| _ | | | | | | | |
|----|----------|-----|-----|-----|-------|------------|--|
| 16 | EP01_05 | 66 | 106 | 156 | 4+R12 | happiness | |
| 16 | EP01_08 | 73 | 94 | 110 | 17 | repression | |
| 16 | EP01_09f | 51 | 76 | 163 | 4+12 | others | |
| 16 | EP04 02f | 181 | 269 | 271 | R12 | happiness | |

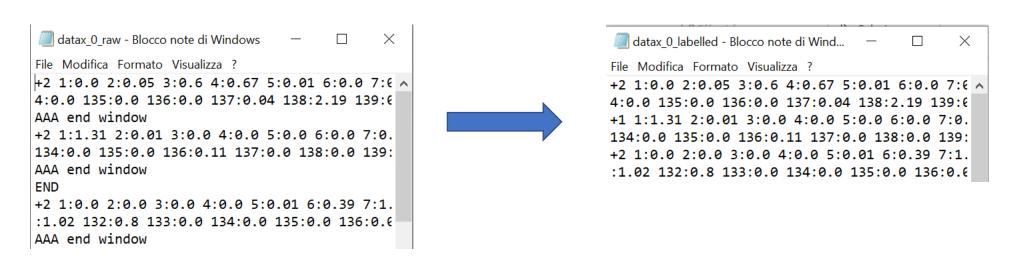
| AU number ♦ | FACS name \$ | Muscular basis ◆ |
|-------------|------------------------|--|
| 0 | Neutral face | |
| 1 | Inner brow raiser | frontalis (pars medialis) |
| 2 | Outer brow raiser | frontalis (pars lateralis) |
| 4 | Brow lowerer | depressor glabellae, depressor supercilii, corrugator supercilii |
| 5 | Upper lid raiser | levator palpebrae superioris, superior tarsal muscle |
| 6 | Cheek raiser | orbicularis oculi (pars orbitalis) |
| 7 | Lid tightener | orbicularis oculi (pars palpebralis) |
| 8 | Lips toward each other | orbicularis oris |
| 9 | Nose wrinkler | levator labii superioris alaeque nasi |
| 10 | Upper lip raiser | levator labii superioris, caput infraorbitalis |
| 11 | Nasolabial deepener | zygomaticus minor |
| 12 | Lip corner puller | zygomaticus major |
| 13 | Sharp lip puller | levator anguli oris (also known as caninus) |
| 14 | Dimpler | buccinator |
| 15 | Lip corner depressor | depressor anguli oris (also known as triangularis) |
| 16 | Lower lip depressor | depressor labii inferioris |
| 17 | Chin raiser | mentalis |

Supponiamo di concentrarci sul training del soggetto 16. A sinistra, la sezione del file excel casme coding relativa al soggetto 16; in seconda colonna abbiamo la sottocartella di riferimento mentre le colonne successive rappresentano l'onset, l'apex e l'offset frame della microespressione.

La penultima colonna indica l'AU che ci serve per individuare la regione del viso interessata dalla microespressione. Supponiamo di occuparci del file raw relativo alla bocca (datax_0), notiamo, grazie alla tabella delle AU qui a sinistra e consultabile online, che le microespressioni con AU 12 coinvolgono la bocca (quelle con AU 4 coinvolgono anche le sopracciglia e quindi dovranno essere etichettate negli altri file raw).

Supponiamo inoltre di adottare una soglia di true positive dell' **80%**, ovvero la finestra corrente contiene effettivamente un esempio positivo se contiene almeno l'80% dei frame della microespressione segnalati dal file excel. Vediamo adesso come etichettare correttamente il file.

| 16 | EP01_05 | 66 | 106 | 156 | 4+R12 | happiness |
|----|----------|-----|-----|-----|-------|------------|
| 16 | EP01_08 | 73 | 94 | 110 | 17 | repression |
| 16 | EP01_09f | 51 | 76 | 163 | 4+12 | others |
| 16 | EP04_02f | 181 | 269 | 271 | R12 | happiness |



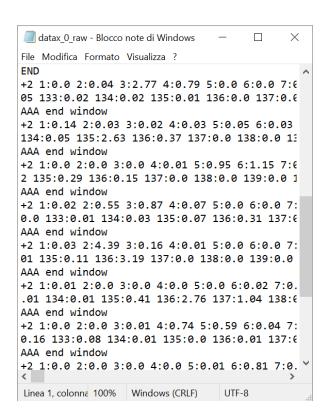
Sopra a sinistra abbiamo una sezione del file raw generato dall' esecuzione del programma mentre a destra una sezione del file correttamente etichettato e pronto ad essere usato per l'addestramento.

