# Amazon SageMaker Project
# Distress Call Classification Models

## Introduction

In today's accelerating demand for poultry, a main concern that arises with farmers is the impact of mass commercialization on the welfare of animals. One of AudioT's initiatives is to utilize recordings from broiler rooms across different farms to explore machine learning capabilities that can enhance our understanding of the animal welfare and help farmers act early when animals' welfare is impacted.

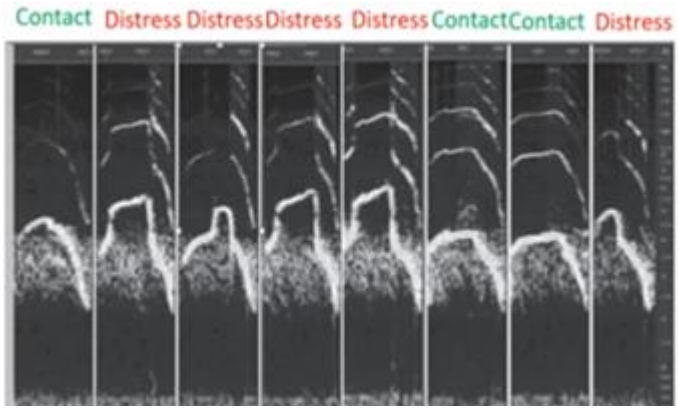

*Figure 1: Showing Sound Waves of Contact and Distress Calls*

In the current AudioT practicum, we aim to use audio data and machine learning to detect disease and distress among animals, specifically chickens, by analyzing the audio of flocks focusing on sounds such as chirps and coughs. The objective is to build a model that will analyze microphone recordings from both commercial farms and one contractor farm, to warn farmers of any distress calls that are detected in the animal sounds. Distress calls, especially for younger chickens, are crucial to detect, as they are indicator of any health problems and can also dictate if this young flock will survive or thrive over the next few weeks. Therefore, building distress call classifiers can help in early prevention of disease spread on the farm and predict the health of younger flocks at initial stages of life.

The purpose of the project is to classify a distress call of chickens that are within the first week of flock development. A distress call when viewed on a spectrogram is known to be a sequence of sharp consecutive chirps as shown below. However, the challenge is that a distress call generally is very similar to a contact (non−distress) call. For older chickens, this would be a challenge, as distress and contact calls are often similar, but for early-stage chickens, domain experts suggest that any type of call is considered to be a distress call.

In our group, each member was assigned different flocks from different months to work on. A flock refers to the month of recording that was received from the commercial farms. Therefore, we had three flocks：Allison was assigned on flock 7, Ramin was assigned with flock 10 and Sara was assigned with flock 12. These flocks correspond to the month in which data started to be gathered, I.e. 7 is September, 10 is October, and 12 is December. In addition, all three team members were analyzing recordings from the same chosen microphone within a commercial farm broiler room. This allowed for us to analyze the same location at three different points of time.

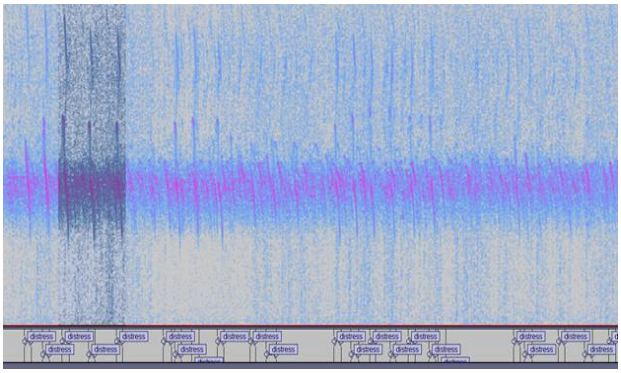

*Figure 2: Audacity Spectrogram of Audio Recording Showing Labels of Cough Sound Spectrogram*

| Start Time | End Time | Label |
|---|---|---|
| 19.335462 | 19.445846 | cough |
| 28.785493 | 28.944936 | cough |
| 29.410998 | 29.644030 | cough |
| 31.851694 | 32.139917 | cough |
| 38.597336 | 38.707719 | cough |
| 40.473851 | 40.602631 | cough |
| 42.270645 | 42.393293 | cough |
| 47.857263 | 48.120956 | cough |
| 54.198166 | 54.376006 | cough |

