CALIFORNIA STATE UNIVERSITY
1949
LONG BEACH

EE 576 Spring 2023

PROJECT 5

By

Tylor Cooks

Date: 5/13/2024

SHARK BEHAVIOR

CLASSIFICATION

Shark Behaviour Classification with an Unbalanced Dataset

Tylor Cooks, 026538081

Electrical Engineering Department, California State University Long Beach

Abstract— **This project used high-resolution time series data to classify shark behaviour into four distinct categories: resting, swimming, feeding, and non-directed motion (NDM). Classification utilizes neural networks, specifically neural networks (NNs), long short-term memory (LSTMs), as well as the K-nearest neighbour (K-NN) approach. The goal of this project is to compare the effectiveness of these techniques on the shark behavioural dataset which include seven shark datasets, each representing a different behavioural class within a specific time frame. Techniques such as SMOTE (Synthetic Minority Over-sampling Technique), ADASYN (Adaptive Synthetic Sampling Approach), and SMOTE-SVM (Synthetic Minority Over-sampling Technique with Support Vector Machine) were employed to address the challenges of biased datasets, particularly the under-representation of feeding behaviours.**

***Keywords*— KNN, supervised learning, unbalanced dataset, feature extraction, oversampling.**

I. Introduction

Classification is a supervised learning approach in machine learning where the goal is to predict the categorical classes of new observations, the test dataset, based on the past observation in which the outcomes were already known, the training dataset. One of the requirements in building an effecting machine learning model is having a balanced dataset. A balanced dataset in the context of classification is a dataset where each class has roughly the same number of data points. This helps a model not becoming biased towards a class during the training phase. However, real-world datasets frequently exhibit class imbalances, where some classes are significantly underrepresented compared to others. This imbalance poses substantial challenges to classification algorithms.[1]

Effective handling of imbalanced datasets involves preprocessing techniques, such as oversampling of the minority classes to help balance out the dataset. Feature extraction also plays a critical role for unbalanced datasets by transforming the data into usable features that results in improved predictive outcomes. By extracting relevant features that capture the important aspects of the data, we can significantly enhance the performance of classification models, particularly in scenarios dominated by class imbalances.[2]

Moreover, the choice of classification technique can have different results while dealing with an imbalanced data. Different algorithms, such as NN, KNN, and LSTM have varying sensitivities to the proportion of classes within the dataset. The results of using such different methods may result in varying ranges of performance results.

This project investigates these different methodologies, highlighting the synergy between feature extraction techniques and different classification algorithms to address the challenges caused by an imbalanced dataset. Through a combined approach, the goal is to improve the robustness and performance of classification models with an unbalanced dataset.

II. Methods

The procedure for this project is to first, feature extraction of the dataset that will improve model performance. Second, apply over-sampling on the minority classes to transform the unbalanced dataset into a more balanced dataset. Lastly is to compare the different models and their performances. The over-sampling algorithms for the minority classes used in this project are: SMOTE, ADASYN, and SMOTE-SVM. The models that are being compared against are: KNN, NN, and LSTM.

A. Splitting the Dataset

The dataset was originally a collection of 28 files that contained an equal distribution of the different classes: feeding, swimming, resting, and NDM. The dataset was split before combing the files for training by taking 1 of each of the classes for the training dataset. The choice was chosen by attempting to have create a training set that had the same distribution of classes. The reason for this methodology, rather than randomly selecting data points, was to ensure that the order of the data remained intact, due to the dataset being a time series, where order is important for training the model.