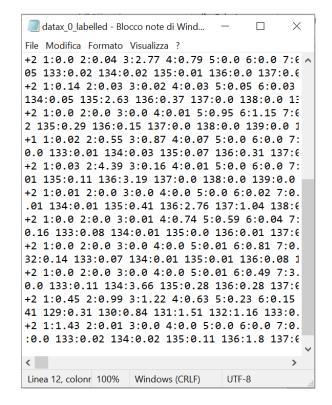
I descrittori sono in formato LibSVM per cui abbiamo un etichetta iniziale (+1 o +2) e una serie di parametri 1: x 2: x 3: x.... Come indicato, il carattere "AAA" indica la fine della finestra mentre "END" la fine della sottocartella.

In questo caso nella prima cartella sono individuate 2 finestre (una che và dal frame 0 a 100 e una da 60 a 160); come notiamo dalla tabella excel la microespressione della bocca della prima cartella è contenuta completamente nella seconda finestra e per soli 24/90 frame nella prima. Per la soglia imposta, la microespressione deve essere segnalata solo nella seconda finestra cambiando l'etichetta in +1; l'etichetta della prima finestra rimane invariata ma bisogna procedere a rimuovere i caratteri di delimitazione come è possibile vedere nel file etichettato.

Si procede poi con lo stesso ragionamento per la seconda sottocartella che avrà la prima finestra successiva al carattere END.

| 16 | EP01_05 | 66 | 106 | 156 | 4+R12 | happiness |
|----|----------|-----|-----|-----|-------|------------|
| 16 | EP01_08 | 73 | 94 | 110 | 17 | repression |
| 16 | EP01_09f | 51 | 76 | 163 | 4+12 | others |
| 16 | EP04_02f | 181 | 269 | 271 | R12 | happiness |





Altro esempio.

Siamo arrivati alla quarta sottocartella; questa contiene molte finestre in quanto il video è particolarmente lungo.

Abbiamo dunque le finestre 0-100, 60-160, 120-220, 180-280, 240-340 ecc..

La microespressione è completamente contenuta nella finestra 4 mentre ha 39/90 frame nella finestra 3 e 31/90 frame nella finestra 5.

Come possiamo vedere dal file etichettato solo la finestra 4 avrà etichetta +1.

Se si pone una soglia molto più bassa, ad esempio 40%, anche la finestra 3 dovrà avere etichetta +1.

Il file raw e quello etichettato sono consultabili inoltre per intero sempre nella cartella training_example.

Eseguito questo procedimento anche per i file raw relativi anche alle regioni delle sopracciglia e ripetendo per altri soggetti si possono combinare i vari esempi in 3 unici file che rappresentano i 3 dataset per la detection di ME di bocca, sopracciglia dx e sx (vedere i file DatasetX presenti in source).

Dato a questo punto uno di questi file (si può usare anche solo il file etichettato del soggetto 16 ma contiene solo 3 esempi positivi), si può inserire il path e chiamare la funzione SVM_model di svm.py.

Ex. con "datax 0" presente in source:

SVM_model("datax_0.txt")

Vengono poi generati i file in figura in basso; il file modello e il file range serviranno poi per le predizioni future.

| datax_0 | 30/07/2020 23:36 | Documento di testo | 23 KB |
|----------------------------|------------------|--------------------|-------|
| <pre>all datax_0.txt</pre> | 30/07/2020 23:36 | File MODEL | 27 KB |
| datax_0.txt | 30/07/2020 23:36 | File RANGE | 4 KB |
| datax_0.txt | 30/07/2020 23:36 | File SCALE | 28 KB |
| datax_0.txt.scale | 30/07/2020 23:36 | File OUT | 4 KB |

```
symscale_exe = r".\windows\svm-scale.exe"
symscale_exe = r".\windows\svm-predict.exe"

assert os.path.exists(svmscale_exe), "svm-scale executable not found"
assert os.path.exists(svmpredict_exe), "svm-predict executable not found"

testPath = "testData" + str(type) + ".txt"

file_name = "testData" + str(type)
model_file = "DatasetX" + str(type) + ".txt.model"
scaled_test_file = file_name + ".scale"
predict_test_file = file_name + ".predict"
range_file = "DatasetX" + str(type) + ".txt.range"

cmd = '{0} -r "{1}" "{2}" > "{3}"'.format(svmscale_exe, range_file, testPath, scaled_test_file)
fprint('Scaling_testing_data...')
Popen(cmd, shell=True, stdout=PIPE).communicate()
```

Supponendo di aver svolto il procedimento anche per datax_1 e datax_2 e di avere i file nella cartella source, se nella funzione SVM_predict sostituiamo "DatasetX_" con "datax_" il predict punterà ai nuovi modelli e le predizioni future saranno basate su questi.