*Figure 3: The label file showing columns of labels of cough start and end time from the spectrogram*

Our data consisted of raw audio recordings from a commercial farm over several days with the intent to classify distress calls when heard in the audio files supplied. In order for us to produce tabular data that we can input into our supervised machine learning models, we needed to manually label distress sounds in the data.
We used the Audacity software to view the spectrograms and the frequency of the audio.

In the software, we were able to point at which times a distress call was detected (see figure 1). The Audacity software generates a text file that shows at which time in the recording there was a label (see figure 2).

The audio data volume that we were given to label was extensive. For each flock, we were focusing on the first 7 days worth of data. This was divided by the day, hour and minute. Each audio data was one minute long, therefore, we each had approximately 10,080 files of data.

And upon analyzing some audio files visually and by listening to them, there was a lot of variation in bird noise. This was due to birds moving around as sometimes they are closer to the microphone and sometimes they are a little bit further from the microphone but the actual distress level is the same. Thus, one would expect that when sampling data from the first 7 days of a flock from the same microphone, one would see different shades on the spectral graph for equal noise at different times. Therefore, when we are sampling our data, we must sample data from different ranges of loudness/prominence on the spectral graph to really capture the data correctly for our model.

Another challenge was determining the best sample of audio files for labeling to use for both training and validating the models from the 10,080 files. We were provided GAM charts that showed unnatural or inconsistent shifts between subsequent audio files. For instance, a burst of a new color may indicate the lights first turning off in the evening, which led to an increase in sound within the broiler room. To start, each team member chose to label 40 data points. We also were given anomaly scores for each audio file. We interpreted this as a measure of distance between the chicks and the microphone; higher anomaly scores had more distinct individual sounds. We tried to choose a balance of audio collected in both day and night, since our models would be trained on a combination of the two. Similarly, we chose a combination of audio files with anomaly scores throughout the total range.

