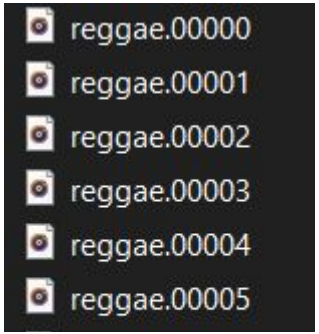
Audio Classification Analysis

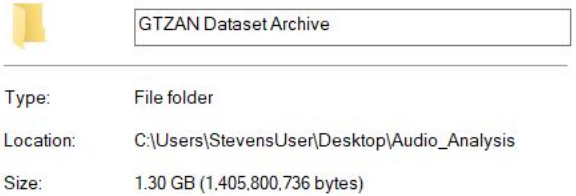
Roman Bellisari

The Task

- Identify and classify audio samples using a neural network architecture
- With 10 possible targets (blues, classical…), output a probabilistic classification using a Multi Layer Perceptron (MLP)
- Choose the target genre with the highest probability



The Dataset



- Marsyas (Music Analysis, Retrieval and Synthesis for Audio Signals)
- GTZAN Genre Collection - George Tzanetakis
- ‘The dataset consists of 1000 audio tracks each 30 seconds long. It contains 10 genres, each represented by 100 tracks. The tracks are all 22050Hz Mono 16-bit audio files in .wav format’
- Approximately 1.2GB
- Download the GTZAN music/speech collection

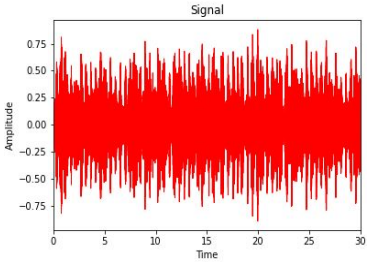
Libraries Used

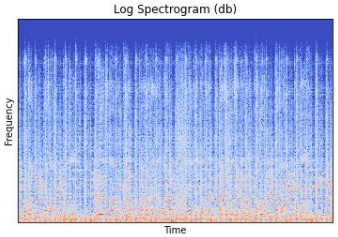
- **Numpy**
- Pandas
- JSON
- **Keras**/TensorFlow
- Scikit Learn (Train/Test Split)
- Math
- **Librosa**
- Matplotlib.pyplot
- VisualKeras
- OS
- Datetime

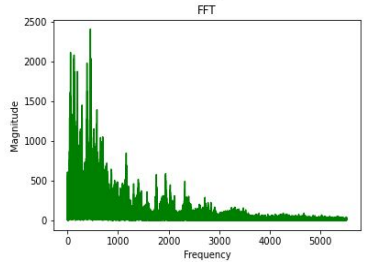
Methodology

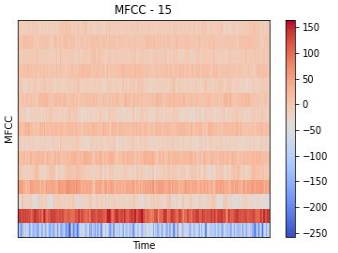
- Initial data analysis, become comfortable with Librosa library and Marsyas dataset
- Process the entire dataset to extract useful information that the computer can analyze (MFCC vectors)
- Save this processed data down as .json or .csv
- Load this data and split into training and testing data (0.25 split)
- Construct MLP model based on training data, use testing data to evaluate performance on unseen data
- Evaluate results, tweak model by updating parameters, repeat above step
- Make predictions on unseen data

Pre Processing





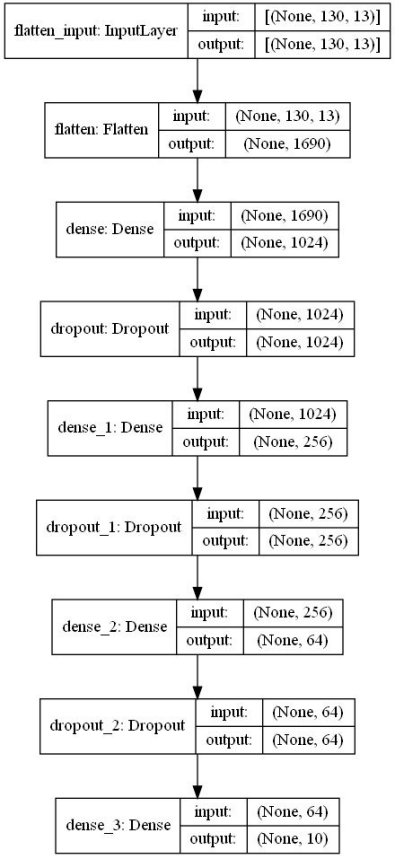




- For each genre, for each file in genre, for each sample in file, extract MFCC vectors and save down to JSON/CSV file
- MFCC - Mel Frequency Cepstrum Coefficients

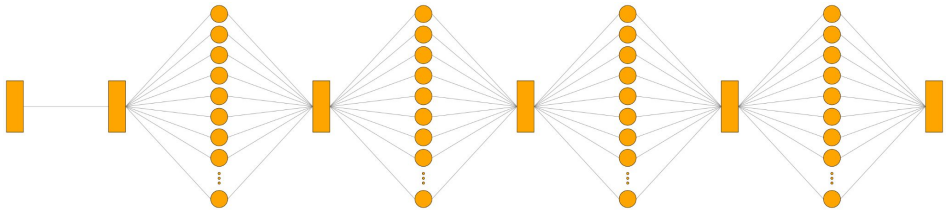

```
{
    "mapping": [
        "blues",
        "classical",
        "country",
        "disco",
        "hiphop",
        "jazz",
        "metal",
        "pop",
        "reggae",
        "rock"
    ],
    "labels": [
        0,
        0,
        0,
        0,
        0,
        0,
```


	A	B	C	D	E	F
1	MFCC_Values	Genre values	Genre Labels	Filename	Segment	MFCC Count
2	-212.6855316	1	blues	blues.00000.wav	0	1
3	100.718605	1	blues	blues.00000.wav	0	1
4	-12.32228279	1	blues	blues.00000.wav	0	1
5	40.39634705	1	blues	blues.00000.wav	0	1
6	1.239256382	1	blues	blues.00000.wav	0	1
7	21.78487015	1	blues	blues.00000.wav	0	1
8	-21.45558929	1	blues	blues.00000.wav	0	1
9	4.364691734	1	blues	blues.00000.wav	0	1
10	-9.350252151	1	blues	blues.00000.wav	0	1

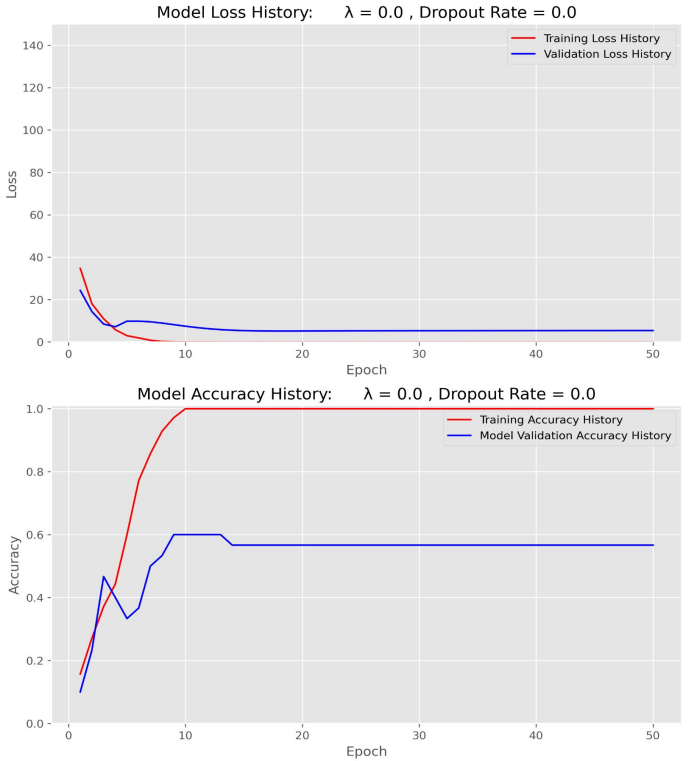


Multi Layer Perceptron Compilation

- One input layer
- 3 fully connected layers (1024, 256, 64)
- One output layers
- keras.optimizers.Adam(learning_rate=**0.0005**)
- model.fit(X_train, y_train, validation_data=(X_test, y_test), batch_size = **750**, epochs=**100**)



