Creating a model for audio use cases by using "Cascade Windowing"

This notebook is used to build a model that use 5 sec. of audio data to create classification prediction by using cascade windowing.

The data you are going to use was collected from microphone sensors (sample rate is 16000) and formatted using the Data Capture Lab.

Upload Sensiml Libraries

In [1]:
```
from sensiml import SensiML
from sensiml.widgets import *
```

Connect server

To access the data first connect to the knowledge builder cloud service

In [2]:
```
dsk = SensiML()
```

Initialize a knowledge builder project

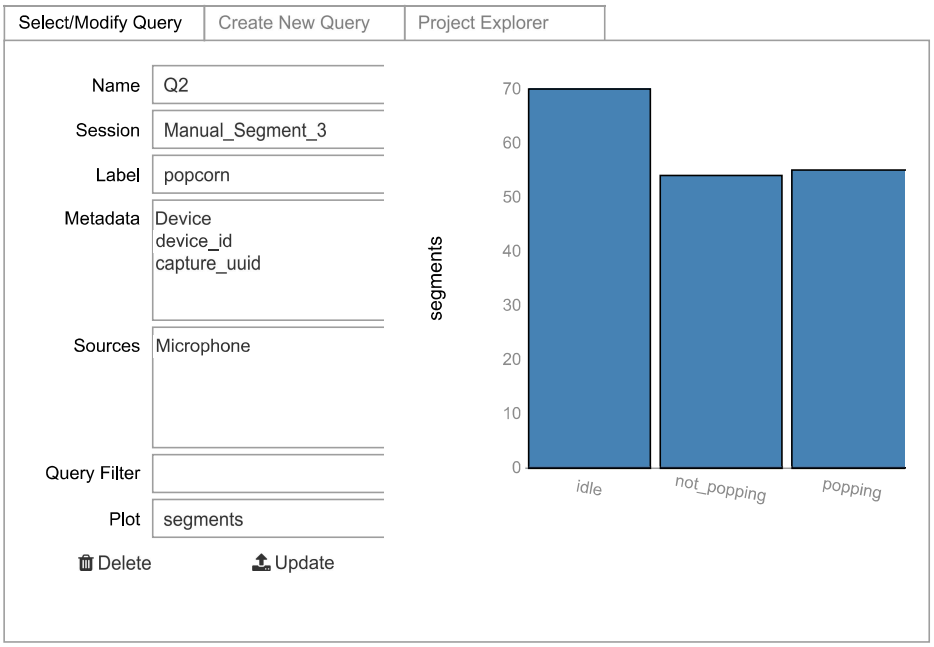
The data for this project has already been labeled and uploaded to the Project "Popcorn_pop".

In [3]:
```
dsk.project = 'Popcorn_pop'
dsk.pipeline = 'pipeline_cast'
```

Create a query

Define the data that will be used to create a model

In [4]:
```
qr = QueryWidget(dsk)
qr.create_widget()
```



In [ ]:
```
```

Define the variables and feaure list

In [5]:
```
sensor_columns = ['Microphone']
sample_rate = 16000

feature_list = [{'subtype_call': 'Area', 'params': {'sample_rate': sample_rate, 'smooth
                {'subtype_call': 'Time', 'params': {'sample_rate': sample_rate}},
                {'subtype_call': 'Rate of Change'},
                {'subtype_call': 'Statistical'},
                {'subtype_call': 'Energy'}]
```

Create the model

If "Cascade Windowing" is defined to 8000, model creates feature set for each 0.5 sec (8000 time points). Since number of cascades in "Feature Cascade" function is defined to 10, it will keep creating feature sets (for each 0.5 sec of data) during a 5 sec. And feature selection algorithm select the most significant features from the feature list for each 0.5 sec during the 5 sec. At the end of each 5 sec. of data, a prediction will be created for classification.