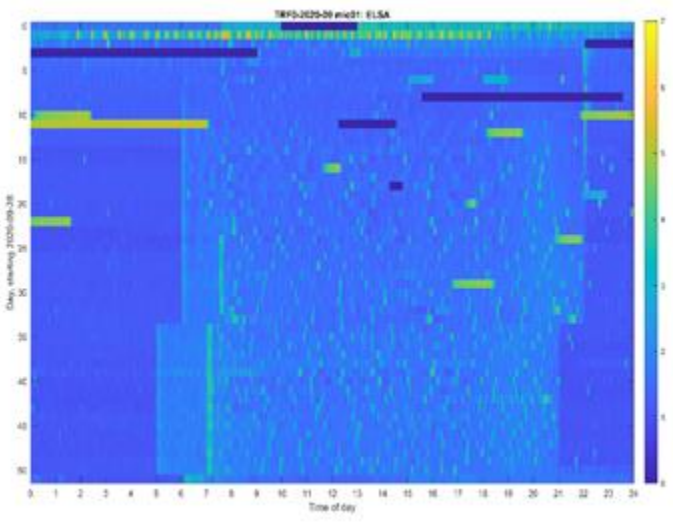

*Figure 4: GAM Chart of Anamoly Scores by Flock Day and Time*

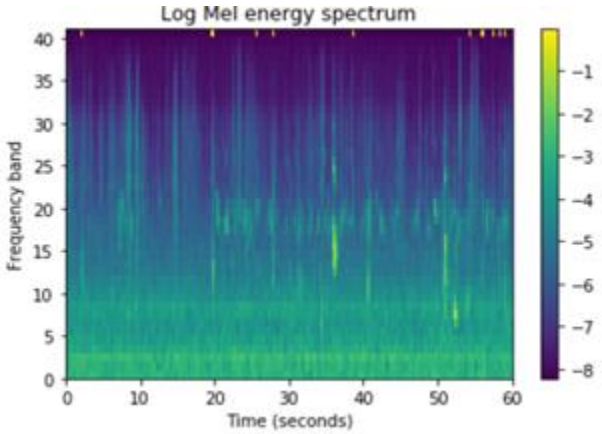


*Figure 5: Log Mel Energy Spectrogram*

This process also introduced room for human error and bias. In the first few weeks, we learned how to effectively label cough sounds and were able to apply this knowledge to other distress sounds. However, it is challenging to identify individual sounds from a large cluster and this task is subjective. Our goal was to attempt to label in a similar way for all three flocks. This is useful to compare model results on each of the different flocks.
To create the features for the model, we needed to process the audio data first. The basic component of the audio processing was done by using soundfile Python package, in which it processes the waveform and several audio properties. The waveform data was then used to compute features in the frequency domain by using log Mel energy which is a representation of the short-term power spectrogram of a sound, based on a linear cosine transform of a log power spectrum on a nonlinear mel scale of frequency.

When creating the features for audio recordings we used the company provided code to produce featres, and we specified a few parameters to create those features. The first parameter is the n_mel parameter, which according to the documentation of the log_mel_energy class, n_mel specifies the number of filters in the Mel-scaled filter bank. This

number of triangular filters will be spaced according to the Mel scale between the frequencies specified by minimum frequency and maximum frequency parameters. This number also determines the number of dimensions each output feature vector will have. Higher values will give features with higher resolution along the frequency axis, while lower numbers will give lower resolution. In speech recognition, this has often (historically) been set to 13.

# Methodology

Once the data was placed on S3 for model development we moved to model development on Amazon SageMaker. SageMaker is a Machine Learning platform that functionalities that allows data wrangling, model building and deployment.

Among group members, each member built separate model on AWS SageMaker using SageMaker's built in algorithms such as KNN, linear learner and Xgboost. We chose various supervised learning algorithms so we could compare the predicted results to the actual results. The goal is to test the accuracy and recall in each model order for us to then compare our results and evaluate which model had the best metrics.

## k-nearest neighbors (KNN)

Below is the results of the metrics of the model trained on training data from each flock:

| | Model Built using All Flocks data |
|---|---|
| Accuracy | 0.81 |
| Recall | 0.18 |
| f1_Score | 0.18 |

The parameters that were chosen for this model using sklearn Grid Search CV optimized on recall

| Metric | n_neighbors | weights |
|---|---|---|
| euclidean | 2 | distance |

Features' parameters were tuned with n_mel=5, and minimum and maximum frequency set to 2000 and 4500 respectively. The figure below shows the log mel energy as set to those parameters. The brighter yellow cell is where the distress call is most distinguished hence why those parameters were chosen.

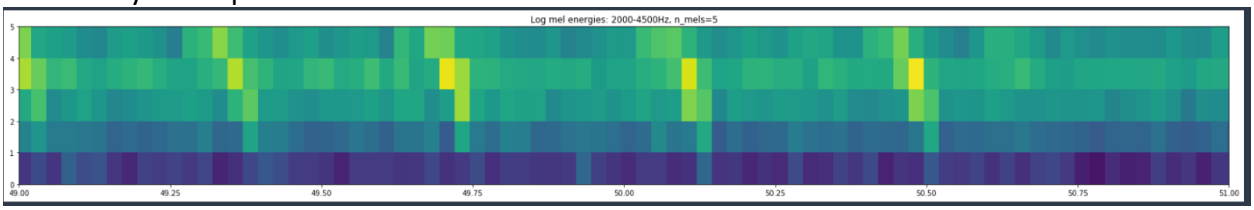


*Figure 6: Log Mel Energy Spectrogram for Distress Calls*

As we can conclude, KNN is not a good model to use for distress detection as the recall score is very low, which means very low positive cases were detected. This indicates that there is either overfitting or not enough data to label.

