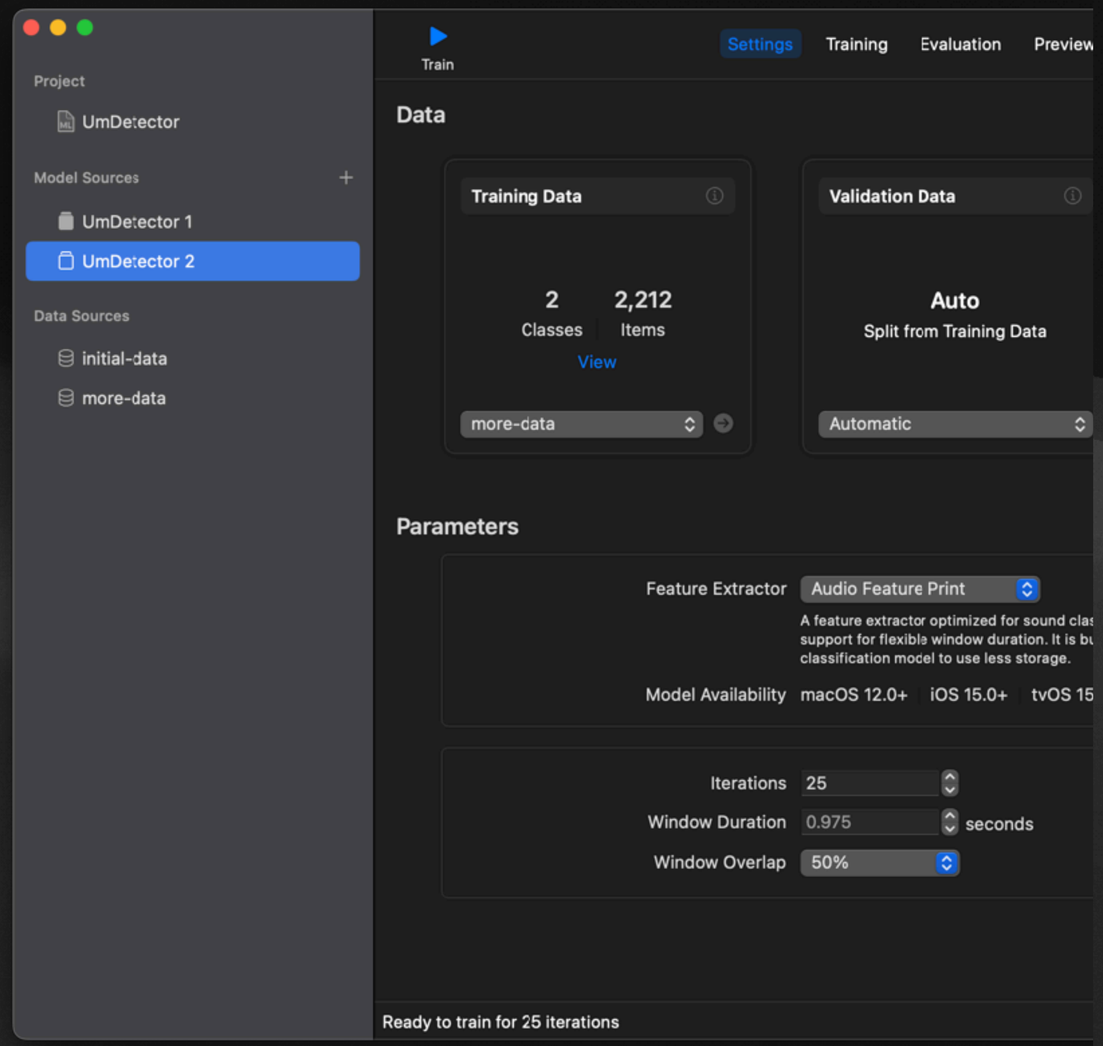
```swift
// Create the analyzer
let audioAnalyzer = try! SNAudioFileAnalyzer(url: audioFileURL)

// Add the request and the observer to the analyzer
try! audioAnalyzer.add(request, withObserver: resultsObserver)

// Start analyzing
audioAnalyzer.analyze()
```