In [6]:
```
dsk.pipeline.reset(delete_cache=False )

dsk.pipeline.set_input_query("Q2")

dsk.pipeline.add_transform("Cascade Windowing",
                           params={"window_size": 8000, "delta": 8000,
                                   "return_segment_index": False})

dsk.pipeline.add_feature_generator(feature_list, function_defaults={'columns': sensor_c

dsk.pipeline.add_transform("Feature Cascade", params={"num_cascades": 10, "slide": Fals

dsk.pipeline.add_transform("Min Max Scale")

dsk.pipeline.add_feature_selector([{'name':'Univariate Selection',
                                    'params':{"number_of_features": 10 }}] )

dsk.pipeline.set_validation_method('Recall')
#dsk.pipeline.set_validation_method("Stratified K-Fold Cross-Validation",
#                                   params={"number_of_folds": 5 })

dsk.pipeline.set_training_algorithm("Random Forest",
                                    params={"max_depth": 3, "n_estimators": 85 })

dsk.pipeline.set_classifier("Decision Tree Ensemble")

dsk.pipeline.set_tvo({'validation_seed': 0 })

results, stats =  dsk.pipeline.execute()
results.summarize()
```

```
Executing Pipeline with Steps:

--------------------------------------------------------------------------
 0.     Name: Q2                                        Type: query
--------------------------------------------------------------------------
--------------------------------------------------------------------------
 1.     Name: Cascade Windowing                         Type: segmenter
--------------------------------------------------------------------------
--------------------------------------------------------------------------
 2.     Name: generator_set                             Type: generatorset
--------------------------------------------------------------------------
--------------------------------------------------------------------------
 3.     Name: Feature Cascade                           Type: transform
--------------------------------------------------------------------------
--------------------------------------------------------------------------
 4.     Name: Min Max Scale                             Type: transform
--------------------------------------------------------------------------
--------------------------------------------------------------------------
 5.     Name: selector_set                              Type: selectorset
--------------------------------------------------------------------------
--------------------------------------------------------------------------
 6.     Name: tvo                                       Type: tvo
--------------------------------------------------------------------------
        Classifier: Decision Tree Ensemble


        Training Algo: Random Forest
                max_depth: 3
                n_estimators: 85

        Validation Method: Recall


--------------------------------------------------------------------------


Status: Running, Time: 0.34, STEP:   0/7, NAME:         Q2, TYPE:      query, BATCH:
9/9 ...
Status: Running, Time: 1.36, STEP:   0/7, NAME:         Q2, TYPE:      query, BATCH:
9/9 ...
Status: Running, Time: 2.38, STEP:   0/7, NAME:         Q2, TYPE:      query, BATCH:
9/9 ...
Status: Running, Time: 3.39, STEP:   0/7, NAME:         Q2, TYPE:      query, BATCH:
9/9 ...
Status: Running, Time: 4.41, STEP:   0/7, NAME:         Q2, TYPE:      query, BATCH:
9/9 ...
Status: Running, Time: 5.42, STEP:   1/7, NAME: Cascade Wi, TYPE:  segmenter, BATCH:
9/9 ...
Status: Running, Time: 6.44, STEP:   2/7, NAME: generator_, TYPE: generators, BATCH:
```

```
9/9 ...

Results Retrieved... Execution Time: 7 min. 22 sec.
TRAINING ALGORITHM: Random Forest
VALIDATION METHOD:  Recall
CLASSIFIER:         Decision Tree Ensemble

AVERAGE METRICS:
F1-SCORE:    90.4   sigma 0.00
SENSITIVITY: 90.8   sigma 0.00
PRECISION:   90.5   sigma 0.00

--------------------------------------

RECALL MODEL RESULTS : SET VALIDATION

MODEL INDEX: Fold 0
        f1-score        train: 91.15  validate: 91.15
        accuracy        train: 90.44  validate: 90.44
        sensitivity     train: 90.48  validate: 90.48
```

In [ ]:

Create confusion matrix

In [7]:
```
model = results.configurations[0].models[0]
cm = model.confusion_matrix_stats
cm_df = cm['validation'].confusion_matrix_data_frame
display(cm_df)
```