## XGBoost

One advantage of the SageMaker platform is the built-in models. We chose to use an XGBoost model, or an extreme gradient boosting model. This type of models attempts to predict the target variables by combining estimates of a set of simpler models. With over 40 hyperparameters that can be modified, XGBoost traditionally performs well due to the ability to tune these parameters. I started with modifying the following parameters in different models: max_depth, min_child_weight, gamma, and eta. Adjusting these parameters can be helpful in ensuring the model reduces overfitting and underfitting. One weakness in utilizing XGBoost is the smaller size of the given data set, due to limited time in manually labeling the existing audio files.

## Logistic Regression

As we are tying classify models with distress calls, the problem represents a binary classification model. And one of the ideas that comes to mind is the Logistic Regression model. This model that was built and deployed on Amazon SageMaker. In fact, there were 2 models that were built and deployed 2 as Logistic Regression models. These are live endpoints that are deployed in the cloud and can make predictions given the data right now. One of them was built using SageMaker's Autopilot feature.

The following confusion matrix was produced from the first model built using sklearn:

| Predicted | 0 | 1 |
|---|---|---|
| Actual | | |
| 0 | 3035 | 2206 |
| 1 | 11303 | 2658 |

Other metrics were also noted:

- Precision = 3035 / (3035 + 11303) = 0.2117
- Recall = 3035 / (3035 + 2206) = 0.5791
- Accuracy = (3035 + 2658) / (3035 + 11303 + 2206 + 2658) = 0.2965
- F1 score = 2 * 0.2117 * 0.5791 / (0.2117 + 0.5791) = 0.3101

Interestingly, the second model that was produced by the SageMaker had a higher F1 metric, 0.41.

The results above show that both models are the best in classifying distress in terms of recall and F1. This is likely due to not having enough data, or labels, and we also need perhaps better labeling. As one of the challenges pointed out in labeling that is being prone to error, better labeling could potentially lead to better classification. More detailed labeling with labeling of every sound, like `distress`, `water`, `fan`, `noise`, etc will produce the kind of data with which multiclass classification on Amazon SageMaker would be possible. The target variable for multiclass classification would have more classes and this is easily processed by SageMaker. There is a template from which to build multiclass classification model. If one could select labels with a freehand contour in Audacity, it would help with building a better model.

## Conclusion and Discussion

Upon looking at the anomaly scores for audio data that was provided for each day that represent the age of the chicken, we wanted to see the variation in noise chicken make by each of the first six days of life. Using flock 12 day data, we can see the variation in average anomaly scores below, it appears that the variation from the plot below is very significant in the first

days of life, showing steep decline in the last three days. This may suggest that to create a good classifier, will want to create a model for each day of life for the flock to perhaps get less variation in the data.

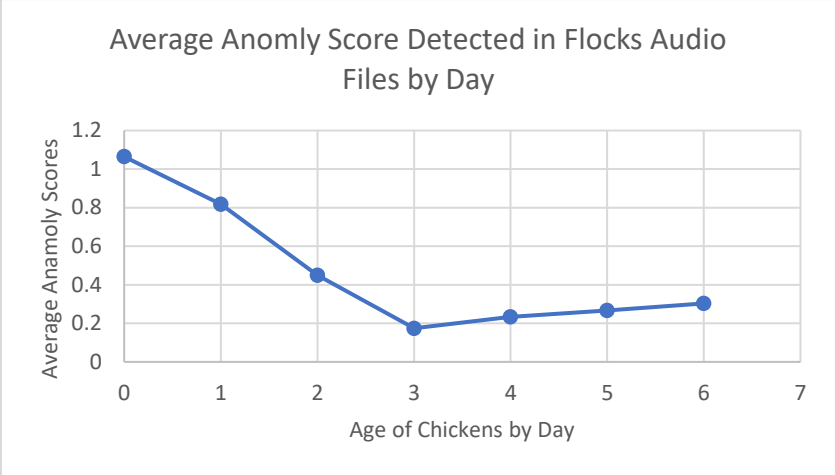


*Figure 6: Showing Anomaly Score Average by Chicken Age*

Another recommendation we have from this project is to have a visualization that shows how a particular flock is doing today with respect to other flocks at time period ***t***. Provided that everything else like temperature, humidity, feeding and time schedules are the same for every flock, and if we were to agree that ***t*** would represent a minute in the life of a flock, would could index every minute of every flock from beginning to end and we could join flocks on index. This would allow us to create a multiline time series chart that shows if the flock on hand is within range of the acceptable anomaly scores. The chart below shows just 3 flocks stacked on top of each other by the minute index. It does have a color code.

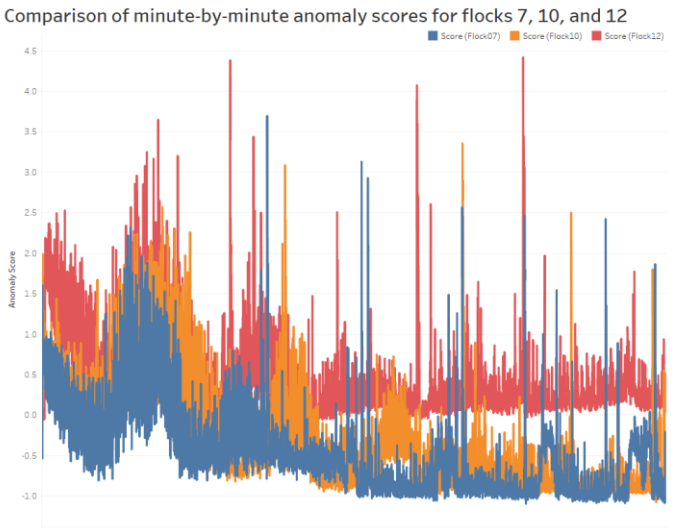


*Figure 7: Anomaly score Time Series for flock 7, 10 and 12*

However, as data about more flocks is acquired, more lines will be added to the chart and color will make it more difficult to understand. At that point, all will matter is if a given flock is within a range most other flocks were, anomaly wise. In that case, the chart would look something like figure 8 below.

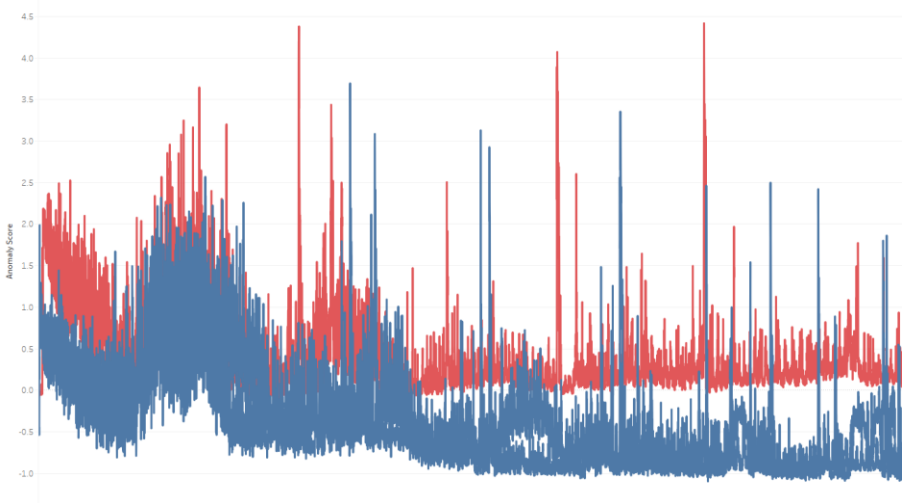


*Figure 8: Demonstration of anomaly scores by flocks*

This model focuses on whether a given flock is within a "band" with other flocks or is more loud than others. The next question would be then: does this anomaly affect the KPI, the bottom line? Should a model be built to describe a relation between being louder than others and the yield?

In conclusion, although the logistic regression model has the best scores among the three models presented, there is a lot of room for improvement. Creating more labels hence more data and creating daily models could perhaps improve the recall and F1 scores thus better detection of distress calls.

Overall, this was a very interesting project to learn about audio processing taking audio files to the labels and creating features, which then get used to build powerful distributed models, hosted in the cloud to be used by commercial farms.