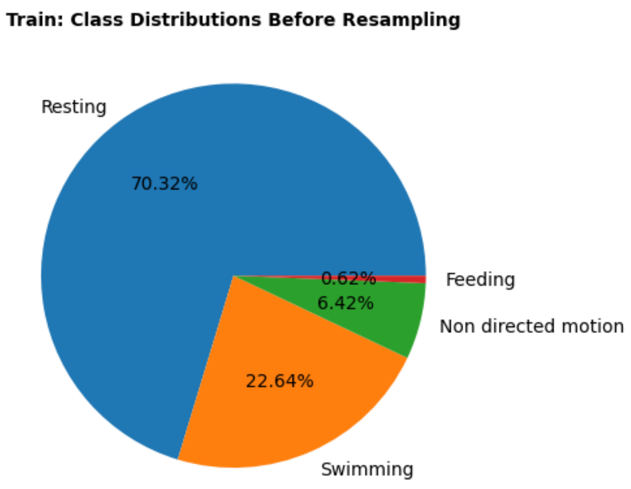


Fig. 1 Training Dataset Distribution

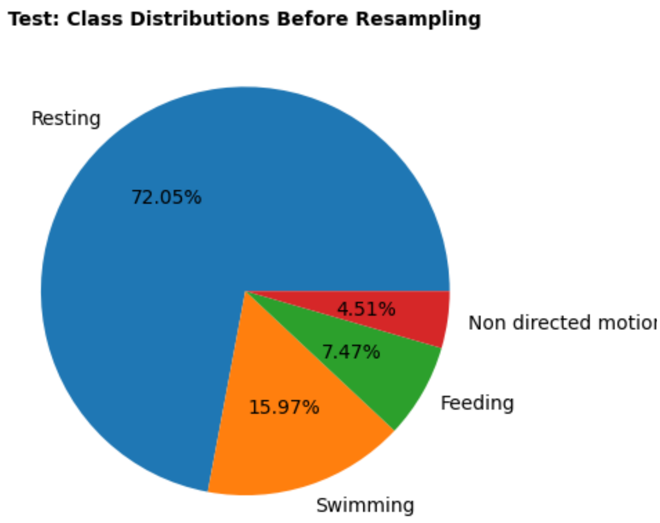


Fig. 2 Testing Dataset Distribution

B. Feature Extraction

The original dataset is multivariate time series including six motions describing the behavioural motion of a shark sampled at 25Hz. The six motions include: X-dynamic, Y-dynamic, Z-dynamic, X-static, Y-static, and Z-static. With v being the feature extracted and $S_{1\text{-}3}$ being the X Y Z dynamic features and $Y_{1\text{-}3}$ representing X Y Z static features. Each feature was performed on 25 samples and return one value, which down sampled the original dataset to per second. After feature extraction, feature performance was initiated to extract which features were most important for classification. The resulting features had the highest correlation with the label they were affiliated with. [1]

The FFT was performed and the highest value of the FFT coefficient was return.

$$v_{FFT,S} = [\max(FFT(S_1))\ \max(FFT(S_2))\ \max(FFT(S_3))\ ] \quad (1)$$

$$v_{FFT,Y} = [\max(FFT(Y_1))\ \max(FFT(Y_2))\ \max(FFT(Y_3))\ ] \quad (2)$$

The standard deviation:

$$v_{std,S} = [\ std(S_1)\ \ std(S_2)\ \ std(S_3)\ ] \quad (3)$$
$$v_{std,Y} = [\ std(Y_1)\ \ std(Y_2)\ \ std(Y_3)\ ] \quad (4)$$

The interquartile range:

$$v_{irq,S} = [\ irq(S_1)\ \ irq(S_2)\ \ irq(S_3)\ ] \quad (5)$$
$$v_{irq,Y} = [\ irq(Y_1)\ \ irq(Y_2)\ \ irq(Y_3)\ ] \quad (6)$$

TABLE I
TRAINING DATA SET AFTER FEATURE EXTRACTION

Classes	Samples
Resting (4)	47878
Swimming (3)	15415
NDM (2)	4370
Feeding (0)	424

TABLE 2
TESTING DATA SET AFTER FEATURE EXTRACTION

Classes	Samples
Resting (3)	1119
Swimming (2)	247
NDM (1)	70
Feeding (0)	116

C. SMOTE

SMOTE creates new minority instances by taking k-nearest neighbour between two feature vectors in the same minority class and multiplying a randomly generated number between 1 and 0. The goal is to balance the class distribution by randomly generating new samples from the original samples in the minority class.

D. ADASYN

ADASYN is a variant of the SMOTE algorithm, however, unlike SMOTE, ADASYN considers the density distribution r_i , which determines the number of synthetic instances generated for samples that are difficult to learn. This method focuses on adaptively shifting the decision boundaries to accommodate hard-to-classify samples. This adaptive approach to modifying decision boundaries based on learning challenges is the primary distinction from SMOTE. [4]

$$r_i = \frac{\#\ majority\ class\ samples}{K} \quad (1)$$

Where r_i indicates the dominance of the majority class in each K-neighbourhood. For the project K = 5, which represents the number of nearest neighbours of the majority int the neighbourhood for the voting process [4]

E. SMOTE-SVM

A variant of smote that uses the SVM algorithm which detects samples to be used for generating new synthetic samples. This approach starts by using a Support Vector Machine (SVM) to transform the dataset into a new format where it's easier to see the boundaries between classes. These boundaries are defined by key data points known as support vectors, then calculate the distances to their nearest neighbours within this transformed space. This method is specifically designed to improve the accuracy of classifying datasets where some classes have far fewer samples than others. [3]

III. Results

The results show the accuracy, F1 scores, as well as the confusion matrix of the different types of schemes used for the shark behaviour classification. When comparing the NN and LSTM, the dataset that produced the best results from the KNN was used to train the model for the NN and LSTM. The ideal oversampling for the minority classes, feeding and NDM were 10,000 samples, while the classes Resting and Swimming remained the same.

Precision measures the positive predictions to the total number of positive predictions made.

$$Precision = \frac{True\ Positive}{True\ Postive + False\ Positive} \quad (7)$$

Recall measures the ability of a model to find all the relevant cases within a dataset, or the ratio of true positives to the total number of positive predictions.

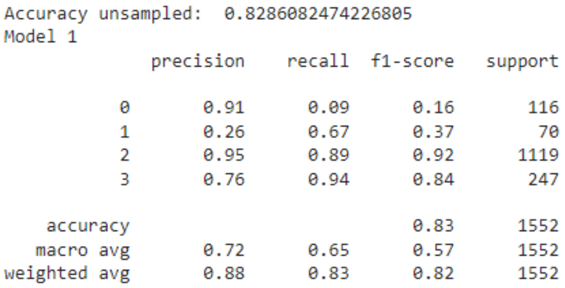
$$Recall \ = \frac{True\ Positive}{True\ Postive + False\ Positive} \quad (8)$$

The F1 score is the harmonic mean of precision and recall and combines the recall and precision to a single metric.

$$F1\ Score = 2 * \frac{Precision * Recall}{Precision + Recall} \quad (9)$$

A. KNN Unbalanced dataset

The results when using the unbalanced dataset without with the feature extraction.



```
Accuracy unsampled:  0.8286082474226805
Model 1
              precision    recall  f1-score   support

           0       0.91      0.09      0.16       116
           1       0.26      0.67      0.37        70
           2       0.95      0.89      0.92      1119
           3       0.76      0.94      0.84       247

    accuracy                           0.83      1552
   macro avg       0.72      0.65      0.57      1552
weighted avg       0.88      0.83      0.82      1552
```

Fig. 3 KNN Unbalanced Metrics

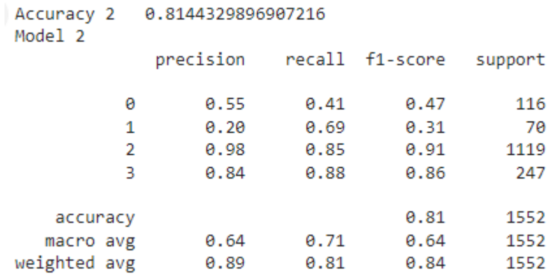
TABLE 3
CONFUSION MATRIX KNN, UNBALANCED DATASET

Feeding	10	36	28	42
NDM	1	47	22	0
Swimming	0	90	996	33
Resting	0	11	3	233
True/Pred	**Feeding**	**NDM**	**Resting**	**Swimming**

The label Feeding (0) is rated the lowest for recall and f1-score, which is to be expected because the feeding dataset was the minority followed by NDM (1).

B. KNN SMOTE

The results when using SMOTE, having the two lowest minority classes sampled to 10,000 samples.



```
Accuracy 2   0.8144329896907216
Model 2
              precision    recall  f1-score   support

           0       0.55      0.41      0.47       116
           1       0.20      0.69      0.31        70
           2       0.98      0.85      0.91      1119
           3       0.84      0.88      0.86       247

    accuracy                           0.81      1552
   macro avg       0.64      0.71      0.64      1552
weighted avg       0.89      0.81      0.84      1552
```

Fig. 4 KNN SMOTE Metrics

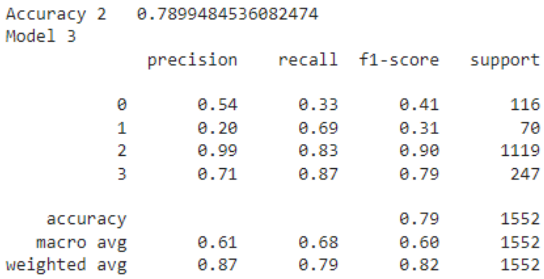
TABLE 4
CONFUSION MATRIX KNN, SMOTE DATASET

Feeding	48	40	10	18
NDM	15	48	7	0
Swimming	16	130	951	22
Resting	8	21	1	217
True/Pred	**Feeding**	**NDM**	**Resting**	**Swimming**

Although the precision decreased, recall and the f1-score showed significant improvements for the Feeding minority class, while the NDM class shows a decrease in precision and the f1-score, while showing marginal improvements in the recall.

C. KNN ADASYN

The results when using ADASYN, having the two lowest minority classes sampled to 10,000 samples.



```
Accuracy 2   0.7899484536082474
Model 3
              precision    recall  f1-score   support

           0       0.54      0.33      0.41       116
           1       0.20      0.69      0.31        70
           2       0.99      0.83      0.90      1119
           3       0.71      0.87      0.79       247

    accuracy                           0.79      1552
   macro avg       0.61      0.68      0.60      1552
weighted avg       0.87      0.79      0.82      1552
```

Fig. 5 KNN ADAYSN Metrics

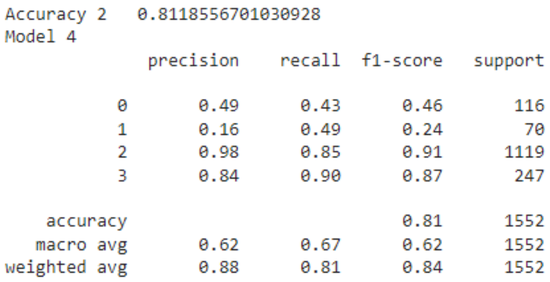
TABLE 5
CONFUSION MATRIX KNN, ADASYN DATASET

Feeding	38	34	8	36
NDM	13	48	4	5
Swimming	12	137	924	46
Resting	8	22	1	216
True/Pred	**Feeding**	**NDM**	**Resting**	**Swimming**

The metrics are lower when compared with the SMOTE sampling technique, although there are still improvements than the unsampled model.

D. KNN SMOTE-SVM

The results when using SMOTE-SVM, having the two lowest minority classes sampled to 10,000 samples. The samples generated when using SMOTE-SVM are the same sampled used to train the NN and LSTM.



```
Accuracy 2   0.8118556701030928
Model 4
              precision    recall  f1-score   support

           0       0.49      0.43      0.46       116
           1       0.16      0.49      0.24        70
           2       0.98      0.85      0.91      1119
           3       0.84      0.90      0.87       247

    accuracy                           0.81      1552
   macro avg       0.62      0.67      0.62      1552
weighted avg       0.88      0.81      0.84      1552
```

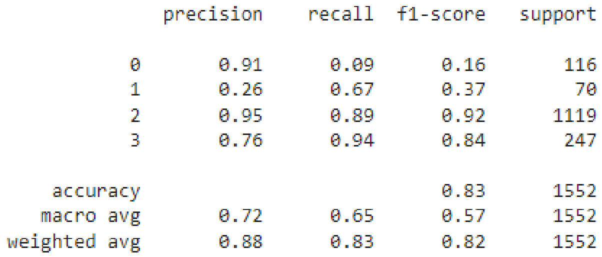
Fig. 6 KNN SMOTE-SVM Metrics

TABLE 6
CONFUSION MATRIX KNN, SMOTE-SVM DATASET

Feeding	50	32	10	24
NDM	30	34	6	0
Swimming	20	126	953	20
Resting	2	20	2	223
True/Pred	**Feeding**	**NDM**	**Resting**	**Swimming**

E. NN

The samples were trained on a standard neural network, the samples used were from the SMOTE-SVM.



```
              precision    recall  f1-score   support

           0       0.91      0.09      0.16       116
           1       0.26      0.67      0.37        70
           2       0.95      0.89      0.92      1119
           3       0.76      0.94      0.84       247

    accuracy                           0.83      1552
   macro avg       0.72      0.65      0.57      1552
weighted avg       0.88      0.83      0.82      1552
```

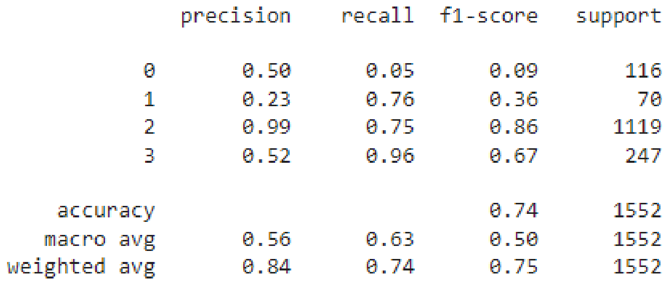
Fig. 7 Neural Network, SMOTE-SVM Dataset Metrics

TABLE 7
CONFUSION MATRIX NN, SMOTE-SVM DATASET

Feeding	3	26	24	63
NDM	1	18	19	31
Swimming	1	156	908	54
Resting	1	0	79	167
True/Pred	**Feeding**	**NDM**	**Resting**	**Swimming**

F. LSTM

The samples were trained on a standard LSTM network, the samples used were from the SMOTE-SVM.



```
              precision    recall  f1-score   support

           0       0.50      0.05      0.09       116
           1       0.23      0.76      0.36        70
           2       0.99      0.75      0.86      1119
           3       0.52      0.96      0.67       247

    accuracy                           0.74      1552
   macro avg       0.56      0.63      0.50      1552
weighted avg       0.84      0.74      0.75      1552
```

Fig. 8 LSTM SMOTE-SVM Dataset Metrics

TABLE 8
CONFUSION MATRIX LSTM, SMOTE-SVM DATASET

Feeding	6	15	2	93
NDM	1	53	7	9
Swimming	1	155	844	119
Resting	4	3	2	238
True/Pred	**Feeding**	**NDM**	**Resting**	**Swimming**

G. Comparing Models

The results showed that the SMOTE and SMOTE-SVM, table 4 and 6, had the best performances from the oversampling techniques. However, all oversampling techniques resulted in better results than the unbalanced KNN results. Both NN and LSTM had the worst results out of all the model trained as seen in table 7-8. The majority of classes, resting and swimming, had the best results in all model schemes.

An interesting result is that there was high misclassification of NDM and swimming class in all model schemes, indicating that there may be a strong correlation between those two classes.

IV. Conclusions

The results showed that the SMOTE and SMOTE-SVM had the best performances from the oversampling techniques, while the neural networks performed the worst. This may be due to a combination of factors which include: the oversampling techniques used, the features extracted, or the neural network themselves. However, the KNN significantly improved the recall and f1 score when compared to the unbalanced KNN. Which suggests that when using an unbalanced dataset with an oversampling algorithm, a KNN may be the best processes in which to create a predictive model.

Acknowledgment

This project wouldn’t have been completed without the help of Dr. Yeh guidance.

References

[1] Y. Yang *et al.*, “Feature extraction, selection, and K-nearest neighbors algorithm for Shark Behavior Classification based on Imbalanced Dataset,” *IEEE Sensors Journal*, vol. 21, no. 5, pp. 6429–6439, Mar. 2021. doi:10.1109/jsen.2020.3038660

[2] A. Gosain and S. Sardana, “Handling class imbalance problem using oversampling techniques: A Review,” *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, Sep. 2017. doi:10.1109/icacci.2017.8125820

[3] L. Demidova and I. Klyueva, “SVM classification: Optimization with the SMOTE algorithm for the class imbalance problem,” *2017 6th Mediterranean Conference on Embedded Computing (MECO)*, Jun. 2017. doi:10.1109/meco.2017.7977136

[4] Haibo He, Yang Bai, E. A. Garcia, and Shutao Li, “Adasyn: Adaptive Synthetic Sampling Approach for imbalanced learning,” *2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence),* Jun. 2008. doi:10.1109/ijcnn.2008.4633969