Clasificación de cantos de pájaros (Mirlos, Arrendajos y Carboneros)

Francisco Rodríguez Cuenca

The audio samples were obtained from:

- https://www.xeno-canto.org

And the crawler was modified from this example:

- birdsonearth

Preprocessing stage

In [1]:
```
import librosa
from scipy.io import wavfile as wav
import numpy as np
```

Sample Rate

In [2]:
```
filename = 'data/10birds/Turdus/1352.wav'

librosa_audio, librosa_sample_rate = librosa.load(filename)
scipy_sample_rate, scipy_audio = wav.read(filename)

print('Original sample rate:', scipy_sample_rate)
print('Librosa sample rate:', librosa_sample_rate)
```

```
Original sample rate: 22050
Librosa sample rate: 22050
```

Bit-depth

In [3]:
```
print('Original audio file min~max range:', np.min(scipy_audio), 'to', np.
print('Librosa audio file min~max range:', np.min(librosa_audio), 'to', np
```

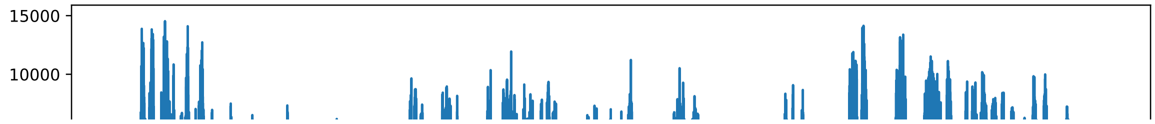
```
Original audio file min~max range: -12746 to 14535
Librosa audio file min~max range: -0.38897705 to 0.443573
```

Merge audio channels

In [4]:
```
import matplotlib.pyplot as plt

# Original audio with 2 channels
plt.figure(figsize=(12, 4))
plt.plot(scipy_audio)
```

Out[4]:
```
[<matplotlib.lines.Line2D at 0x7fb4081f28d0>]
```

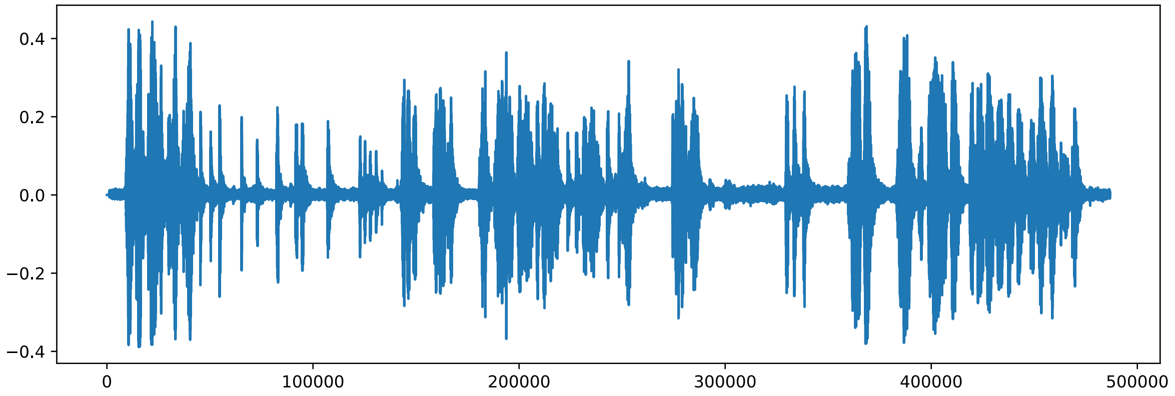


In [5]:

```
# Librosa audio with channels merged
plt.figure(figsize=(12, 4))
plt.plot(librosa_audio)
```

Out[5]:

```
[<matplotlib.lines.Line2D at 0x7fb3eb16a510>]
```



Extract Features

Extracting a MFCC

In [6]:

```
mfccs = librosa.feature.mfcc(y=librosa_audio, sr=librosa_sample_rate, n_mf
print(mfccs.shape) #That's 173 samples and 40 Mel-frequency cepstral coeff
```