More Data

	Initial	More Data
Clean	681	2046
Um	51	166
Total	736	2212

Create ML

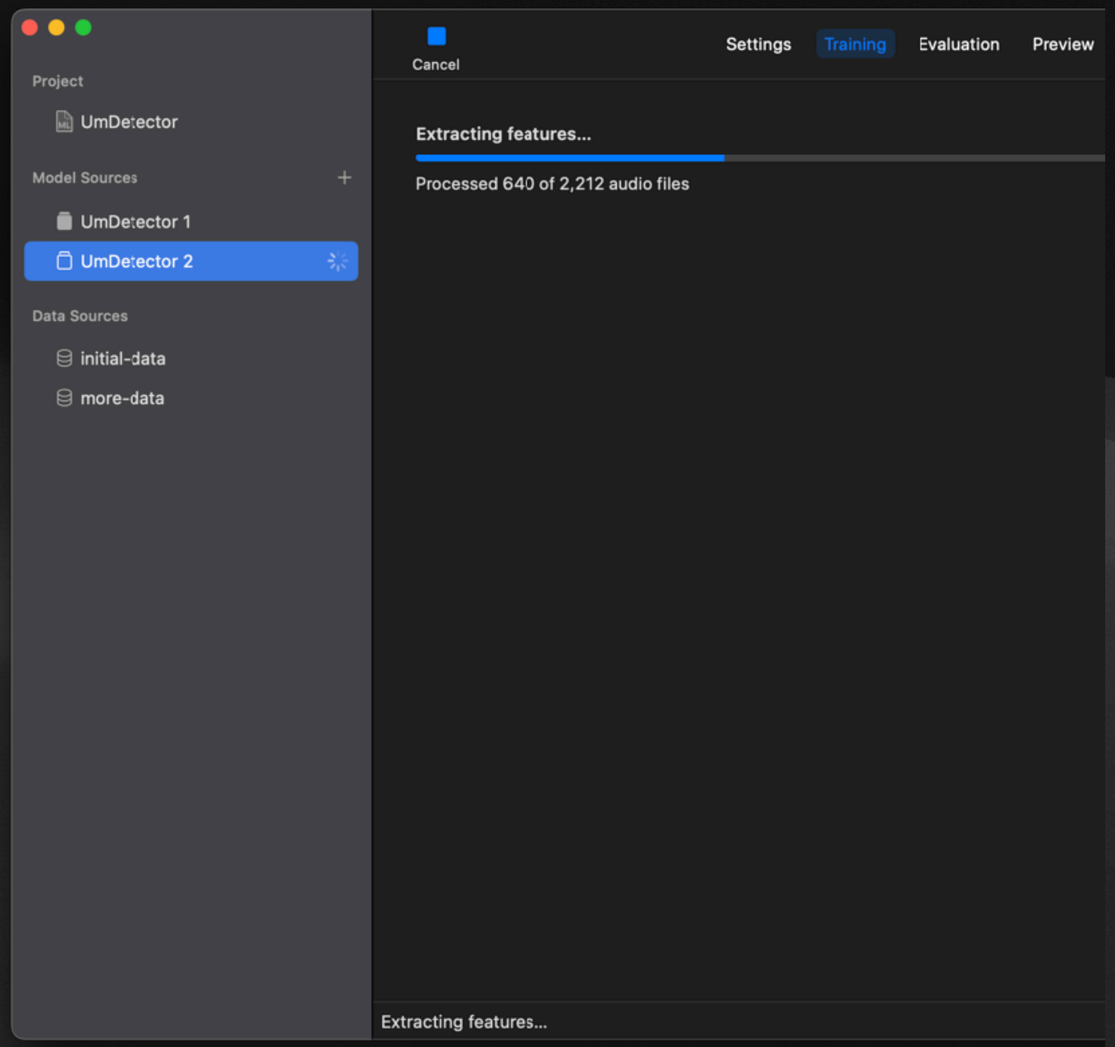
Add new dataset with more data



Create ML

Add new dataset with more data

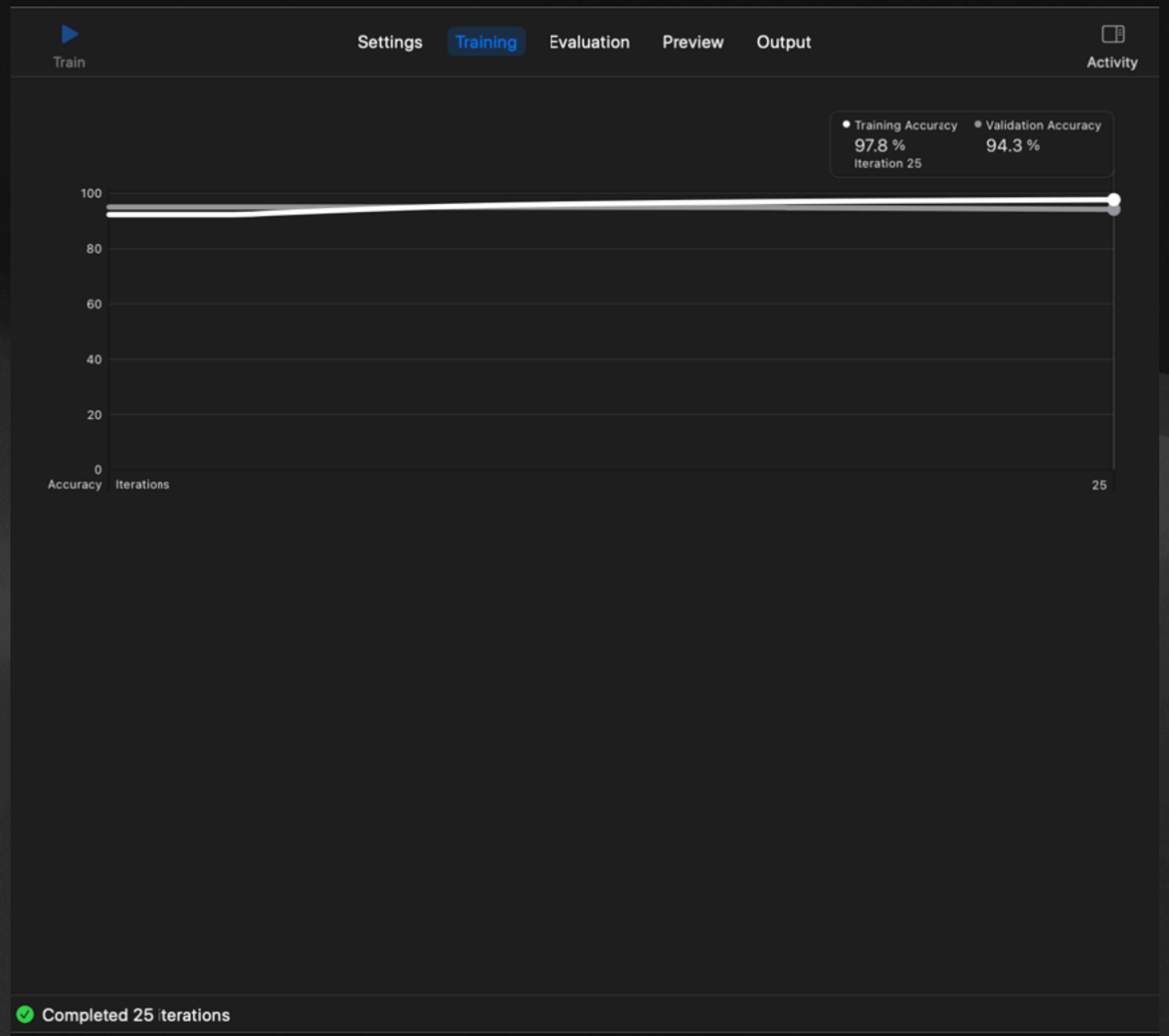
Retrain



Accuracy

Training accuracy: 97.8%

Validation accuracy: 94.3%

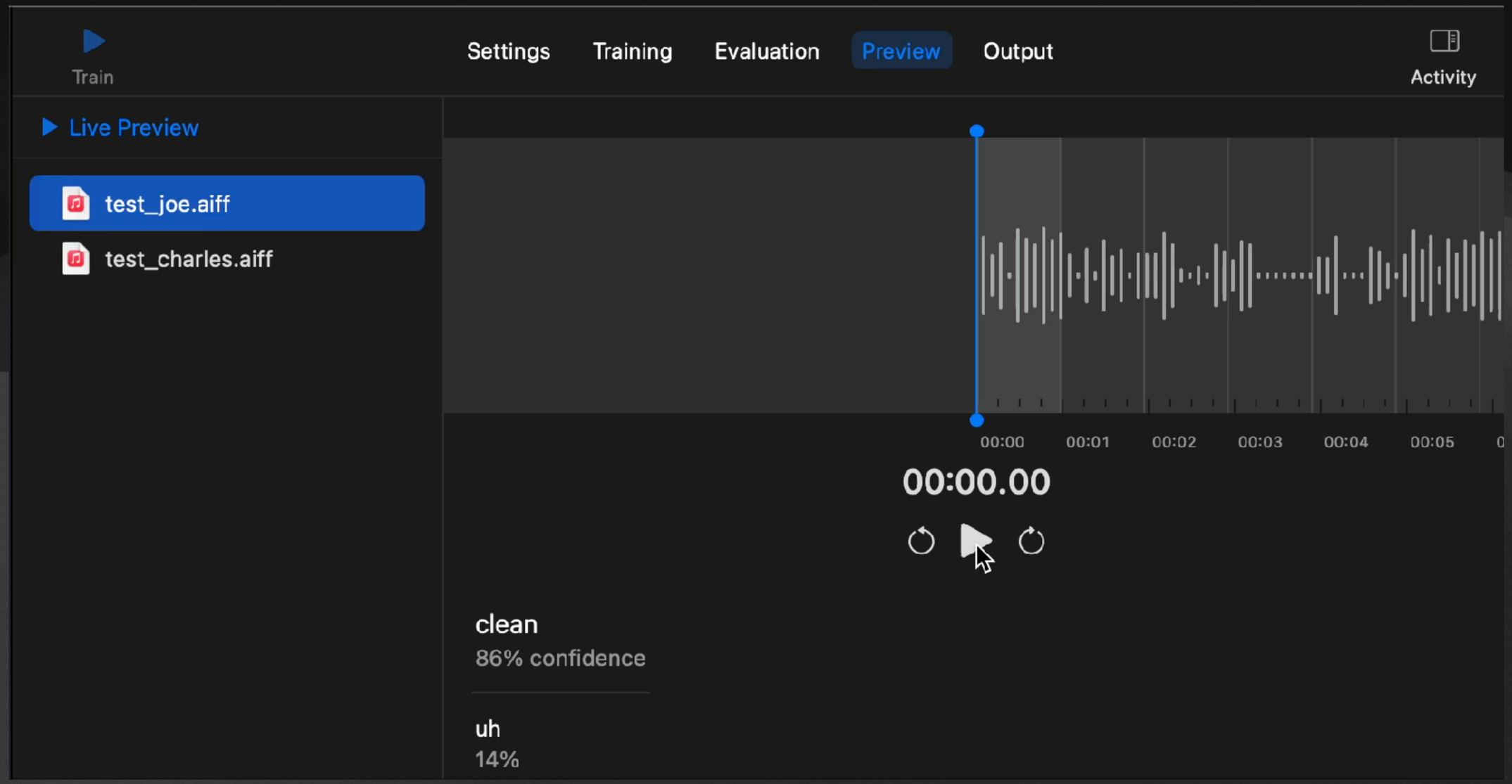


More Data

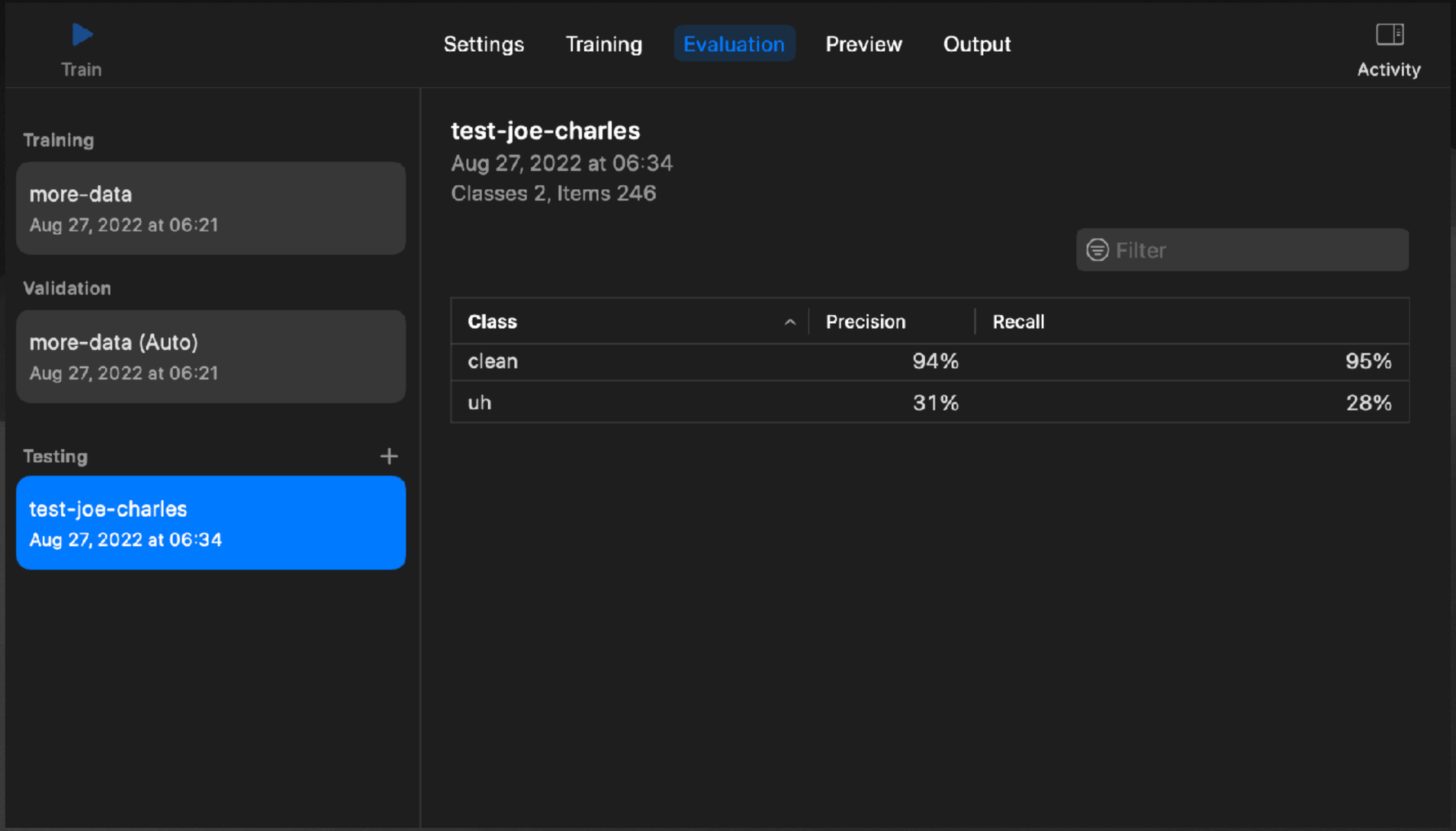
92.5%

	Initial	More Data
Clean	681	2046
Um	51	166
Total	736	2212

Accuracy - Subjective



Testing - Objective



Testing - Objective

90%

What does this do for us?

Improves the accuracy and robustness of Charles- and Joe-recorded audio

Does nothing for women and children

Attempt #4 - Still More Machine Learning

Augmented Data

Synthetically change our data to make the model more robust

Randomly change pitch of some samples

Randomly add noise to some samples

Ideally on the fly

Pre-baking is ok in a pinch

How do you do this?

Apple’s audio frameworks

Command line tool - sox

SOX

SOund eXchange

Many sound manipulation features

Install via Homebrew

>

> sox

```
> sox example.aif
```

```
> sox example.aif tmp.aif
```

```
> sox example.aif tmp.aif pitch 400.0
```

```
> sox example.aif tmp.aif pitch 400.0 loudness -10
```

>

> SOX

```
> sox tmp.aif
```

```
> sox tmp.aif -p synth whitenoise vol 0.05
```

```
> sox tmp.aif -p synth whitenoise vol 0.05 | sox
```

```
> sox tmp.aif -p synth whitenoise vol 0.05 | sox -m tmp.aif -
```

```
> sox tmp.aif -p synth whitenoise vol 0.05 | sox -m tmp.aif - example_augmented.aif
```

https://stackoverflow.com/questions/26971240/how-do-i-run-a-terminal-command-in-a-swift-script-e-g-xcodebuild

```swift
import Foundation

func safeShell(_ command: String) throws -> String {
    let task = Process()
    let pipe = Pipe()

    task.standardOutput = pipe
    task.standardError = pipe
    task.arguments = ["-c", command]
    task.executableURL = URL(fileURLWithPath: "/bin/zsh")
    task.standardInput = nil

    try task.run()

    let data = pipe.fileHandleForReading.readDataToEndOfFile()
    let output = String(data: data, encoding: .utf8)!

    return output
}
```

https://stackoverflow.com/questions/26971240/how-do-i-run-a-terminal-command-in-a-swift-script-e-g-xcodebuild

```
do {
    print(try safeShell(“sox --help“))
} catch {
    print("\(error)")
}
```

Pitch change example 1

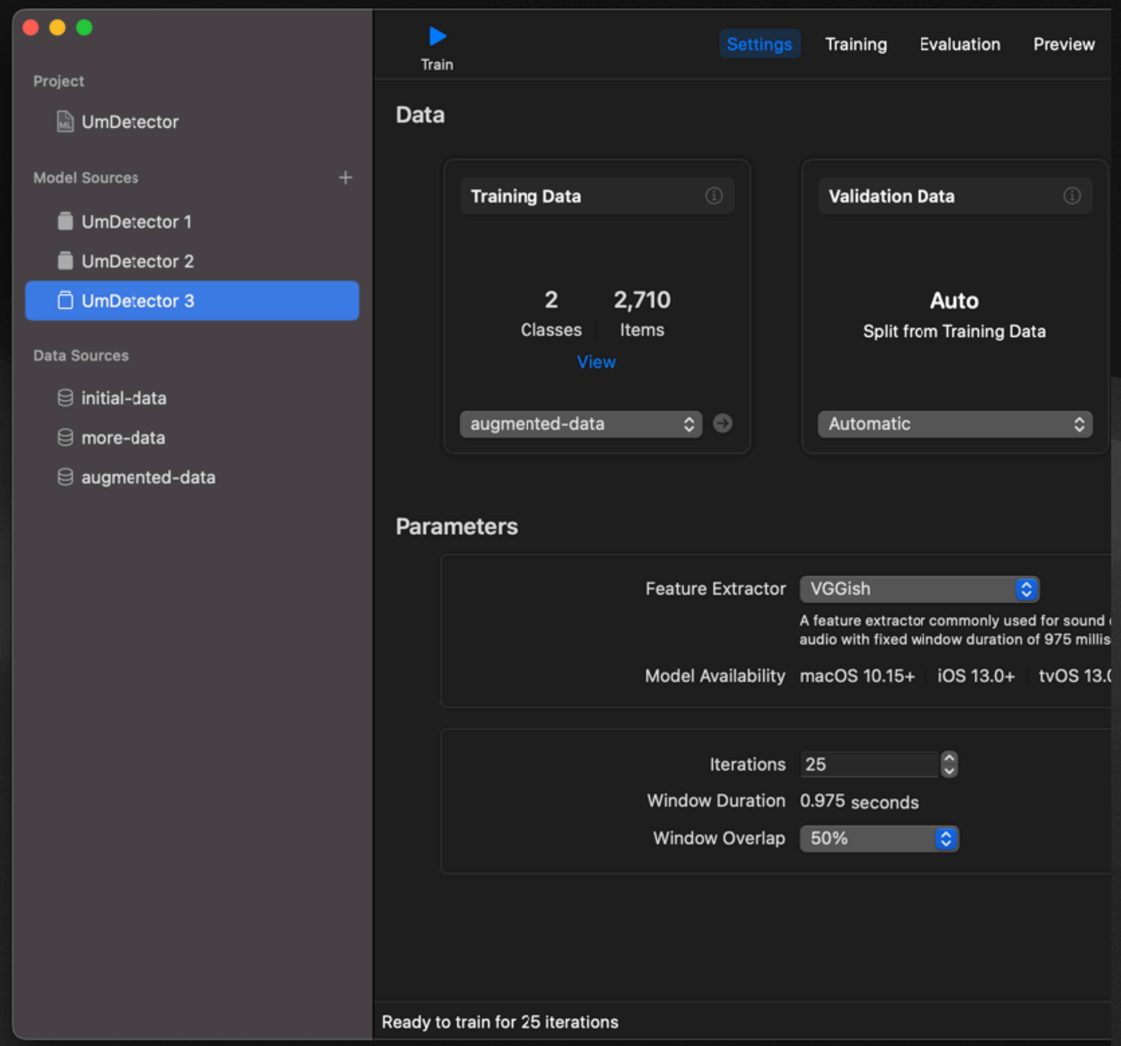
Pitch change example 2

Augmented Data

	Initial	More Data	Augmented
Clean	681	2046	2046
Um	51	166	664
Total	736	2212	2710

Create ML

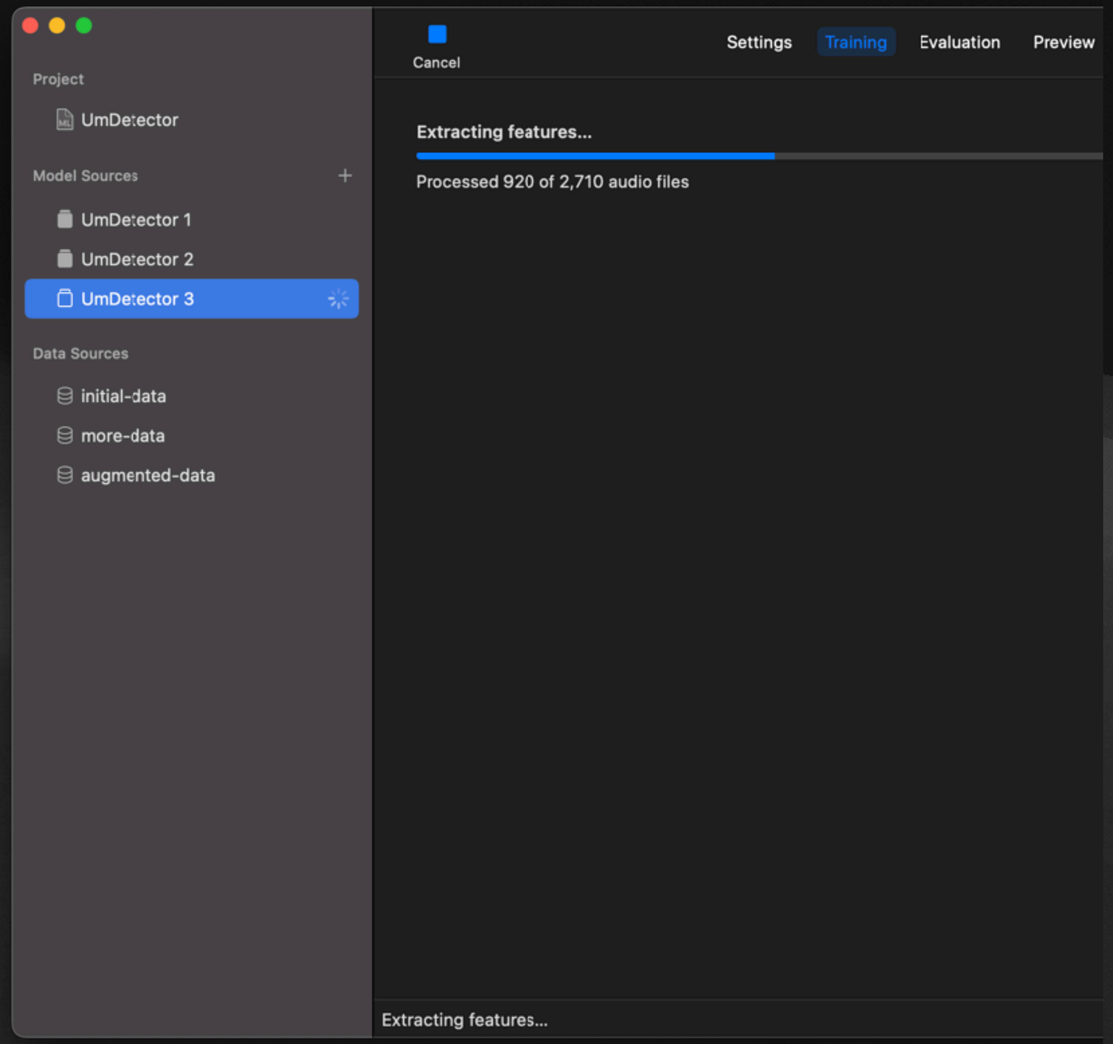
Add new augmented dataset



Create ML

Add new augmented dataset

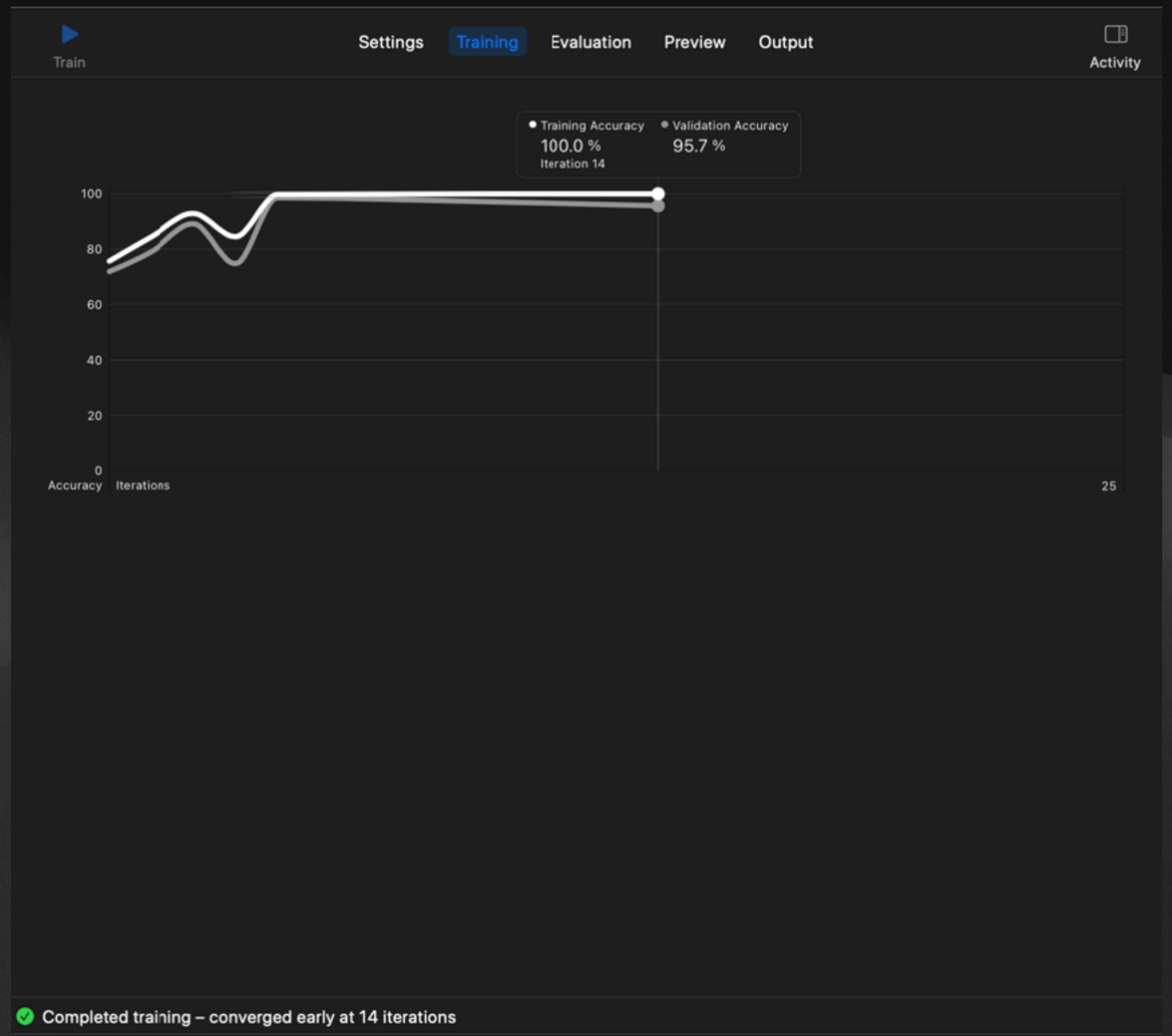
Retrain



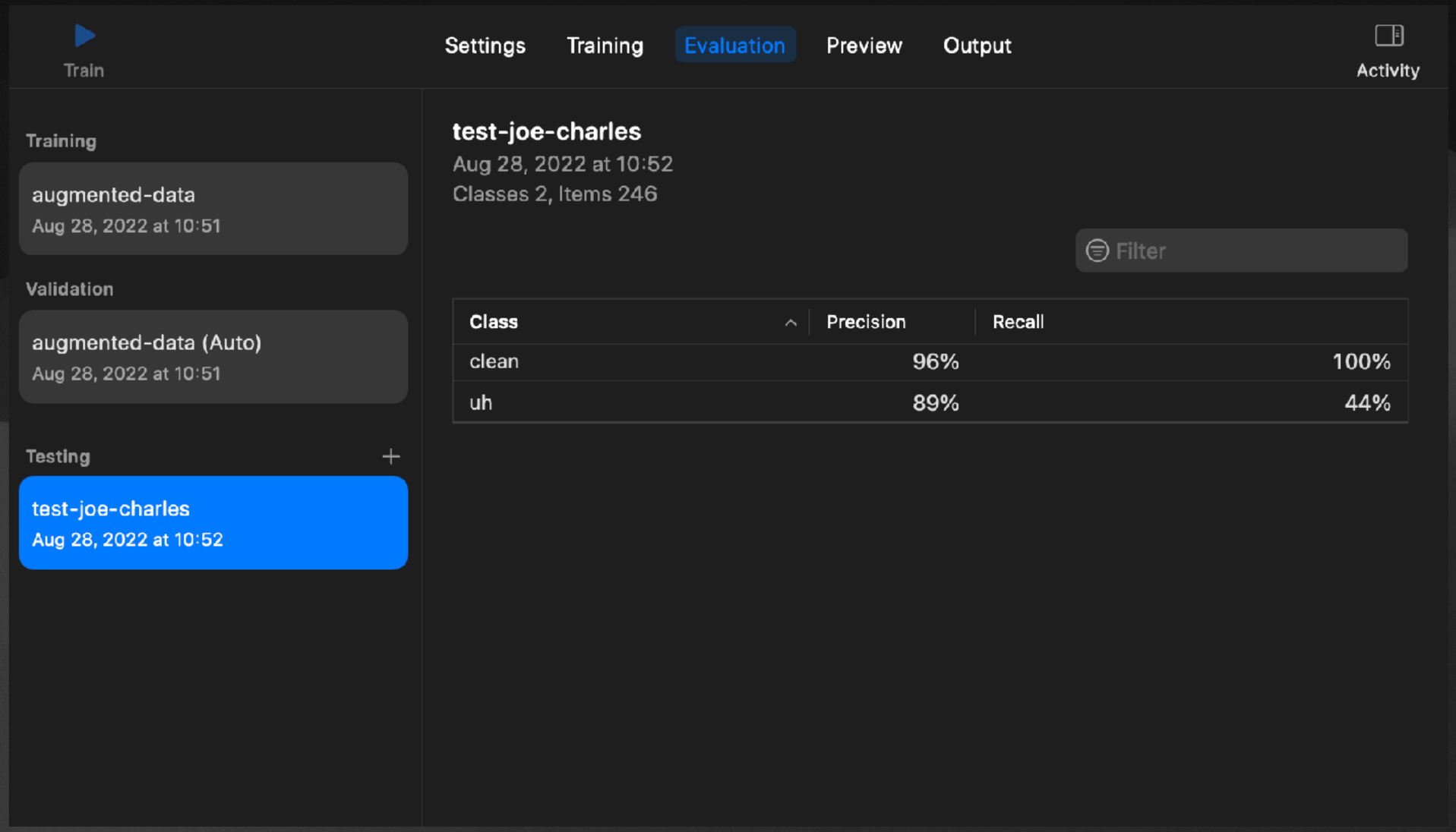
Accuracy

Training accuracy: 100.0%

Validation accuracy: 95.7%



Testing - Objective



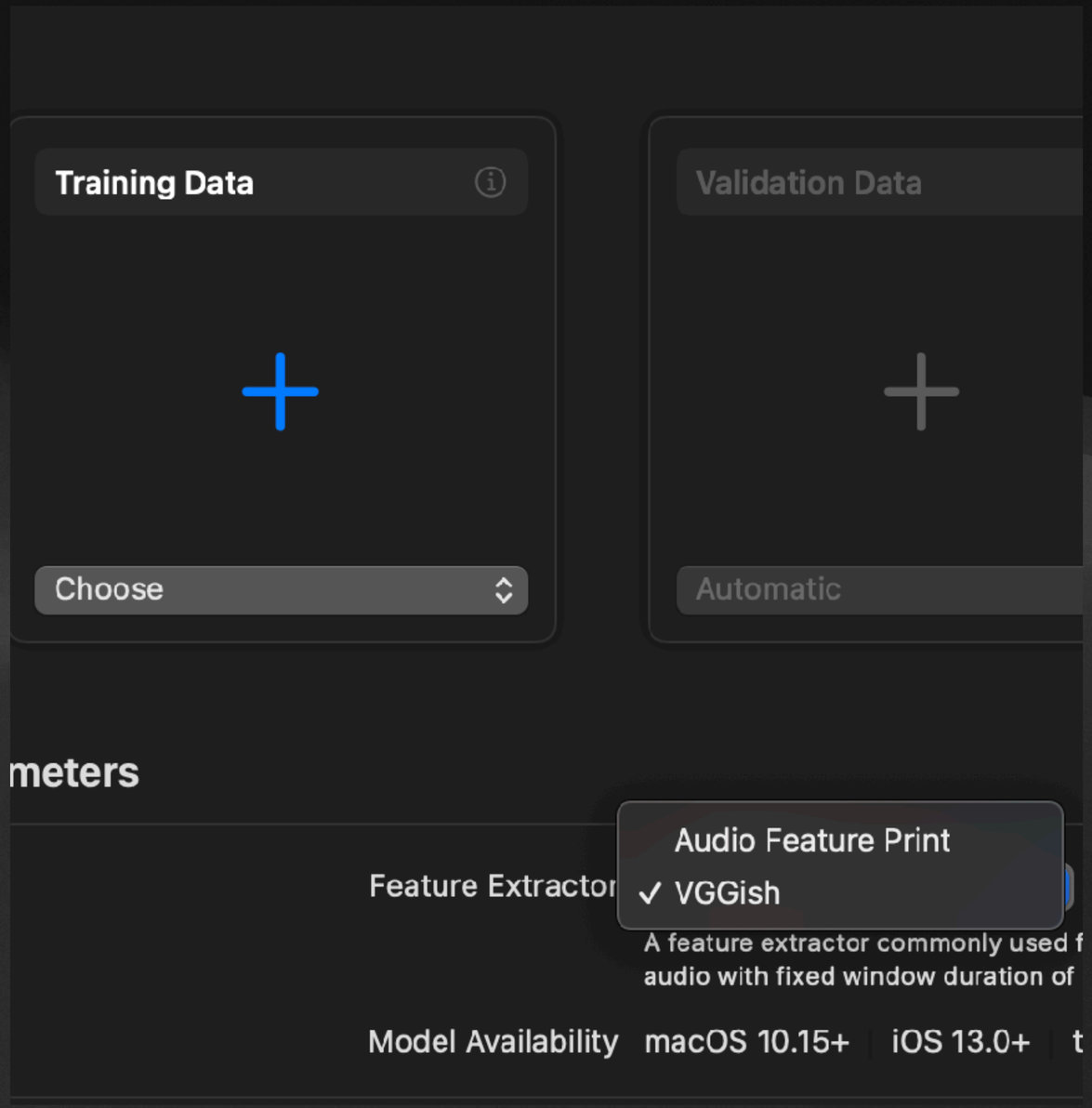
Testing - Objective

96%

Model Change

Feature extractor

- Apple’s Audio Feature Print - 5KB
- VGGish - 4.6MB

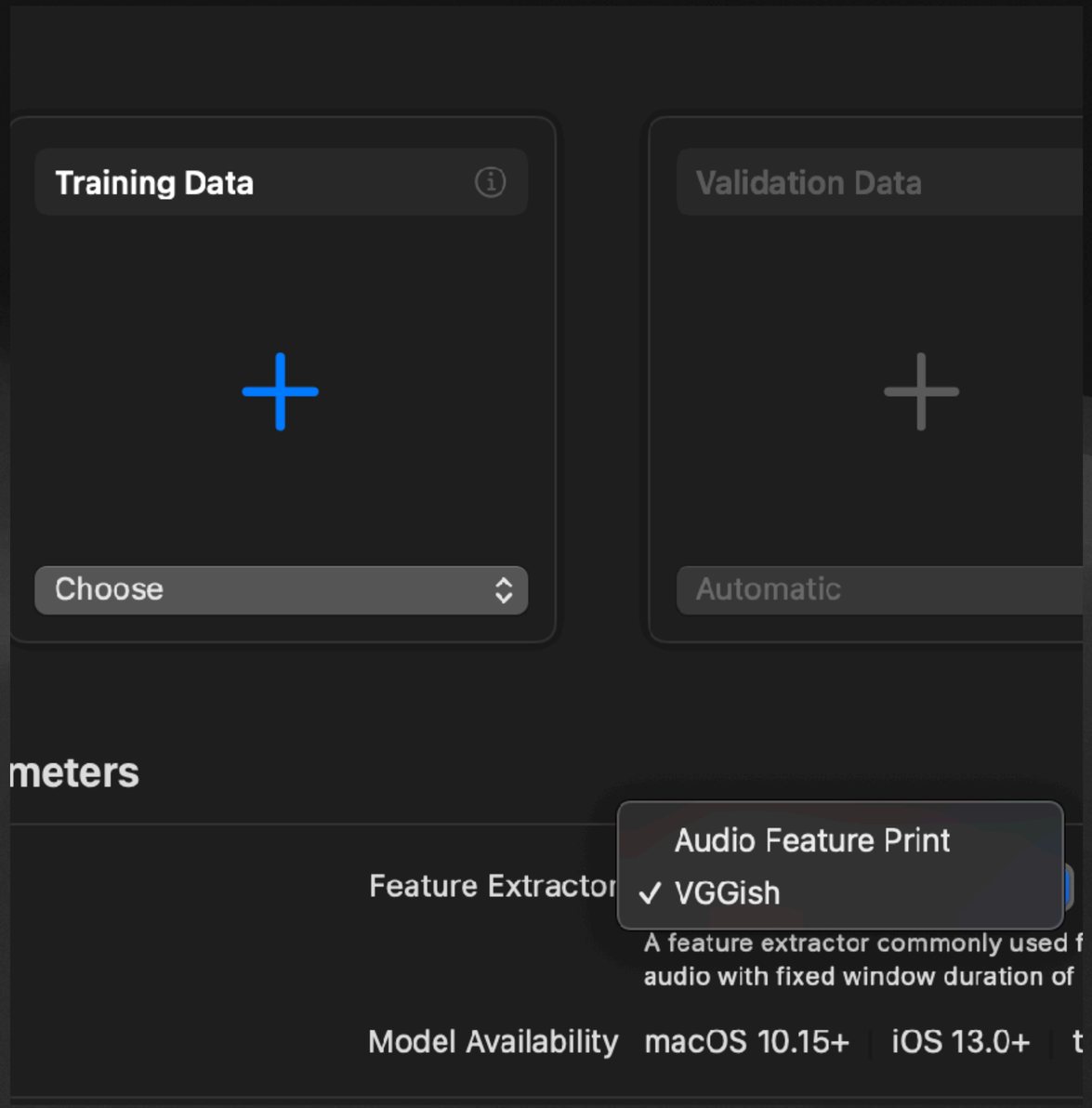


Model Change

Feature extractor

- Apple’s Audio Feature Print - 5KB
- VGGish - 4.6MB

Seemed to work better for this problem after a certain amount of data



What does this do for us?

Improves the accuracy and robustness of Charles and Joe recorded audio

Improves accuracy for women and children

Attempt #5 - Yet Still More Machine Learning

MORE
DATA

LORE
DATA

More Data

Record your own audio

Find more podcasters

Open Source Datasets?

Open Source Datasets

LibriSpeech - 1000 hours of English speech

Based on public domain audiobooks

http://www.openslr.org/12

HarperValleyBank - simulated contact center calls to Harper Valley Bank

8 kHz

https://github.com/cricketclub/gridspace-stanford-harper-valley

People's Speech - licensed for academic AND commercial use

30,000 hours of transcribed audio with diverse speakers

https://mlcommons.org/en/peoples-speech/

People’s Speech

Heard about it on a podcast

Licensed for academic AND commercial use

30,000 hours of transcribed audio with diverse speakers

2TB of data

Too much data!