	ClassIn	idle	not_popping	popping	UNK	UNC	Support	Sens(%)
0	idle	43.0	0.000000	0.000000	0.0	0.0	43	100.000000
1	not_popping	0.0	32.000000	3.000000	0.0	0.0	35	91.428571
2	popping	0.0	7.000000	28.000000	0.0	0.0	35	80.000000
3	UNK	0.0	0.000000	0.000000	0.0	0.0	0	NaN
4	Total	43.0	39.000000	31.000000	0.0	0.0	113	0.000000
5	PosPred(%)	100.0	82.051282	90.322581	NaN	0.0	Acc(%)	91.150442

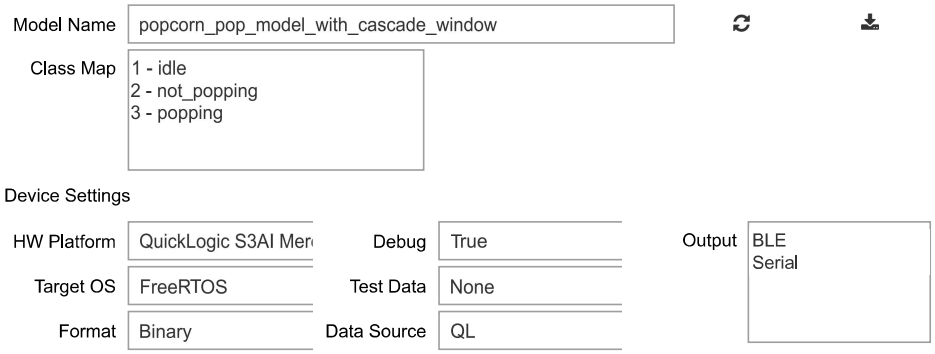
Save your model

In [8]:
```
model.knowledgepack.save('popcorn_pop_model_with_cascade_window')
```

```
Knowledgepack 'popcorn_pop_model_with_cascade_window' updated.
```

Download the Model

In [9]:
```
DownloadWidget(dsk).create_widget()
```



Knowledge Pack saved to C:\Users\mcetin\sensiml_notebook\PoC_popcorn_pop\knowledgepacks\kp_6e7161b8-9066-4cc8-8edc-4bf2defe333a_QuickLogic-S3AI-Merced_bin_1.3.1_d.zip

```
{'target_platform': 16, 'test_data': None, 'debug': True, 'application': 'mercedproto_
ai_app', 'sample_rate': '', 'output_options': ['ble', 'serial'], 'kb_description': {'p
opcorn_pop_model_with_cascade_window': {'uuid': '6e7161b8-9066-4cc8-8edc-4bf2defe333
a', 'results': {'1 ': 'Report', '2 ': 'Report', '3 ': 'Report'}, 'source': 'bd75f835-4
952-4cae-a976-fef420e6b1c8'}}}
Building Binary with configuration
target_platform : 16
test_data : None
debug : True
application : mercedproto_ai_app
sample_rate :
output_options : ['ble', 'serial']
kb_description : {"popcorn_pop_model_with_cascade_window": {"uuid": "6e7161b8-9066-4cc
8-8edc-4bf2defe333a", "results": {"1 ": "Report", "2 ": "Report", "3 ": "Report"}, "so
urce": "bd75f835-4952-4cae-a976-fef420e6b1c8"}}
...........KnowledgePack saved to
C:\Users\mcetin\sensiml_notebook\PoC_popcorn_pop\knowledgepacks\kp_6e7161b8-9066-4cc8-
8edc-4bf2defe333a_QuickLogic-S3AI-Merced_bin_1.3.1_d.zip
```

In [ ]:
```
```

Validating the results

Read the captured files

In [147]:
```
test_files = dsk.list_captures().Name.tolist()
test_files
```

Out[147]:
```
['not_popping_20190910T233941.csv',
 'not_popping_20190910T185948.csv',
 'not_popping_20190910T185600.csv',
 'not_popping_20190910T185227.csv']
```

Running recognize signal

Check the SegmentEnd column between the consequtive prediction. The difference between them is 80000 time points which refers to one prediction for each 5 sec of data.