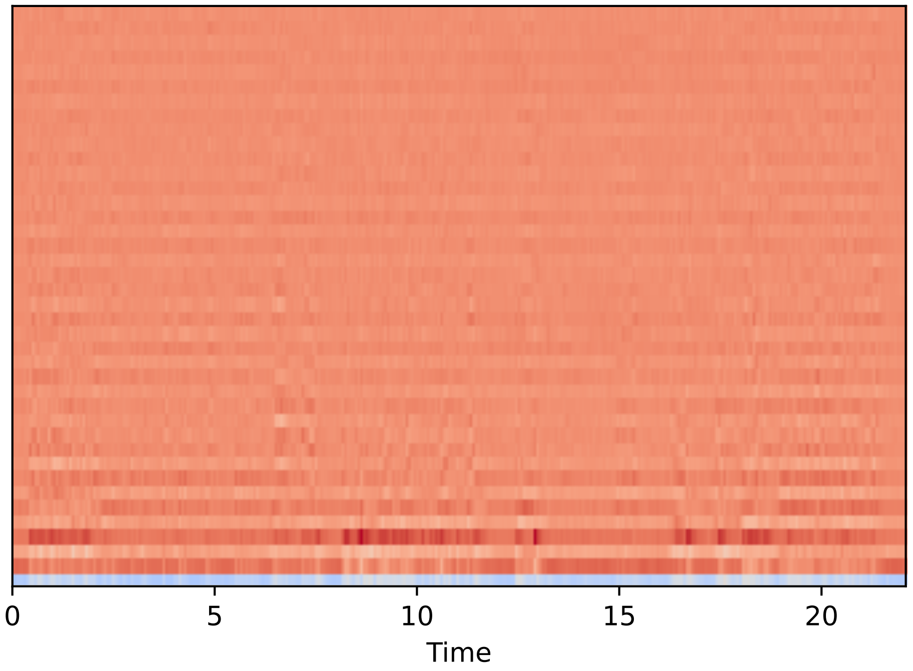
```
(40, 951)
```

In [7]:

```
import librosa.display
librosa.display.specshow(mfccs, sr=librosa_sample_rate, x_axis='time')
```

Out[7]:

```
<matplotlib.collections.QuadMesh at 0x7fb3ea8f74d0>
```



Extracting MFCC's for every file

In [8]:
```
import pandas as pd
import os
import librosa
from glob import glob
```

In [9]:
```
def extract_features(file_name):

    try:
        audio, sample_rate = librosa.load(file_name, res_type='kaiser_fast
        mfccs = librosa.feature.mfcc(y=audio, sr=sample_rate, n_mfcc=40)
        mfccsscaled = np.mean(mfccs.T,axis=0)

    except Exception as e:
        print("Error encountered while parsing file: ", file)
        return None

    return mfccsscaled
```

In [10]:
```
carbonero = glob(pathname="data/10birds/Parus/*")
mirlo = glob(pathname="data/10birds/Turdus/*")
arrendajo = glob(pathname="data/10birds/Garrulus/*")
```

In [11]:
```
data = (
    list(map(lambda path : ("Carbonero", extract_features(path)), carboner
    list(map(lambda path : ("Mirlo", extract_features(path)), mirlo)) +
    list(map(lambda path : ("Arrendajo", extract_features(path)), arrendaj
)
```

```
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/librosa/core/audio.py:162: UserWarning: PySoundFile faile
d. Trying audioread instead.
  warnings.warn("PySoundFile failed. Trying audioread instead.")
```

In [12]:
```
featuresdf = pd.DataFrame(data, columns= ["class_label", "feature"], )
featuresdf
```

Out[12]:

	class_label	feature
0	Carbonero	[-512.55927, 45.171978, 11.400741, 44.23818, 5...
1	Carbonero	[-588.4902, 1.3267289, -3.036135, 9.091025, -1...
2	Carbonero	[-341.96182, 99.5176, -15.277686, 10.666541, -...
3	Carbonero	[-307.98322, 18.405989, -14.905345, -1.4368179...
4	Carbonero	[-448.7659, 19.839348, -12.141717, 30.371658, ...
...	...	...
1436	Arrendajo	[-379.0589, 72.06425, -43.798336, -23.38201, -...
1437	Arrendajo	[-302.92432, 60.98192, -43.70341, 9.114791, -1...
1438	Arrendajo	[-332.45734, 54.379097, -35.348785, 7.796014, ...
1439	Arrendajo	[-403.32278, -75.939156, -75.14334, -4.367598,...
1440	Arrendajo	[-367.9945, -19.263664, -73.50193, -66.882675,...