Overfitting

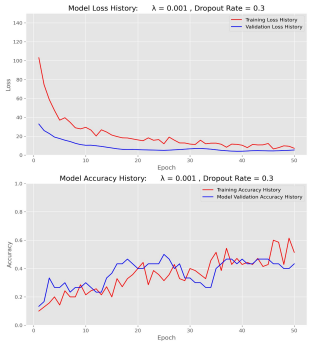


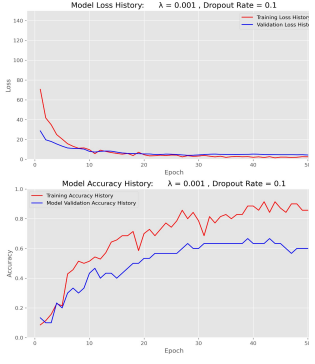
Tuning Hyperparameters

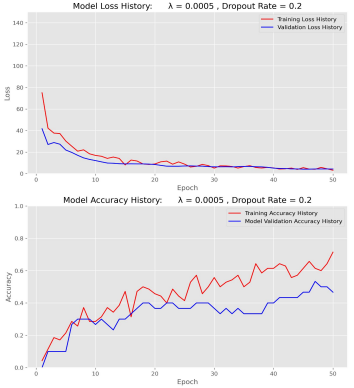
Combinations of

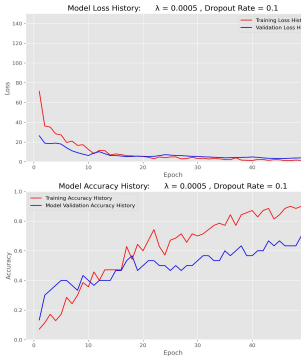
λ = [0.0005, **0.001**, 0.0025]

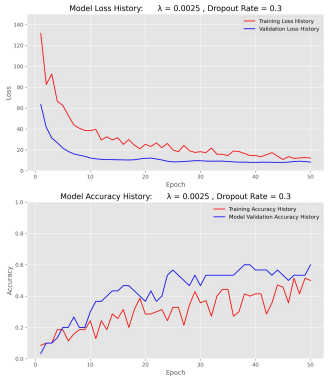
Dropout Rate = [0.1, **0.2**, 0.3]

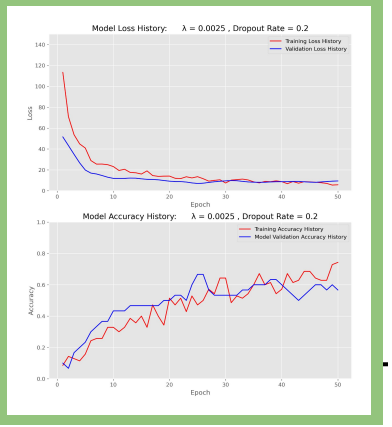


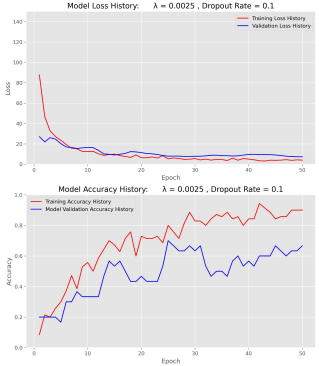


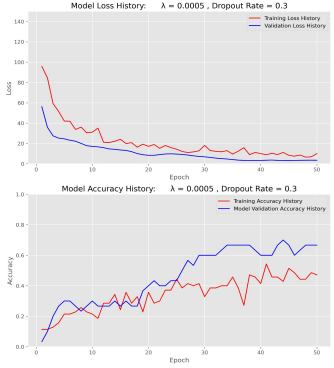






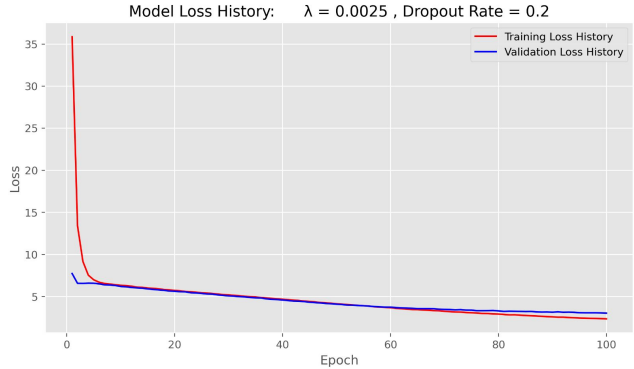






Results

- Maximum validation accuracy ~ 60 - 65%




```
# Predicting on Validation data
np.set_printoptions(suppress=True)

reverse_genre_lookup = {1: 'blues', 2: 'classical', 3: 'country',
                        4: 'disco', 5: 'hiphop', 6: 'jazz',
                        7: 'metal', 8: 'pop', 9: 'reggae',
                        10: 'rock'}

pred_sample = model.predict(X_test)[88] #Arbitrary Sample
pred_class = reverse_genre_lookup[np.argmax(pred_sample) + 1]

print(pred_sample)
print(pred_class)
```

```
[0.54177713 0.00209504 0.17671739 0.00625514 0.01669832 0.12706721
 0.04205757 0.00064518 0.01643534 0.07025157]
blues
```

```
In [31]:
end = datetime.now()
print ("Completed, Runtime: ", end-start)
```

```
Completed, Runtime:  0:06:39.682746
```

Additional Steps to Take

- Use Convolutional Neural Network (CNN) instead of MLP
- Explore withholding certain genres from the classification and training process to limit genre interference/overlap
- More training data
- More network layers (more time/computational resources)