Active Learning

Run current model on a subset of the data

Determine which “uh”s, “um”s, and clean audio it is not super confident about

Use that to train a new model

Optionally rinse and repeat

Active Learning

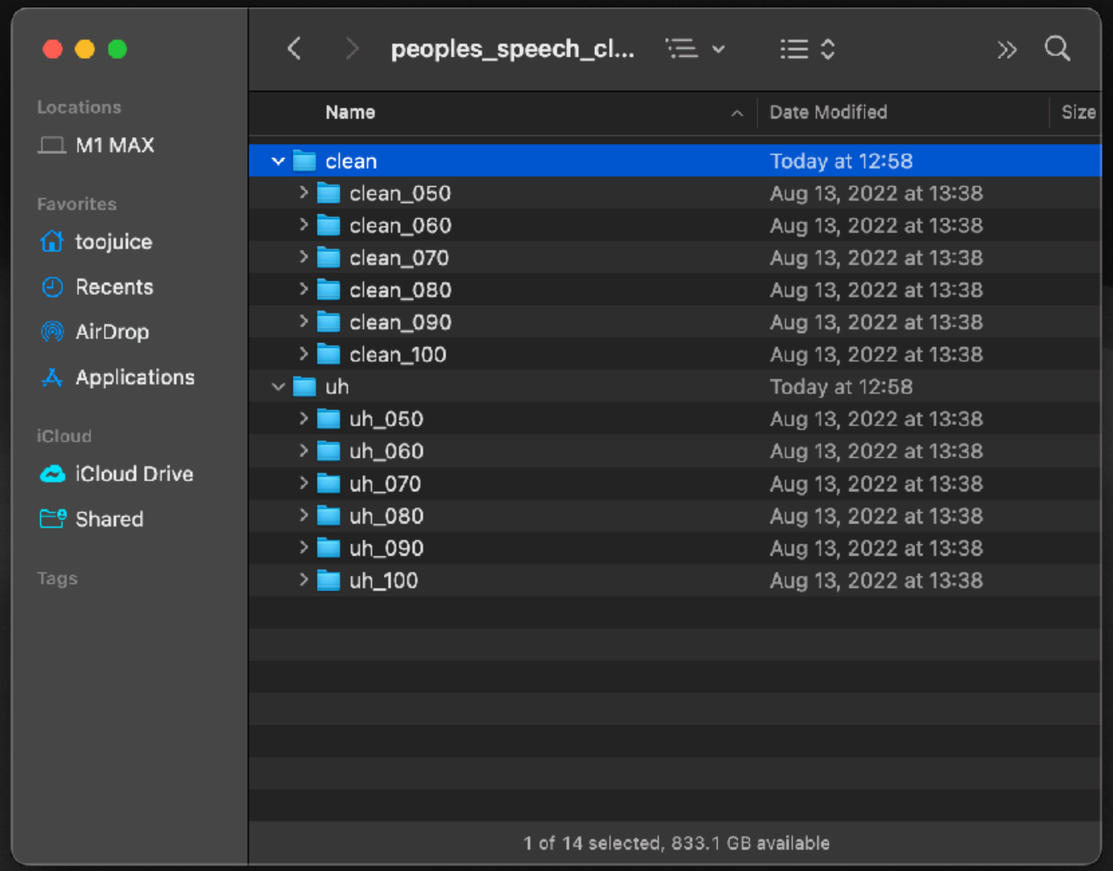
Swift CLI to run inference on audio

Sort files into confidence level

Start with the least confidence files

Generated 2.5 hours of data

Only added 13 minutes to dataset

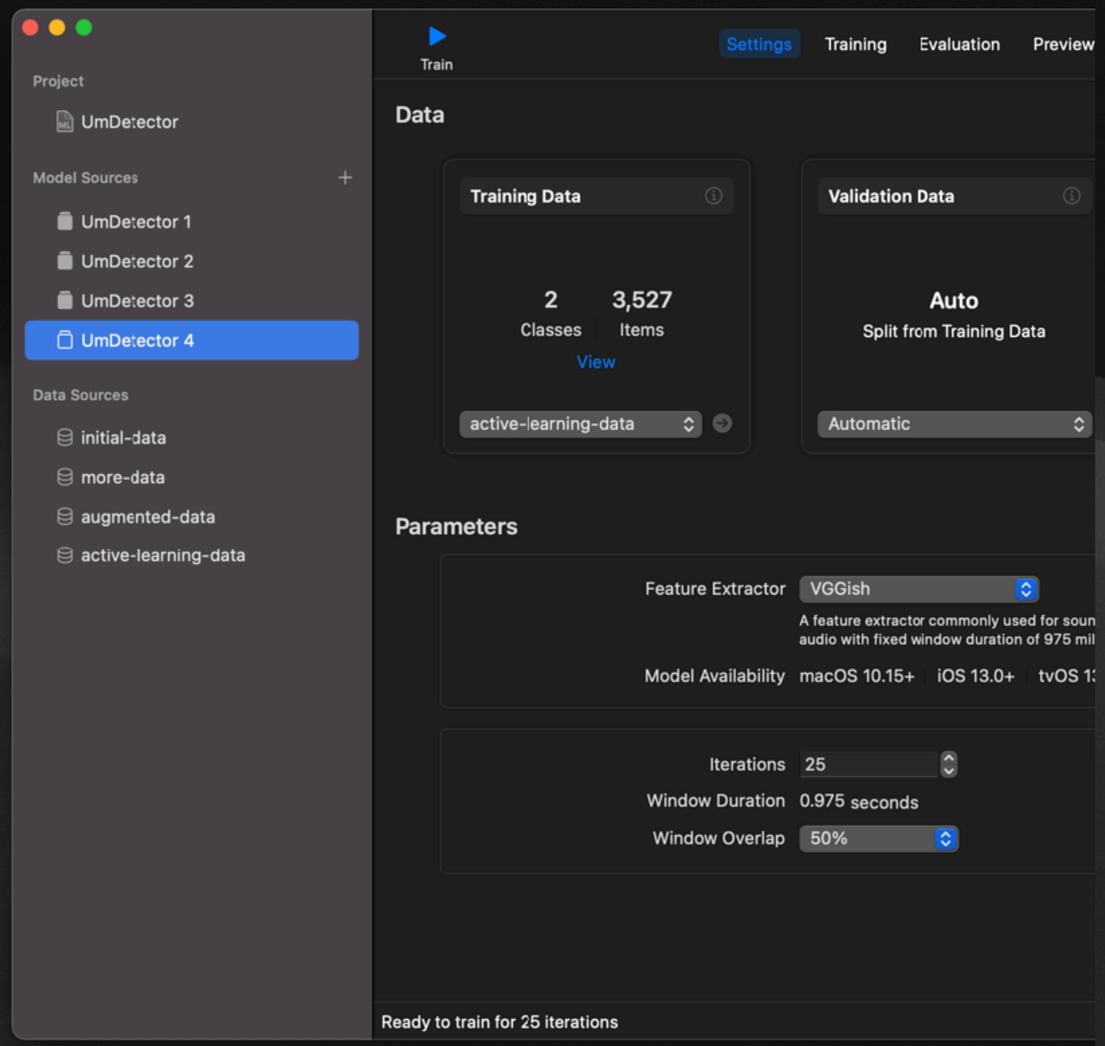


Augmented Data

	Initial	More Data	Augmented	Active Learning
Clean	681	2046	2046	2620
Um	51	166	664	907
Total	736	2212	2710	3527

Create ML

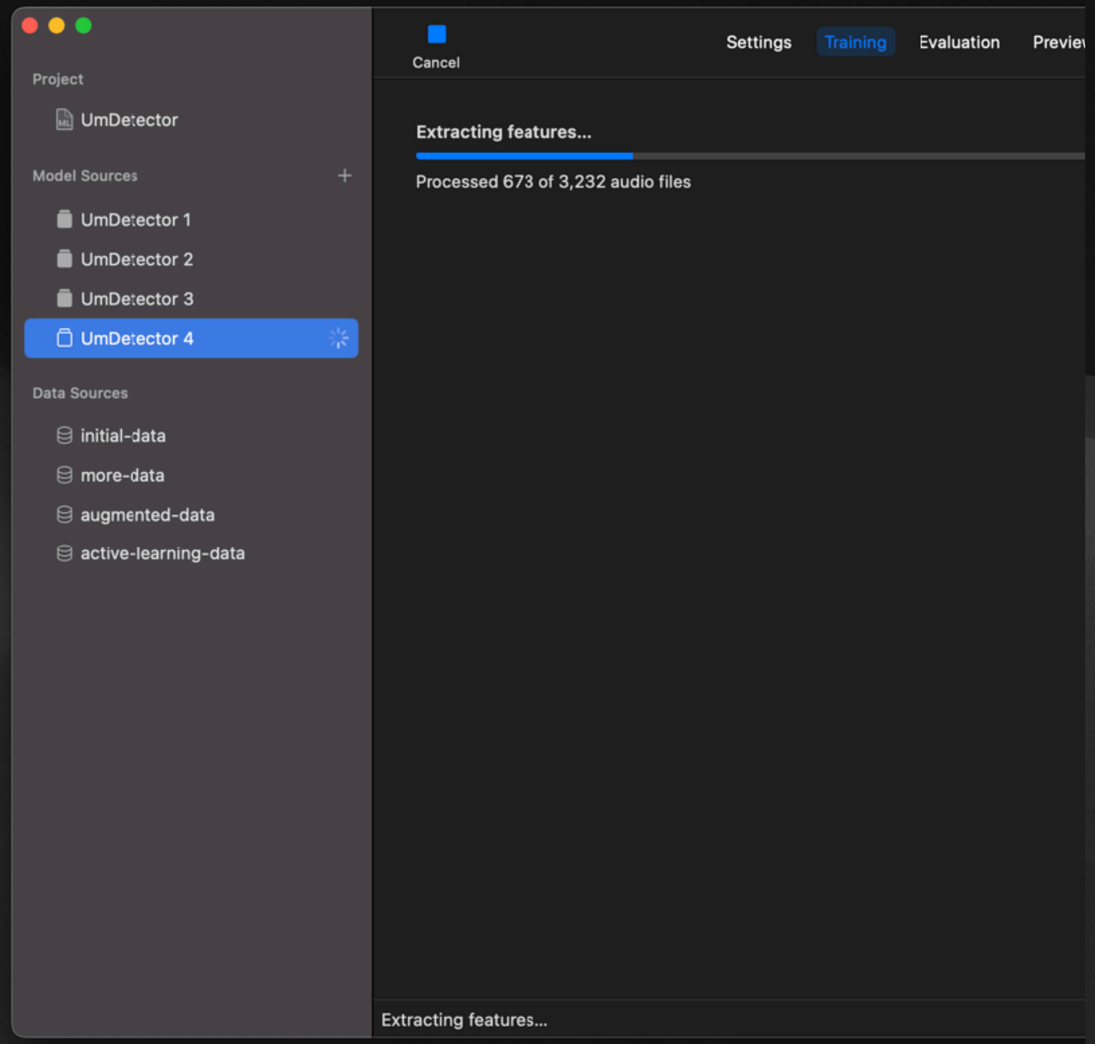
Add new active learning dataset



Create ML

Add new active learning dataset

Retrain

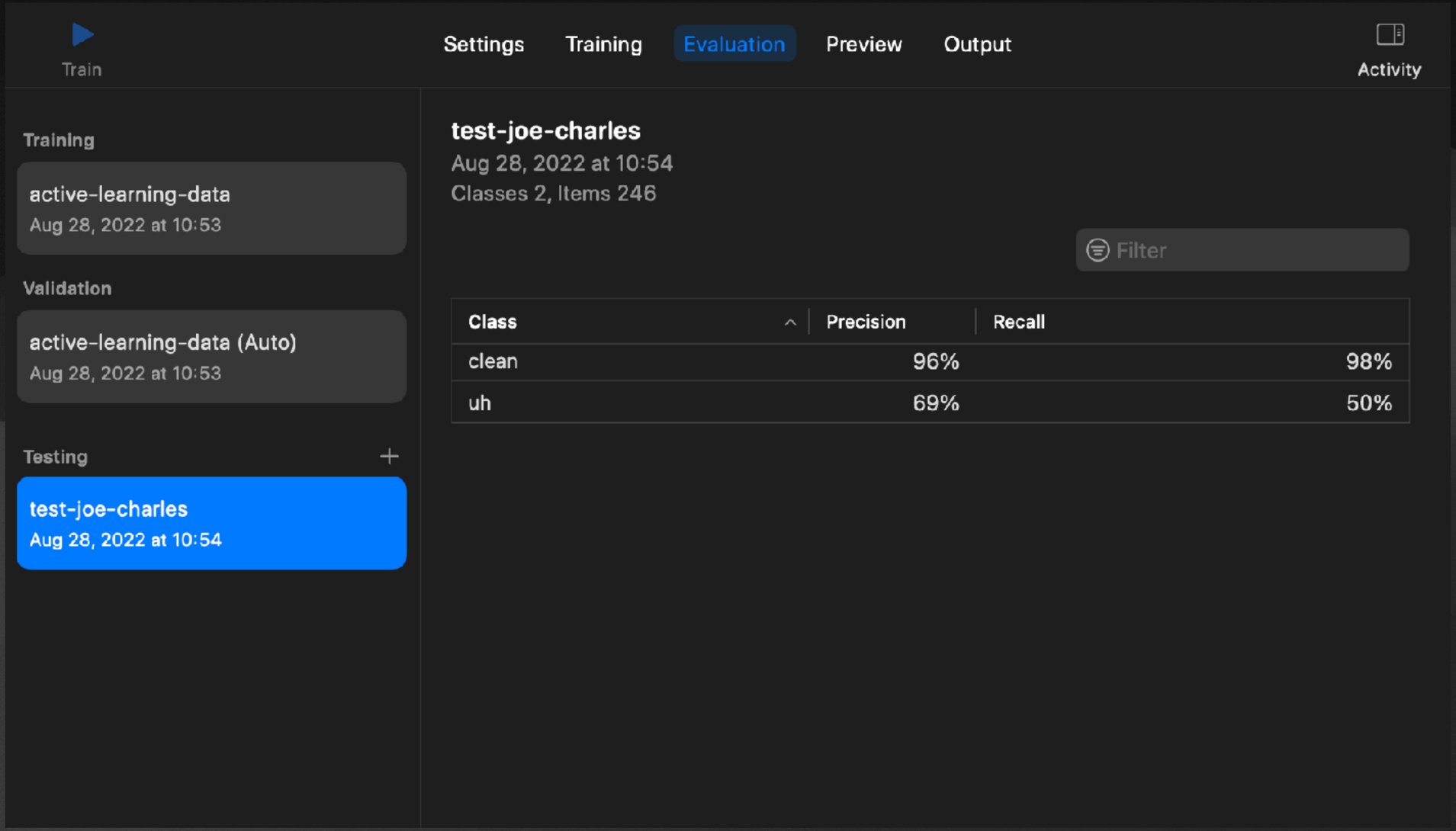


Accuracy

Training accuracy: 100.0%

Validation accuracy: 92.2%

Testing - Objective



Testing - Objective

94%

Wait a minute…

Why Shouldn’t This Work?

Are we really using the right tool for the job?

Ideally, you’d have a speech recognizer that doesn’t filter out “um”s and “uh”s

This method also introduces difficulties of it’s own

Difficulties

“The” pronounced as “thuh”

“A” pronounced as “uh”

Words that begin with “un” (understand)

Different accents may be troublesome

But it worked well enough

Phew!

Summary

Speech Recognition

Collecting data

Training models using Create ML

Metrics for analyzing our models

Augmenting data to improve our models

Active learning

Lots of bad jokes

What’s left to do?

Uhhhh...

Get more data?

https://dagshub.com/yonomitt/Um-Detector

Thank You

@yonomitt

https://dagshub.com/yonomitt/Um-Detector

One more thing…

```
let microManager =
MicrophoneManager.shared
```

Thank You

@yonomitt

https://dagshub.com/yonomitt/Um-Detector