1441 rows × 2 columns

In [13]:
```
featuresdf.class_label.unique()
```

Out[13]:
```
array(['Carbonero', 'Mirlo', 'Arrendajo'], dtype=object)
```

Convert the data and labels

In [14]:
```
from sklearn.preprocessing import LabelEncoder
from keras.utils import to_categorical

# Convert features and corresponding classification labels into numpy array
X = np.array(featuresdf.feature.tolist())
y = np.array(featuresdf.class_label.tolist())

# Encode the classification labels
le = LabelEncoder()
yy = to_categorical(le.fit_transform(y))
```

Split the dataset

In [15]:
```
# split the dataset
from sklearn.model_selection import train_test_split

x_train, x_test, y_train, y_test = train_test_split(X, yy, test_size=0.2,
```

Initial model architecture - MLP

In [16]:
```
import numpy as np
from keras.models import Sequential
from keras.layers import Dense, Dropout, Activation, Flatten
from keras.layers import Convolution2D, MaxPooling2D
from keras.optimizers import Adam
from keras.utils import np_utils
from sklearn import metrics

num_labels = yy.shape[1]
filter_size = 2

# Construct model
model = Sequential()

model.add(Dense(256, input_shape=(40,)))
model.add(Activation('relu')) #sigmoid
model.add(Dropout(0.5))

model.add(Dense(256))
model.add(Activation('relu')) #sigmoid
model.add(Dropout(0.5))

model.add(Dense(num_labels))
model.add(Activation('softmax'))
```

Compiling the model

In [17]:
```
# Compile the model
model.compile(loss='categorical_crossentropy', metrics=['accuracy'], optim
```

In [18]:

```
# Display model architecture summary
model.summary()

# Calculate pre-training accuracy
score = model.evaluate(x_test, y_test, verbose=0)
accuracy = 100*score[1]

print("Pre-training accuracy: %.4f%%" % accuracy)
```

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #   
=================================================================
dense (Dense)                (None, 256)               10496     
_________________________________________________________________
activation (Activation)      (None, 256)               0         
_________________________________________________________________
dropout (Dropout)            (None, 256)               0         
_________________________________________________________________
dense_1 (Dense)              (None, 256)               65792     
_________________________________________________________________
activation_1 (Activation)    (None, 256)               0         
_________________________________________________________________
dropout_1 (Dropout)          (None, 256)               0         
_________________________________________________________________
dense_2 (Dense)              (None, 3)                 771       
_________________________________________________________________
activation_2 (Activation)    (None, 3)                 0         
=================================================================
Total params: 77,059
Trainable params: 77,059
Non-trainable params: 0
_________________________________________________________________
Pre-training accuracy: 33.2180%
```

Training

In [19]:

```
from keras.callbacks import ModelCheckpoint
from datetime import datetime

num_epochs = 100
num_batch_size = 32

checkpointer = ModelCheckpoint(filepath='saved_models/weights.best.basic_ml
                               verbose=1, save_best_only=True)
start = datetime.now()

history = model.fit(x_train, y_train, batch_size=num_batch_size, epochs=nu


duration = datetime.now() - start
print("Training completed in time: ", duration)
```

```
s 2ms/step - loss: 1.1858 - accuracy: 0.4288 - val_loss: 1.0944 - val_accur
acy: 0.3426

Epoch 00010: val_loss did not improve from 1.04967
Epoch 11/100
36/36 [==============================] - 0s 2ms/step - loss: 1.1745 - accur
acy: 0.4097 - val_loss: 1.0953 - val_accuracy: 0.3460

Epoch 00011: val_loss did not improve from 1.04967
Epoch 12/100
```

```
36/36 [==============================] - 0s 2ms/step - loss: 1.1567 - accur
acy: 0.4505 - val_loss: 1.0868 - val_accuracy: 0.3426

Epoch 00012: val_loss did not improve from 1.04967
Epoch 13/100
36/36 [==============================] - 0s 2ms/step - loss: 1.1450 - accur
acy: 0.4193 - val_loss: 1.0843 - val_accuracy: 0.3841

Epoch 00013: val_loss did not improve from 1.04967
Epoch 14/100
36/36 [==============================] - 0s 2ms/step - loss: 1.1137 - accur
acy: 0.4210 - val_loss: 1.0863 - val_accuracy: 0.3772

Epoch 00014: val_loss did not improve from 1.04967
Epoch 15/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0602 - accur
acy: 0.4444 - val_loss: 1.0697 - val_accuracy: 0.4567