PS: SegmentStart colum refers the last cascade of the Feature Cascades

In [148]:
```
rr,ss = model.recognize_signal(capturefile=test_files[1] )
rr
```

```
Status: Running, Time: 0.09, Executing recognize signal. ...
Status: Running, Time: 0.42, Executing recognize signal. .

Results Retrieved... Execution Time: 0 min. 35 sec.
Results retrieved.
```

Out[148]:

	Classification	ClassificationName	FeatureVector	ModelName	SegmentEnd	SegmentID	SegmentLength
0	1	idle	[4, 0, 0, 0, 1, 0, 0, 0, 2, 3]	0	80000	0	8000
1	2	not_popping	[182, 156, 156, 156, 152, 156, 156, 156, 154, ...	0	160000	1	8000
2	2	not_popping	[169, 169, 169, 169, 170, 169, 169, 169, 162, ...	0	240000	2	8000
3	2	not_popping	[190, 182, 182, 182, 189, 182, 182, 182, 173, ...	0	320000	3	8000
4	2	not_popping	[168, 183, 183, 183, 192, 183, 183, 183, 184, ...	0	400000	4	8000
5	2	not_popping	[171, 169, 169, 169, 178, 169, 169, 169, 176, ...	0	480000	5	8000
6	2	not_popping	[182, 167, 167, 167, 168, 167, 167, 167, 164, ...	0	560000	6	8000
7	2	not_popping	[175, 179, 179, 179, 185, 179, 179, 179, 174, ...	0	640000	7	8000
8	2	not_popping	[167, 181, 181, 181, 189, 181, 181, 181, 176, ...	0	720000	8	8000
9	3	popping	[210, 198, 198, 198, 214, 198, 198, 198, 203, ...	0	800000	9	8000
10	3	popping	[217, 219, 219, 219, 233, 219, 219, 219, 213, ...	0	880000	10	8000
11	3	popping	[202, 222, 222, 222, 235, 222, 222, 222, 211, ...	0	960000	11	8000
12	3	popping	[205, 206, 206, 206, 214, 206, 206, 206, 210, ...	0	1040000	12	8000
13	3	popping	[207, 215, 215, 215, 223, 215, 215, 215, 204, ...	0	1120000	13	8000
14	3	popping	[220, 246, 246, 246, 230, 246, 246, 246, 212, ...	0	1200000	14	8000
15	3	popping	[229, 239, 239, 239, 233, 239, 239, 239, 243, ...	0	1280000	15	8000
16	3	popping	[255, 242, 242, 242, 219, 242, 243, 242, 202, ...	0	1360000	16	8000
17	3	popping	[250, 223, 223, 223, 208, 223, 224, 223, 213, ...	0	1440000	17	8000

	Classification	ClassificationName	FeatureVector	ModelName	SegmentEnd	SegmentID	SegmentLength
18	3	popping	[226, 232, 232, 232, 208, 232, 232, 232, 209, ...	0	1520000	18	8000
19	3	popping	[192, 243, 243, 243, 224, 243, 243, 243, 213, ...	0	1600000	19	8000
20	2	not_popping	[194, 192, 192, 192, 199, 192, 193, 192, 188, ...	0	1680000	20	8000
21	2	not_popping	[202, 185, 185, 185, 198, 185, 184, 185, 184, ...	0	1760000	21	8000
22	2	not_popping	[196, 199, 199, 199, 206, 199, 197, 199, 194, ...	0	1840000	22	8000
23	2	not_popping	[188, 188, 188, 188, 192, 188, 188, 188, 196, ...	0	1920000	23	8000
24	2	not_popping	[183, 195, 195, 195, 202, 195, 195, 195, 187, ...	0	2000000	24	8000
25	2	not_popping	[206, 195, 195, 195, 199, 195, 195, 195, 187, ...	0	2080000	25	8000

In [ ]:

In [ ]:

Instruction for Flashing Knowledgepack to QuickLogic - Merced

1. Copy the knowledpack to QlFlashImage_Release_v1.3\images\mercedproto_ai_app_recog\IAR
2. EDIT the file: "flash_all_iar.json"
3. Find "MCU_M4.bin" tag in flash_all_iar.json file
4. Update "input" to knowledpack name
5. Power cycle JLINK + MERCED board.
6. In 3 secont, Execute batch file: "flash_via_jlink.bat com15" <- Replace COM15 with your comport.

In [14]:

In [ ]: