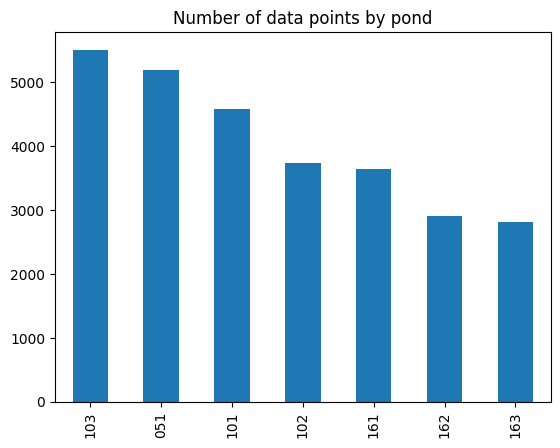
**EATING POSTURE RECOGNITION OF FISH USING ACCELEROMETER SENSOR DATA**

**Introduction**

To maintain fish nutritional value, the way in which fish are handled, processed and stored is critical. The eating posture of fish is one of the critical aspects in its preprocessing. In this project, I studied a model that can recognize the eating posture of fish using accelerometer sensor data.

**Data**

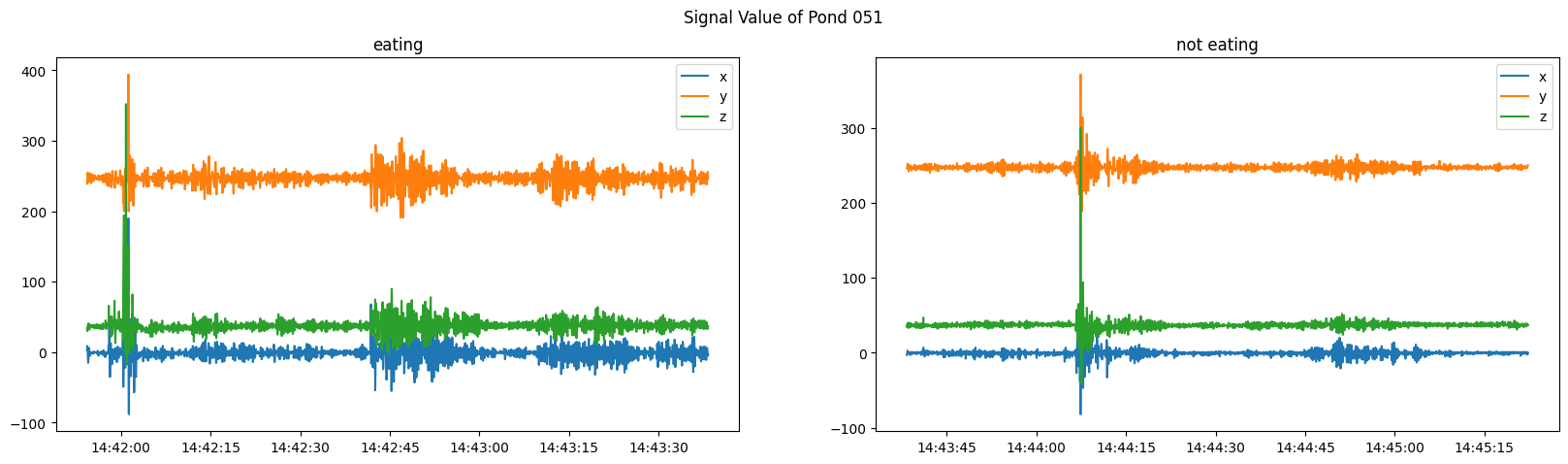
The data used in this project consists of accelerometer sensor readings taken from a group of fish from different ponds while they were in different eating postures (eating or not eating). The dara includes the timestamp, and three axes sensor data of x, y, and z. The data used take the period of 2022-07-06 14:40:02 GMT+7 to 2022-07-06 14:58:57.640000 GMT +7.

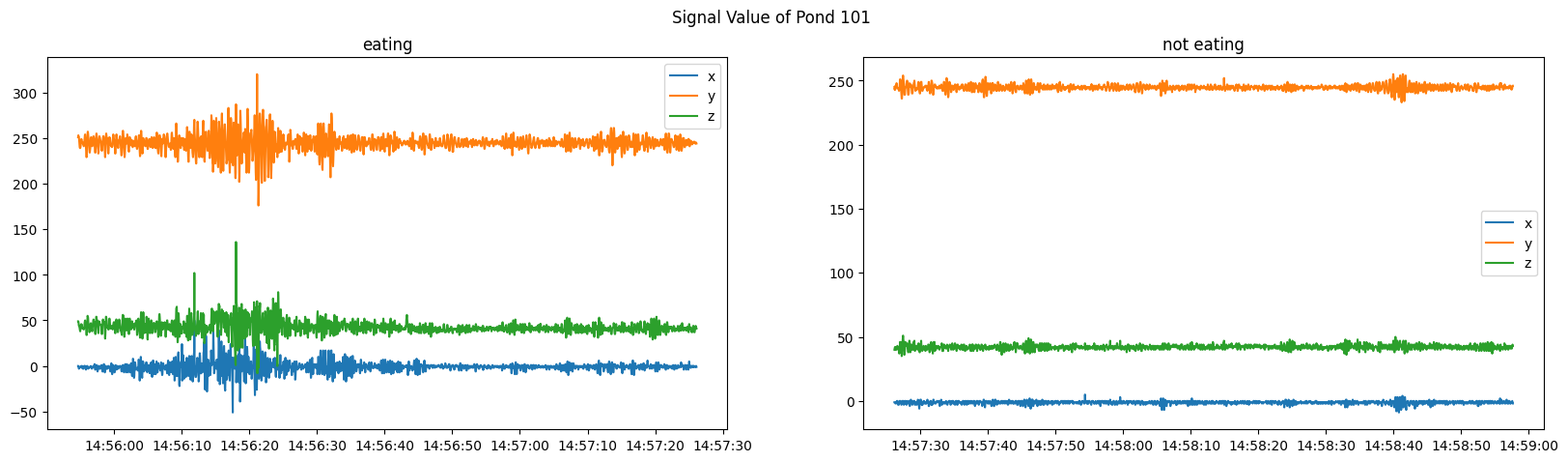


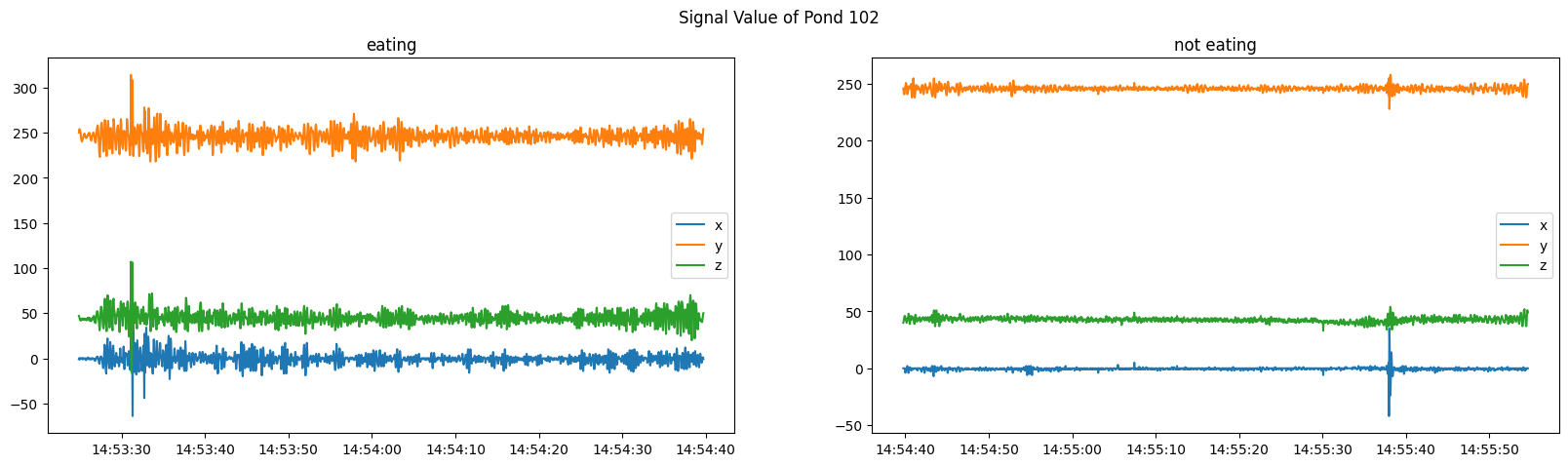
Data consists of 28,392 data points which at most come from a fish group (pond) of 103.

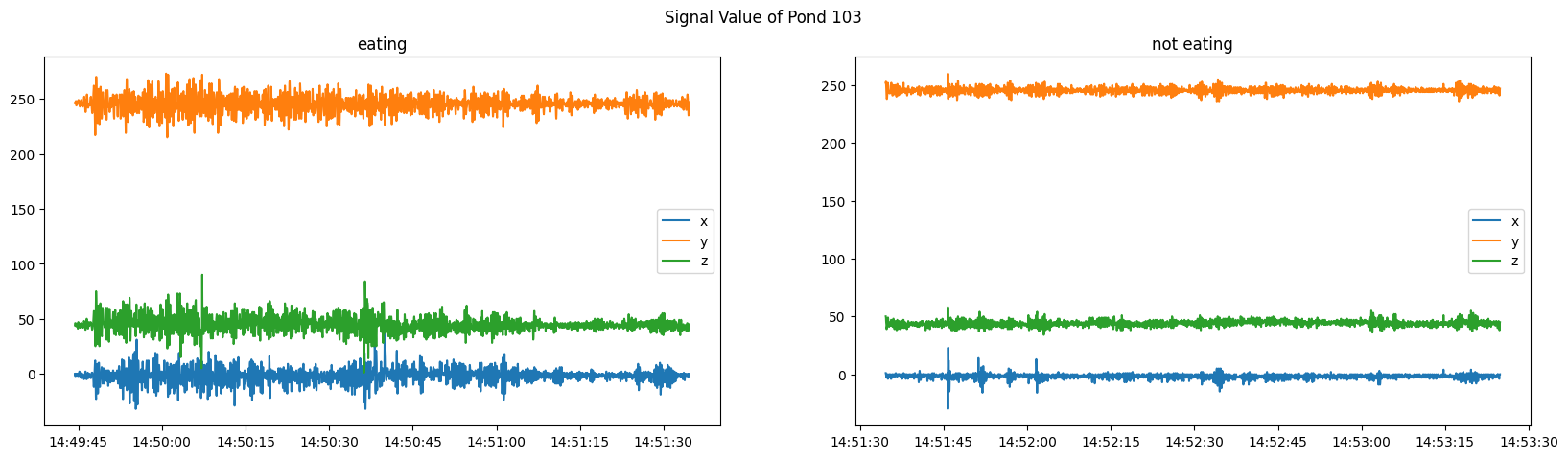
**Data Exploration**

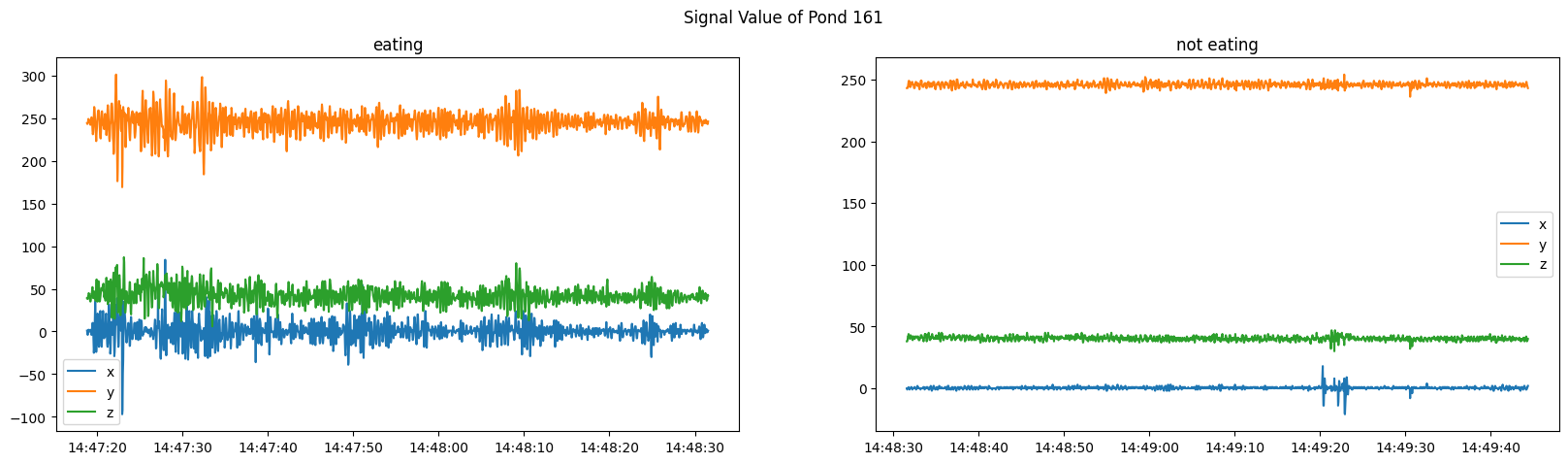
Before building any ML approach, I try to digest the x,y,z feature first for the sake of gaining a sense of what the data look like.

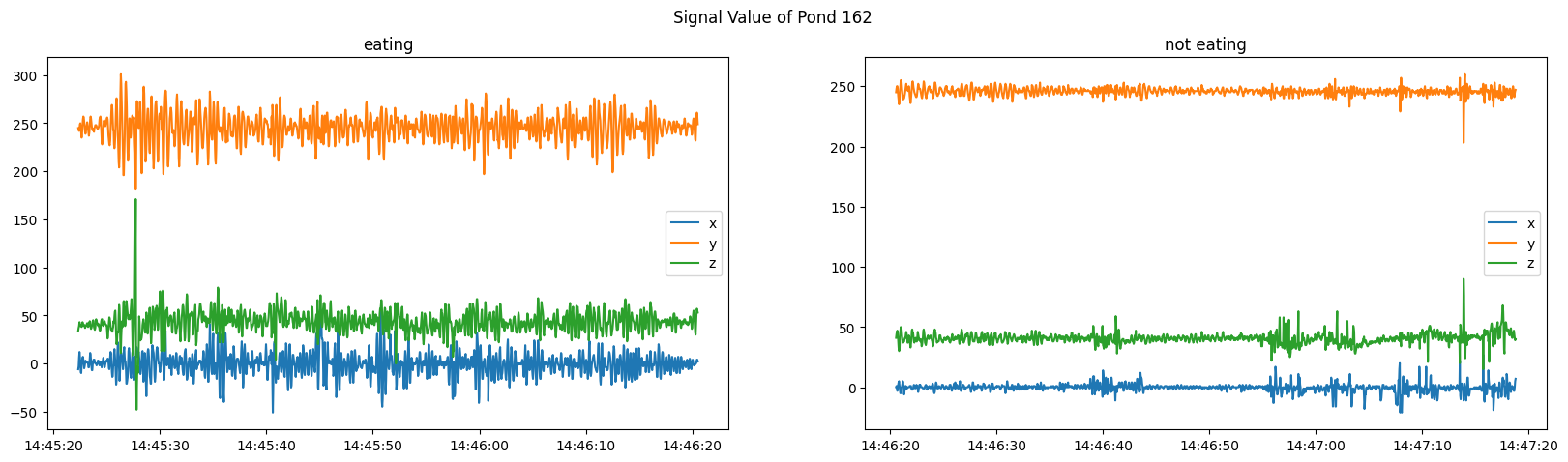


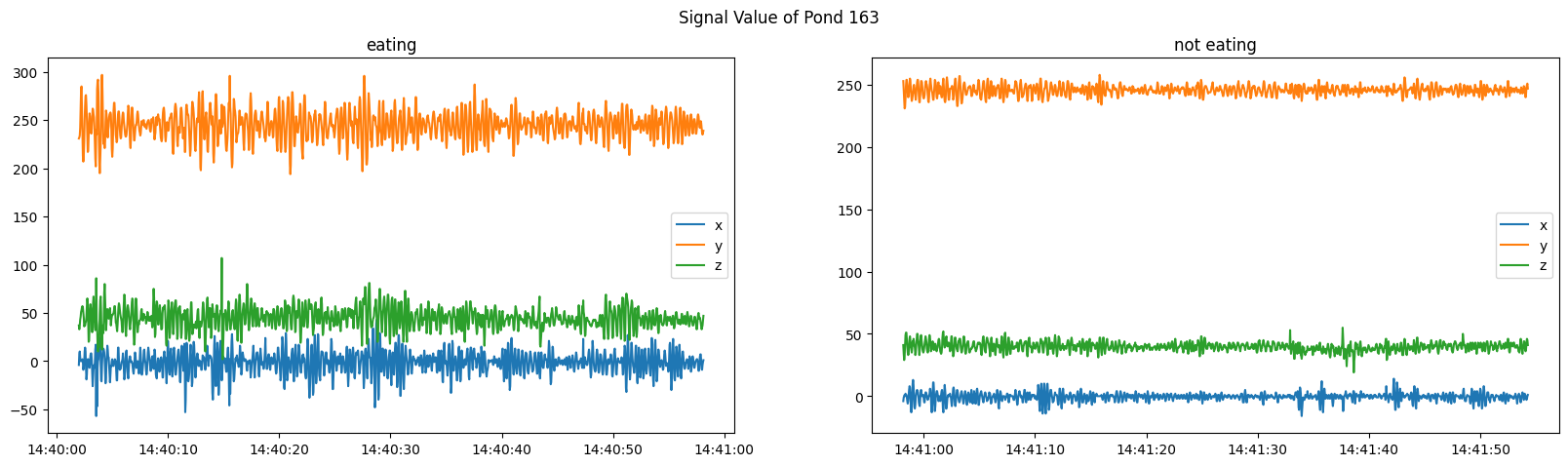






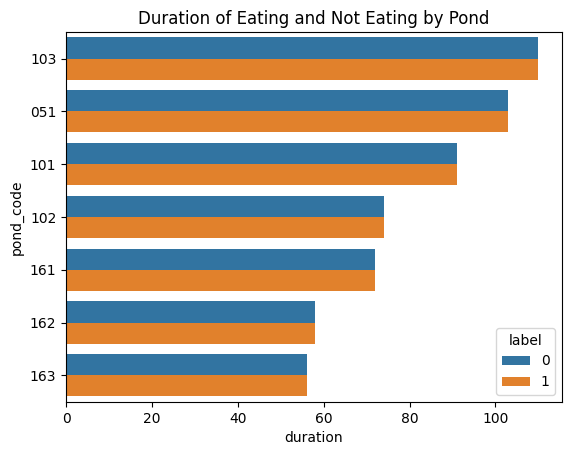






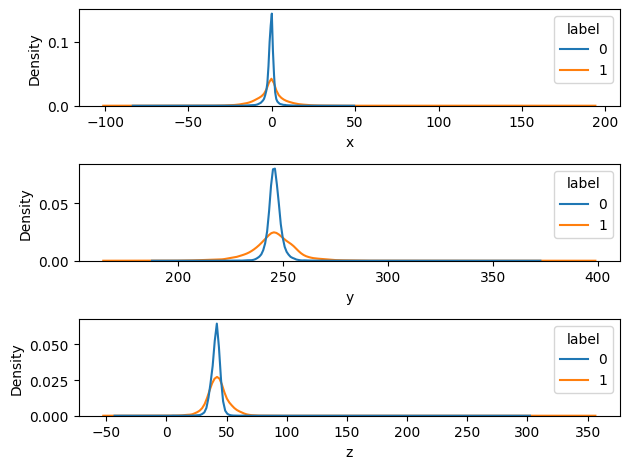
Frankly speaking, the sensor data for the eating position may fluctuate more than when the fish is not eating. The cause might be due to numerous factors. One of which we can think of is that the food is moving around in the pond/ tank, causing the water to move and sensor data to fluctuate. In addition, this phenomenon can be more pronounced if the fish is consuming small particles or simply the food that fish consumed is not easily digestible. For that, considering and studying other aspects such as food characteristics might give us a better understanding of it.

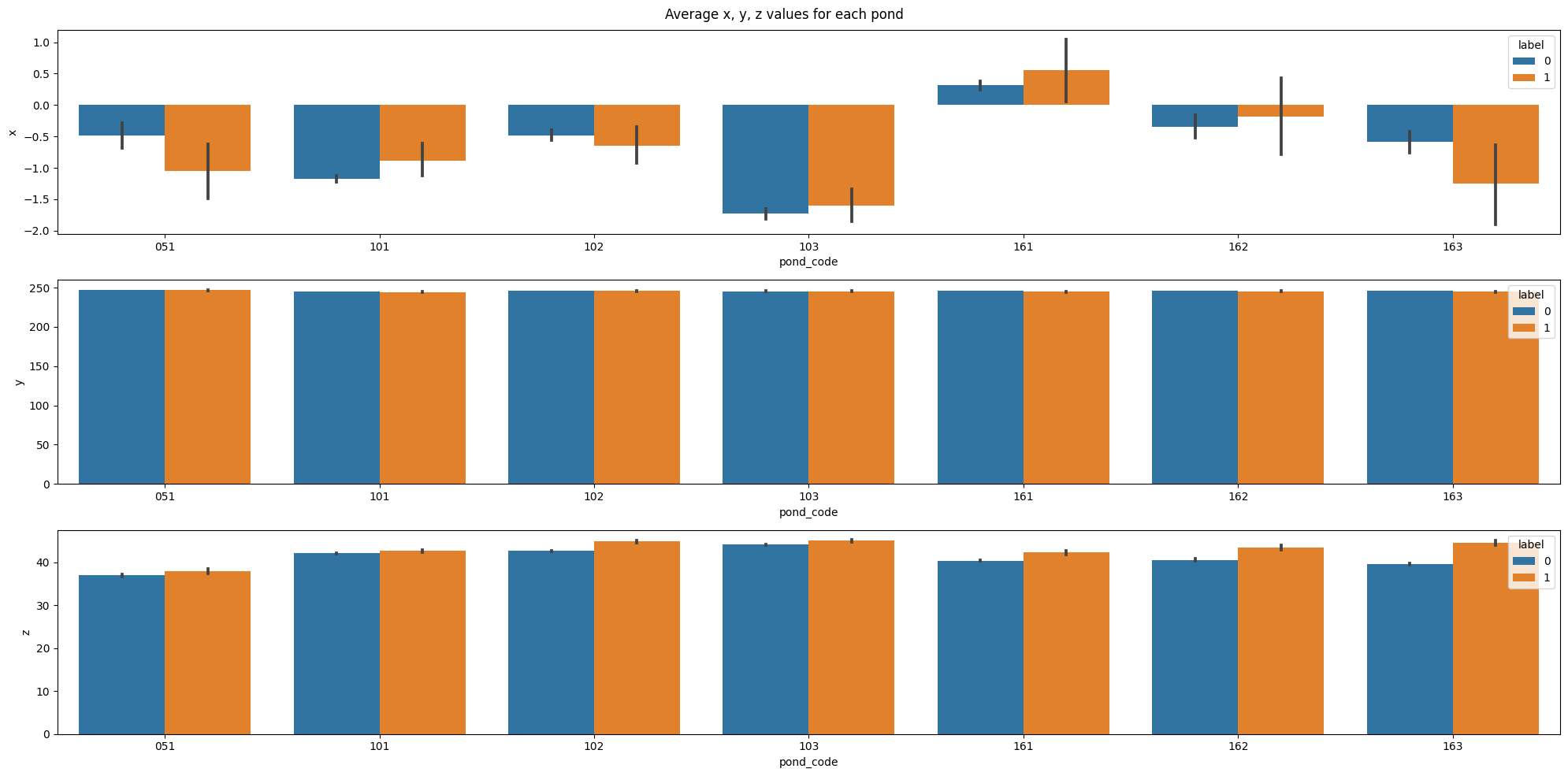
Moreover, the fluctuation can be caused by the movement of the fish itself. When they are eating, they may be moving around in the pond to access the food or simply avoid other fish that also try to eat.



The problem becomes interesting when I see that after a certain amount of time, the data show that the fish are not eating anymore. I try to calculate how long each pond spends on eating. Again, pond number 103 spent the longest time eating. There might be numerous factors behind it. First, the number of fish in pond 103 is greater than others. Next, a larger pond/tank size may provide more space. We also need to study the type of food fed. If the fish are being fed pellets that sink to the bottom of the pond, it may take longer to consume. Also, different fish species might have different feeding behavior, and so on.

Next important aspect to study is the feeding schedule. If the feeding schedule is inconsistent, some fish may be hungrier and consume more food in a shorter amount of time. By the graph above, we can say that some ponds have an earlier feeding schedule. Since the problem only mentions that the feeding process is an auto feeder, I’ll assume that this auto feeder works in a parallel manner. To decide which feeding schedule to follow, we need to study other factors such as the fish type, tank size, etc. Another factor before deciding which feeding schedule to follow is, what kind of domain/business improvement to be expected by arranging this schedule. Since instruction doesn’t provide it, we’ll go with the current schedule.





**Model Building**

**Experiment 1**

I try to treat it as time series classification, which means that we want to predict **n** times ahead, what will be the fish position. But again, the aims of the instruction are not clearly stated, I can't be sure whether this recognition is urgently needed or not. When treating it as a time series problem, my model is not performing well because the data start from eating position to not eating. Thus, in the test set we will have only non eating position data. For that reason, I ignore this approach.

**Experiment 2**

Second trial, I tried to transform the signal data of x,y,z position with Fourier Transform.

Model: rf

Confusion Matrix:

[[3061 1198]

[1490 2769]]

Classification Report:

precision recall f1-score support

0 0.67 0.72 0.69 4259

1 0.70 0.65 0.67 4259

accuracy 0.68 8518

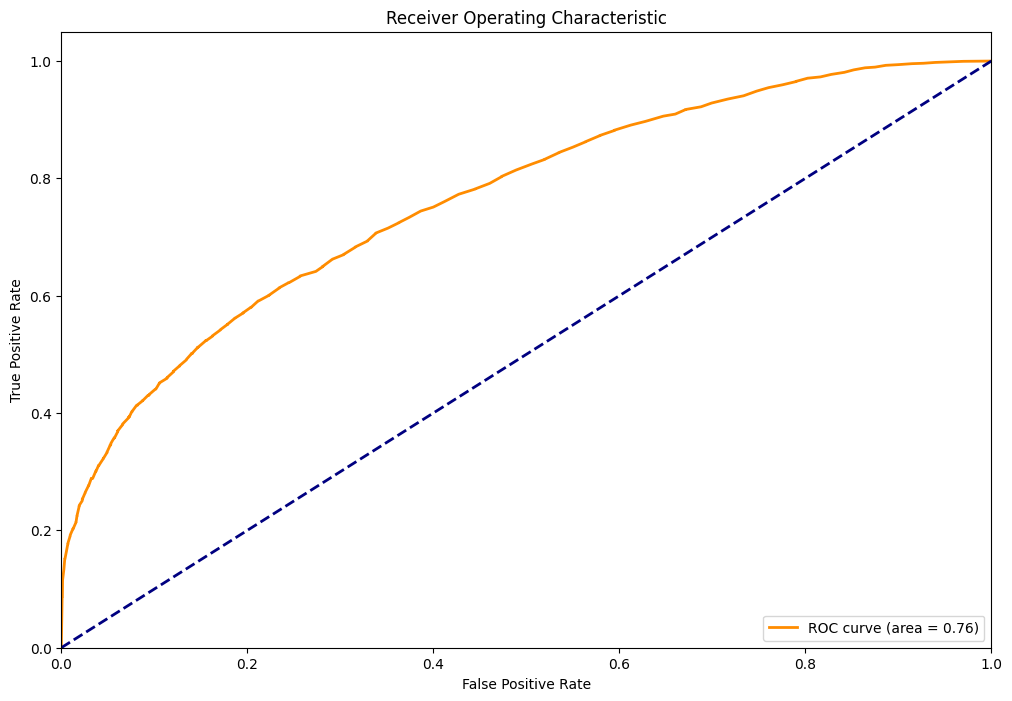
macro avg 0.69 0.68 0.68 8518

weighted avg 0.69 0.68 0.68 8518

ROC AUC Score: 0.6844329654848557

weighted avg 0.69 0.68 0.68 8518

=====================================



Model: xgb

Confusion Matrix:

[[3336 923]

[1490 2769]]

Classification Report:

precision recall f1-score support

0 0.69 0.78 0.73 4259

1 0.75 0.65 0.70 4259

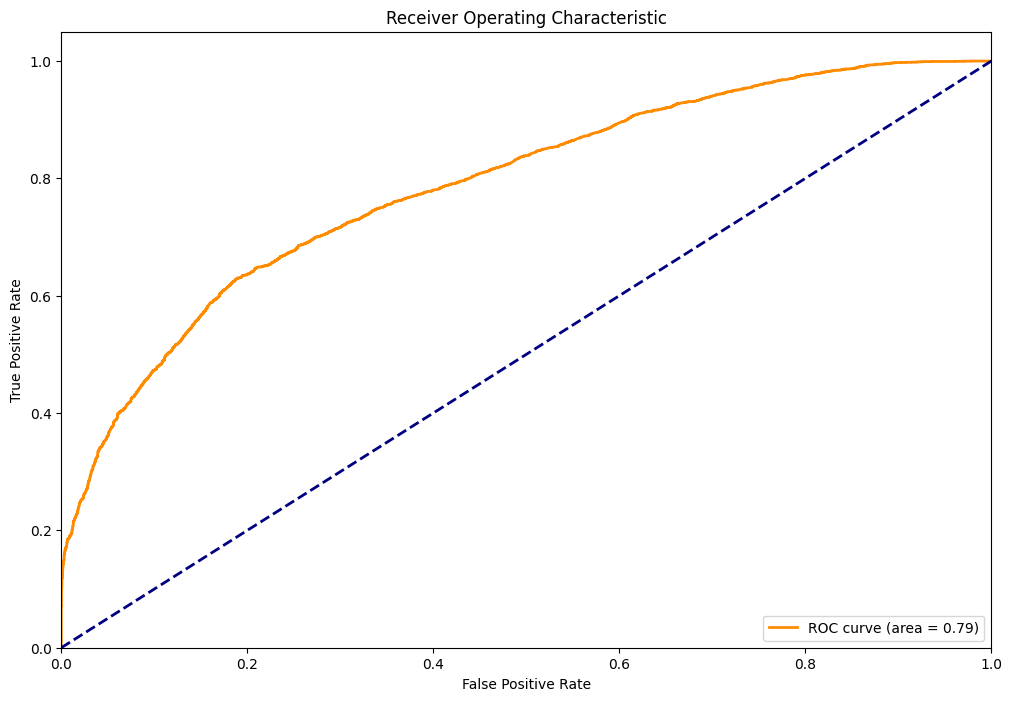
accuracy 0.72 8518

macro avg 0.72 0.72 0.72 8518

weighted avg 0.72 0.72 0.72 8518

ROC AUC Score: 0.716717539328481

=====================================



Model: stacked

Confusion Matrix:

[[3304 955]

[1467 2792]]

Classification Report:

precision recall f1-score support

0 0.69 0.78 0.73 4259

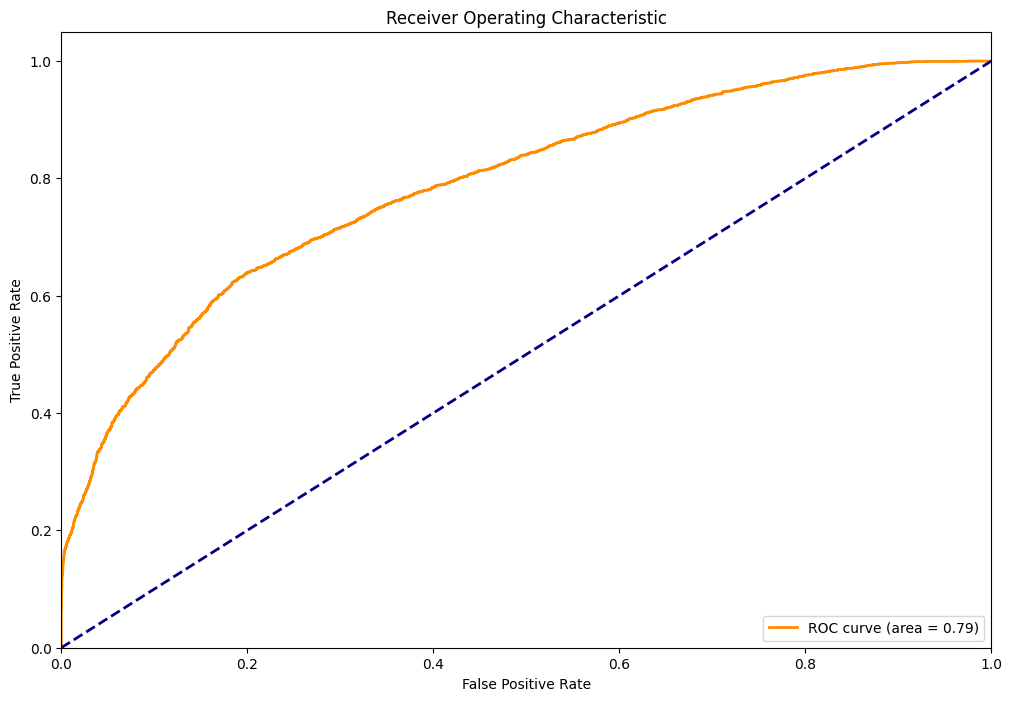
1 0.75 0.66 0.70 4259

accuracy 0.72 8518

macro avg 0.72 0.72 0.71 8518

weighted avg 0.72 0.72 0.71 8518

ROC AUC Score: 0.7156609532754167

****

The performance is greater than 70% but still doesn’t achieve the rule of thumb of 80%. But again the good metrics might be different on aquaculture problems.

**Experiment 3**

For the third experiment. I use window rolling and time\_lagging up to time t. At first I try to conduct with lag = 3,5,7,10. At the lag = 10, my model achieved 88% accuracy (please refer to experiment.ipynb). But I found the phenomenon that the increase of lag will impact the increase of accuracy. Thus, I use a time lag of 20 and a rolling window of 5. I concatenate data from various ponds/tanks into one by adding dummy features for the pond code.

Model: rf

Confusion Matrix:

[[3830 431]

[ 376 3862]]

Classification Report:

precision recall f1-score support

0 0.91 0.90 0.90 4261

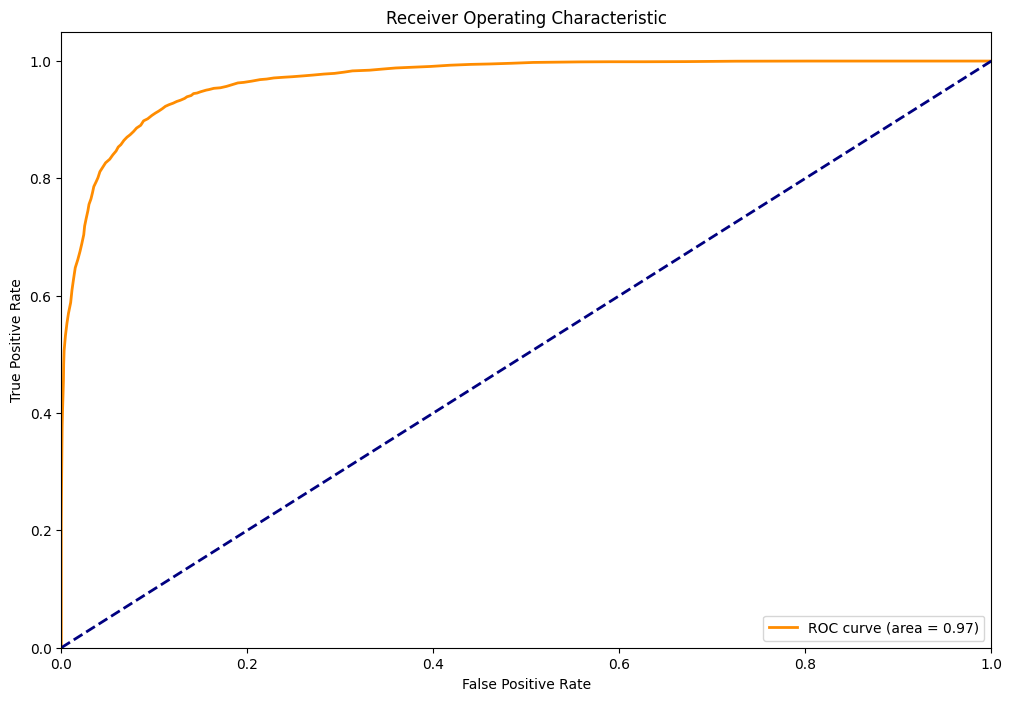
1 0.90 0.91 0.91 4238

accuracy 0.91 8499

macro avg 0.91 0.91 0.91 8499

weighted avg 0.91 0.91 0.91 8499

ROC AUC Score: 0.9050644701734699



=====================================

Model: xgb

Confusion Matrix:

[[3915 346]

[ 412 3826]]

Classification Report:

precision recall f1-score support

0 0.90 0.92 0.91 4261

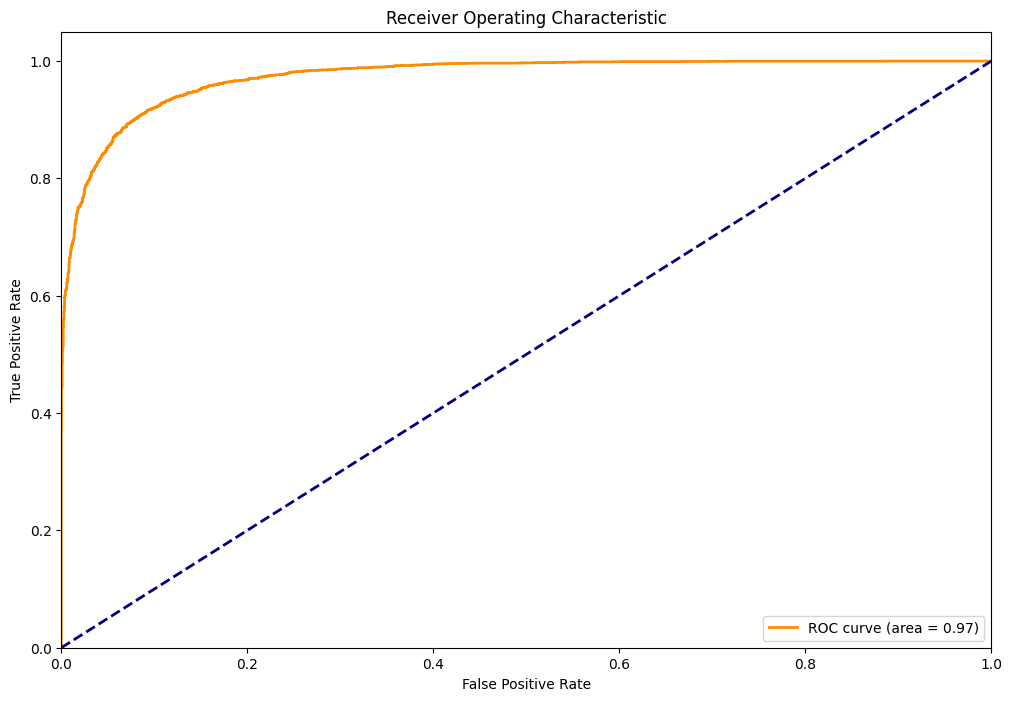
1 0.92 0.90 0.91 4238

accuracy 0.91 8499

macro avg 0.91 0.91 0.91 8499

weighted avg 0.91 0.91 0.91 8499

ROC AUC Score: 0.9107913681813353



=====================================

Model: stacked

Confusion Matrix:

[[3991 270]

[ 524 3714]]

Classification Report:

precision recall f1-score support

0 0.88 0.94 0.91 4261

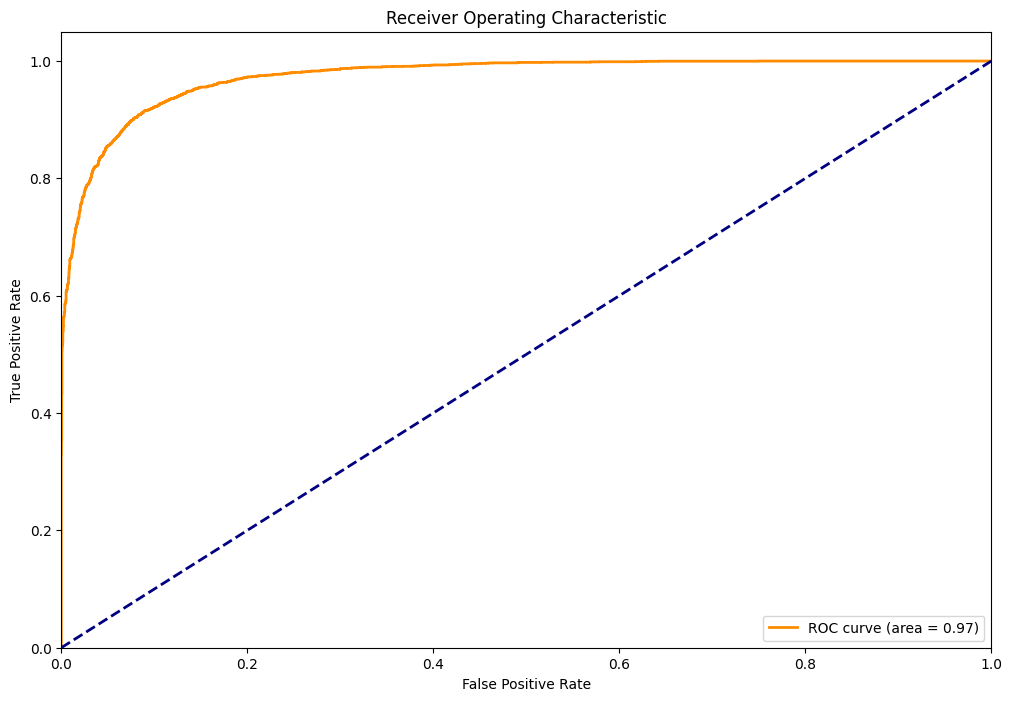
1 0.93 0.88 0.90 4238

accuracy 0.91 8499

macro avg 0.91 0.91 0.91 8499

weighted avg 0.91 0.91 0.91 8499

ROC AUC Score: 0.9064956824404403



For the third experiment, my XGBoost model outperforms the random forest and stacked model. Again, the information of project aims need to be defined so at least I can sense what this model will contribute for.

**CONCLUSION**

To sum up, in this project we are able to develop 91% accuracy of fish eating gestures by using accelerometer sensor data. Future work could involve other timestamps since the feeding schedule isn’t mentioned in the problem or other aspects to improve the performance.

**REFERENCES**

Human activity recognition using magnetic induction-based motion signals and deep recurrent neural networks <https://www.nature.com/articles/s41467-020-15086-2>

<https://www.mdpi.com/1424-8220/20/4/1167>

<https://www.sciencedirect.com/science/article/pii/S1877050914008643>

<https://www.analyticsvidhya.com/blog/2021/07/implementing-lstm-for-human-activity-recognition-using-smartphone-accelerometer-data/>

etc