Epoch 00015: val_loss did not improve from 1.04967
Epoch 16/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0909 - accur
acy: 0.4757 - val_loss: 1.0536 - val_accuracy: 0.4567

Epoch 00016: val_loss did not improve from 1.04967
Epoch 17/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0634 - accur
acy: 0.4696 - val_loss: 1.0614 - val_accuracy: 0.4567

Epoch 00017: val_loss did not improve from 1.04967
Epoch 18/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0586 - accur
acy: 0.4965 - val_loss: 1.0343 - val_accuracy: 0.4983

Epoch 00018: val_loss improved from 1.04967 to 1.03435, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 19/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0230 - accur
acy: 0.5052 - val_loss: 1.0142 - val_accuracy: 0.4983

Epoch 00019: val_loss improved from 1.03435 to 1.01419, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 20/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0226 - accur
acy: 0.4722 - val_loss: 1.0011 - val_accuracy: 0.5190

Epoch 00020: val_loss improved from 1.01419 to 1.00114, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 21/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0301 - accur
acy: 0.4983 - val_loss: 0.9946 - val_accuracy: 0.5363

Epoch 00021: val_loss improved from 1.00114 to 0.99464, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 22/100
36/36 [==============================] - 0s 2ms/step - loss: 1.0217 - accur
acy: 0.4991 - val_loss: 0.9836 - val_accuracy: 0.5536

Epoch 00022: val_loss improved from 0.99464 to 0.98359, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 23/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9815 - accur
acy: 0.5156 - val_loss: 0.9678 - val_accuracy: 0.5813

Epoch 00023: val_loss improved from 0.98359 to 0.96778, saving model to sav
```

```
ed_models/weights.best.basic_mlp.hdf5
Epoch 24/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9877 - accur
acy: 0.5113 - val_loss: 0.9502 - val_accuracy: 0.5502

Epoch 00024: val_loss improved from 0.96778 to 0.95025, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 25/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9700 - accur
acy: 0.5295 - val_loss: 0.9304 - val_accuracy: 0.5882

Epoch 00025: val_loss improved from 0.95025 to 0.93036, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 26/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9563 - accur
acy: 0.5339 - val_loss: 0.9305 - val_accuracy: 0.5882

Epoch 00026: val_loss did not improve from 0.93036
Epoch 27/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9227 - accur
acy: 0.5477 - val_loss: 0.9120 - val_accuracy: 0.5779

Epoch 00027: val_loss improved from 0.93036 to 0.91203, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 28/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9450 - accur
acy: 0.5512 - val_loss: 0.9133 - val_accuracy: 0.5848

Epoch 00028: val_loss did not improve from 0.91203
Epoch 29/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9156 - accur
acy: 0.5668 - val_loss: 0.8948 - val_accuracy: 0.5779

Epoch 00029: val_loss improved from 0.91203 to 0.89482, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 30/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9331 - accur
acy: 0.5538 - val_loss: 0.9062 - val_accuracy: 0.5433

Epoch 00030: val_loss did not improve from 0.89482
Epoch 31/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9192 - accur
acy: 0.5651 - val_loss: 0.8914 - val_accuracy: 0.5882

Epoch 00031: val_loss improved from 0.89482 to 0.89142, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 32/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9133 - accur
acy: 0.5920 - val_loss: 0.8649 - val_accuracy: 0.5848

Epoch 00032: val_loss improved from 0.89142 to 0.86493, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 33/100
36/36 [==============================] - 0s 2ms/step - loss: 0.9138 - accur
acy: 0.5807 - val_loss: 0.8746 - val_accuracy: 0.5848

Epoch 00033: val_loss did not improve from 0.86493
Epoch 34/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8894 - accur
acy: 0.5885 - val_loss: 0.8697 - val_accuracy: 0.5986

Epoch 00034: val_loss did not improve from 0.86493
Epoch 35/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8832 - accur
```

```
acy: 0.5920 - val_loss: 0.8630 - val_accuracy: 0.5675

Epoch 00035: val_loss improved from 0.86493 to 0.86301, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 36/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8737 - accur
acy: 0.5920 - val_loss: 0.8546 - val_accuracy: 0.5779

Epoch 00036: val_loss improved from 0.86301 to 0.85462, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 37/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8696 - accur
acy: 0.5972 - val_loss: 0.8251 - val_accuracy: 0.5848

Epoch 00037: val_loss improved from 0.85462 to 0.82514, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 38/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8687 - accur
acy: 0.6007 - val_loss: 0.8370 - val_accuracy: 0.5779

Epoch 00038: val_loss did not improve from 0.82514
Epoch 39/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8677 - accur
acy: 0.6233 - val_loss: 0.8148 - val_accuracy: 0.6021

Epoch 00039: val_loss improved from 0.82514 to 0.81475, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 40/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8430 - accur
acy: 0.6267 - val_loss: 0.8328 - val_accuracy: 0.6090

Epoch 00040: val_loss did not improve from 0.81475
Epoch 41/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8504 - accur
acy: 0.6241 - val_loss: 0.8061 - val_accuracy: 0.6159

Epoch 00041: val_loss improved from 0.81475 to 0.80609, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 42/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8252 - accur
acy: 0.6450 - val_loss: 0.8116 - val_accuracy: 0.6055

Epoch 00042: val_loss did not improve from 0.80609
Epoch 43/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8051 - accur
acy: 0.6276 - val_loss: 0.7988 - val_accuracy: 0.6401

Epoch 00043: val_loss improved from 0.80609 to 0.79875, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 44/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8376 - accur
acy: 0.6207 - val_loss: 0.7930 - val_accuracy: 0.6436

Epoch 00044: val_loss improved from 0.79875 to 0.79300, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 45/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8086 - accur
acy: 0.6562 - val_loss: 0.7924 - val_accuracy: 0.6401

Epoch 00045: val_loss improved from 0.79300 to 0.79240, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 46/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8014 - accur
acy: 0.6293 - val_loss: 0.7811 - val_accuracy: 0.6436
```

```
Epoch 00046: val_loss improved from 0.79240 to 0.78114, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 47/100
36/36 [==============================] - 0s 2ms/step - loss: 0.8115 - accur
acy: 0.6484 - val_loss: 0.7868 - val_accuracy: 0.6298

Epoch 00047: val_loss did not improve from 0.78114
Epoch 48/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7901 - accur
acy: 0.6571 - val_loss: 0.7800 - val_accuracy: 0.6436

Epoch 00048: val_loss improved from 0.78114 to 0.78004, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 49/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7903 - accur
acy: 0.6432 - val_loss: 0.7856 - val_accuracy: 0.6125

Epoch 00049: val_loss did not improve from 0.78004
Epoch 50/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7906 - accur
acy: 0.6528 - val_loss: 0.7927 - val_accuracy: 0.6194

Epoch 00050: val_loss did not improve from 0.78004
Epoch 51/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7618 - accur
acy: 0.6788 - val_loss: 0.7714 - val_accuracy: 0.6332

Epoch 00051: val_loss improved from 0.78004 to 0.77142, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 52/100
36/36 [==============================] - 0s 3ms/step - loss: 0.7625 - accur
acy: 0.6797 - val_loss: 0.7749 - val_accuracy: 0.6401

Epoch 00052: val_loss did not improve from 0.77142
Epoch 53/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7738 - accur
acy: 0.6510 - val_loss: 0.7623 - val_accuracy: 0.6298

Epoch 00053: val_loss improved from 0.77142 to 0.76233, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 54/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7357 - accur
acy: 0.6814 - val_loss: 0.7635 - val_accuracy: 0.6367

Epoch 00054: val_loss did not improve from 0.76233
Epoch 55/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7452 - accur
acy: 0.6927 - val_loss: 0.7549 - val_accuracy: 0.6609

Epoch 00055: val_loss improved from 0.76233 to 0.75488, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 56/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7546 - accur
acy: 0.6727 - val_loss: 0.7572 - val_accuracy: 0.6678

Epoch 00056: val_loss did not improve from 0.75488
Epoch 57/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7422 - accur
acy: 0.6762 - val_loss: 0.7541 - val_accuracy: 0.6505

Epoch 00057: val_loss improved from 0.75488 to 0.75407, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 58/100
```

```
36/36 [==============================] - 0s 2ms/step - loss: 0.7490 - accur
acy: 0.6918 - val_loss: 0.7414 - val_accuracy: 0.6401

Epoch 00058: val_loss improved from 0.75407 to 0.74136, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 59/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7437 - accur
acy: 0.6719 - val_loss: 0.7505 - val_accuracy: 0.6540

Epoch 00059: val_loss did not improve from 0.74136
Epoch 60/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7621 - accur
acy: 0.6710 - val_loss: 0.7412 - val_accuracy: 0.6747

Epoch 00060: val_loss improved from 0.74136 to 0.74116, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 61/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7212 - accur
acy: 0.6979 - val_loss: 0.7322 - val_accuracy: 0.6540

Epoch 00061: val_loss improved from 0.74116 to 0.73225, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 62/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7319 - accur
acy: 0.6953 - val_loss: 0.7403 - val_accuracy: 0.6678

Epoch 00062: val_loss did not improve from 0.73225
Epoch 63/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7145 - accur
acy: 0.6936 - val_loss: 0.7503 - val_accuracy: 0.6574

Epoch 00063: val_loss did not improve from 0.73225
Epoch 64/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7222 - accur
acy: 0.6858 - val_loss: 0.7444 - val_accuracy: 0.6540

Epoch 00064: val_loss did not improve from 0.73225
Epoch 65/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7303 - accur
acy: 0.6849 - val_loss: 0.7395 - val_accuracy: 0.6505

Epoch 00065: val_loss did not improve from 0.73225
Epoch 66/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7010 - accur
acy: 0.7031 - val_loss: 0.7494 - val_accuracy: 0.6609

Epoch 00066: val_loss did not improve from 0.73225
Epoch 67/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6742 - accur
acy: 0.7101 - val_loss: 0.7349 - val_accuracy: 0.6644

Epoch 00067: val_loss did not improve from 0.73225
Epoch 68/100
36/36 [==============================] - 0s 2ms/step - loss: 0.7210 - accur
acy: 0.6988 - val_loss: 0.7316 - val_accuracy: 0.6644

Epoch 00068: val_loss improved from 0.73225 to 0.73161, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 69/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6749 - accur
acy: 0.7153 - val_loss: 0.7321 - val_accuracy: 0.6678

Epoch 00069: val_loss did not improve from 0.73161
Epoch 70/100
```

```
36/36 [==============================] - 0s 2ms/step - loss: 0.6951 - accur
acy: 0.6944 - val_loss: 0.7225 - val_accuracy: 0.6782

Epoch 00070: val_loss improved from 0.73161 to 0.72251, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 71/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6680 - accur
acy: 0.7127 - val_loss: 0.7334 - val_accuracy: 0.6713

Epoch 00071: val_loss did not improve from 0.72251
Epoch 72/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6826 - accur
acy: 0.7405 - val_loss: 0.7324 - val_accuracy: 0.6644

Epoch 00072: val_loss did not improve from 0.72251
Epoch 73/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6515 - accur
acy: 0.7266 - val_loss: 0.7257 - val_accuracy: 0.6574

Epoch 00073: val_loss did not improve from 0.72251
Epoch 74/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6602 - accur
acy: 0.7326 - val_loss: 0.7211 - val_accuracy: 0.6678

Epoch 00074: val_loss improved from 0.72251 to 0.72114, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 75/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6413 - accur
acy: 0.7335 - val_loss: 0.7344 - val_accuracy: 0.6817

Epoch 00075: val_loss did not improve from 0.72114
Epoch 76/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6786 - accur
acy: 0.7352 - val_loss: 0.7195 - val_accuracy: 0.6886

Epoch 00076: val_loss improved from 0.72114 to 0.71952, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 77/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6712 - accur
acy: 0.7109 - val_loss: 0.7190 - val_accuracy: 0.6817

Epoch 00077: val_loss improved from 0.71952 to 0.71899, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 78/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6521 - accur
acy: 0.7214 - val_loss: 0.7245 - val_accuracy: 0.6782

Epoch 00078: val_loss did not improve from 0.71899
Epoch 79/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6491 - accur
acy: 0.7309 - val_loss: 0.7314 - val_accuracy: 0.6747

Epoch 00079: val_loss did not improve from 0.71899
Epoch 80/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6733 - accur
acy: 0.7153 - val_loss: 0.7091 - val_accuracy: 0.6747

Epoch 00080: val_loss improved from 0.71899 to 0.70914, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 81/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6514 - accur
acy: 0.7274 - val_loss: 0.7188 - val_accuracy: 0.6782

Epoch 00081: val_loss did not improve from 0.70914
Epoch 82/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6652 - accur
acy: 0.7240 - val_loss: 0.7159 - val_accuracy: 0.6886

Epoch 00082: val_loss did not improve from 0.70914
Epoch 83/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6349 - accur
acy: 0.7352 - val_loss: 0.7122 - val_accuracy: 0.6955

Epoch 00083: val_loss did not improve from 0.70914
Epoch 84/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6623 - accur
acy: 0.7205 - val_loss: 0.7153 - val_accuracy: 0.6920

Epoch 00084: val_loss did not improve from 0.70914
Epoch 85/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6298 - accur
acy: 0.7378 - val_loss: 0.7192 - val_accuracy: 0.6747

Epoch 00085: val_loss did not improve from 0.70914
Epoch 86/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6129 - accur
acy: 0.7378 - val_loss: 0.7143 - val_accuracy: 0.6747

Epoch 00086: val_loss did not improve from 0.70914
Epoch 87/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6139 - accur
acy: 0.7457 - val_loss: 0.7246 - val_accuracy: 0.6747

Epoch 00087: val_loss did not improve from 0.70914
Epoch 88/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6388 - accur
acy: 0.7318 - val_loss: 0.7231 - val_accuracy: 0.6920

Epoch 00088: val_loss did not improve from 0.70914
Epoch 89/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6325 - accur
acy: 0.7292 - val_loss: 0.7244 - val_accuracy: 0.6678

Epoch 00089: val_loss did not improve from 0.70914
Epoch 90/100
36/36 [==============================] - 0s 2ms/step - loss: 0.5895 - accur
acy: 0.7613 - val_loss: 0.7089 - val_accuracy: 0.6886

Epoch 00090: val_loss improved from 0.70914 to 0.70887, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 91/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6303 - accur
acy: 0.7387 - val_loss: 0.7055 - val_accuracy: 0.7024

Epoch 00091: val_loss improved from 0.70887 to 0.70551, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 92/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6083 - accur
acy: 0.7422 - val_loss: 0.6975 - val_accuracy: 0.6713

Epoch 00092: val_loss improved from 0.70551 to 0.69750, saving model to sav
ed_models/weights.best.basic_mlp.hdf5
Epoch 93/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6257 - accur
acy: 0.7405 - val_loss: 0.7170 - val_accuracy: 0.6782

Epoch 00093: val_loss did not improve from 0.69750
Epoch 94/100
```

```
36/36 [==============================] - 0s 2ms/step - loss: 0.6255 - accur
acy: 0.7483 - val_loss: 0.7062 - val_accuracy: 0.7024

Epoch 00094: val_loss did not improve from 0.69750
Epoch 95/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6211 - accur
acy: 0.7457 - val_loss: 0.7048 - val_accuracy: 0.6920

Epoch 00095: val_loss did not improve from 0.69750
Epoch 96/100
36/36 [==============================] - 0s 2ms/step - loss: 0.6129 - accur
acy: 0.7535 - val_loss: 0.7094 - val_accuracy: 0.6851

Epoch 00096: val_loss did not improve from 0.69750
Epoch 97/100
36/36 [==============================] - 0s 2ms/step - loss: 0.5899 - accur
acy: 0.7622 - val_loss: 0.7075 - val_accuracy: 0.6920

Epoch 00097: val_loss did not improve from 0.69750
Epoch 98/100
36/36 [==============================] - 0s 2ms/step - loss: 0.5901 - accur
acy: 0.7517 - val_loss: 0.7010 - val_accuracy: 0.6955

Epoch 00098: val_loss did not improve from 0.69750
Epoch 99/100
36/36 [==============================] - 0s 2ms/step - loss: 0.5911 - accur
acy: 0.7405 - val_loss: 0.7127 - val_accuracy: 0.6955

Epoch 00099: val_loss did not improve from 0.69750
Epoch 100/100
36/36 [==============================] - 0s 2ms/step - loss: 0.5827 - accur
acy: 0.7648 - val_loss: 0.7069 - val_accuracy: 0.6990

Epoch 00100: val_loss did not improve from 0.69750
```

Test the model

In [20]:

```
# Evaluating the model on the training and testing set
score = model.evaluate(x_train, y_train, verbose=0)
print("Training Accuracy: ", score[1])

score = model.evaluate(x_test, y_test, verbose=0)
print("Testing Accuracy: ", score[1])
```

```
Training Accuracy:  0.8133680820465088
Testing Accuracy:  0.6989619135856628
```

Validation

Test with sample data

Examples with some sample data downloaded from youtube.

In [21]:

```
def extract_feature(file_name):

    try:
        audio_data, sample_rate = librosa.load(file_name, res_type='kaiser_
        mfccs = librosa.feature.mfcc(y=audio_data, sr=sample_rate, n_mfcc=4
        mfccsscaled = np.mean(mfccs.T,axis=0)

    except Exception as e:
        print("Error encountered while parsing file: ", file)
        return None, None

    return np.array([mfccsscaled])
```

In [22]:

```
def print_prediction(file_name):
    prediction_feature = extract_feature(file_name)

    predicted_vector = model.predict_classes(prediction_feature)
    predicted_class = le.inverse_transform(predicted_vector)
    print("The predicted class is:", predicted_class[0], '\n')

    predicted_proba_vector = model.predict_proba(prediction_feature)
    predicted_proba = predicted_proba_vector[0]
    for i in range(len(predicted_proba)):
        category = le.inverse_transform(np.array([i]))
        print(category[0], "\t\t : ", format(predicted_proba[i], '.32f') )
```

In [23]:

```
# Class: Mirlo

filename = 'data/test/Especial Mirlo o Turdus + Canto adalaves.mp3'
print_prediction(filename)
```

```
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/librosa/core/audio.py:162: UserWarning: PySoundFile faile
d. Trying audioread instead.
  warnings.warn("PySoundFile failed. Trying audioread instead.")
The predicted class is: Mirlo

Arrendajo                :  0.41962343454360961914062500000000
Carbonero                :  0.09376125782728195190429687500000
Mirlo            :  0.48661521077156066894531250000000
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWar
ning: `model.predict_classes()` is deprecated and will be removed after 202
1-01-01. Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if
your model does multi-class classification   (e.g. if it uses a `softmax` l
ast-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-lay
er activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWar
ning: `model.predict_proba()` is deprecated and will be removed after 2021-
01-01. Please use `model.predict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
```

Correct!!

In [24]:

```
# Class: Arrendajo

filename = 'data/test/Arrendajo común ( Garrulus glandarius ) Jay.mp3'
print_prediction(filename)
```

```
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/librosa/core/audio.py:162: UserWarning: PySoundFile faile
d. Trying audioread instead.
  warnings.warn("PySoundFile failed. Trying audioread instead.")
The predicted class is: Mirlo

Arrendajo                :  0.30180105566978454589843750000000
Carbonero                :  0.04473211243748664855957031250000
Mirlo            :  0.65346688032150268554687500000000
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWar
ning: `model.predict_classes()` is deprecated and will be removed after 202
1-01-01. Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if
your model does multi-class classification   (e.g. if it uses a `softmax` l
ast-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-lay
er activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWar
ning: `model.predict_proba()` is deprecated and will be removed after 2021-
01-01. Please use `model.predict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
```

Almost correct :(

In [25]:

```python
# Class: Carbonero

filename = 'data/test/Canto del Carbonero.mp3'
print_prediction(filename)
```

```
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/librosa/core/audio.py:162: UserWarning: PySoundFile faile
d. Trying audioread instead.
  warnings.warn("PySoundFile failed. Trying audioread instead.")
The predicted class is: Carbonero

Arrendajo                :  0.12400624155998229980468750000000
Carbonero                :  0.68368899822235107421875000000000
Mirlo            :  0.19230476021766662597656250000000
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/tensorflow/python/keras/engine/sequential.py:450: UserWar
ning: `model.predict_classes()` is deprecated and will be removed after 202
1-01-01. Please use instead:* `np.argmax(model.predict(x), axis=-1)`,   if
your model does multi-class classification   (e.g. if it uses a `softmax` l
ast-layer activation).* `(model.predict(x) > 0.5).astype("int32")`,   if yo
ur model does binary classification   (e.g. if it uses a `sigmoid` last-lay
er activation).
  warnings.warn('`model.predict_classes()` is deprecated and '
/home/fran/Documents/MBD/SegundoCuatrimestre/ML/DeepLearning/env/lib/python
3.7/site-packages/tensorflow/python/keras/engine/sequential.py:425: UserWar
ning: `model.predict_proba()` is deprecated and will be removed after 2021-
01-01. Please use `model.predict()` instead.
  warnings.warn('`model.predict_proba()` is deprecated and '
```

Correct! :)

Conclusions

The model is clearly not perfect, although it has not been overtrained. With an 81% accuracy in training and 70 % in test, I think it is safe to say that maybe it would be advisable to recollect more samples to improve the results.

Another problem is that many of the samples are mixed with other similar bird calls, because they have been recorded in the wild, that could be another path for further improvement, the isolation of the sounds so the model can be trained better.