Identify Glitches in Gravitational Waves

Name:	**Pandey Satyam Dhiraj**
Registration No./Roll No.:	21198
Institute/University Name:	IISER Bhopal
Program/Stream:	Physics
Problem Release date:	17 August 2023
Date of Submission:	19 November 2023

1 Introduction

This project is focused on the identification of glitches in gravitational waves data, aimed at differentiating these anomalies from true gravitational waves data.Glitches happen frequently enough that they often can be coincident in the two detectors and can mimic astrophysical signals. Classifying and characterizing glitches is imperative in the effort to target and eliminate these artifacts, paving the way for more astrophysical signals to be detected.[1] There are 6000 training instances with eight features, among which six are numerical and 2 (named ifo and id) are categorical, with ifo having only two categories (L1 and H1) and a total of 22 classes.There is an high imbalance in the data set with suggests the use of random forest.
In this project, various machine learning models from the sklearn or the scikit learn library in python namely K-nearest neighbours, Support VectorMachines, Random Forest Decision Trees, Logistic Regression along with adaboost boosting method was used were used to train the data and develop an accurate model to classify glitches[3]. That model was then run on the test data. In another instance of the rum onehot encoding was used improved the result showing importance of ifo feature.Feature selection was also performed based on the correlation matrix.

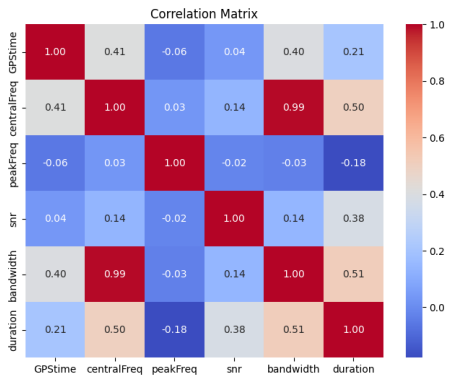

Figure 1: Correlation Matrix

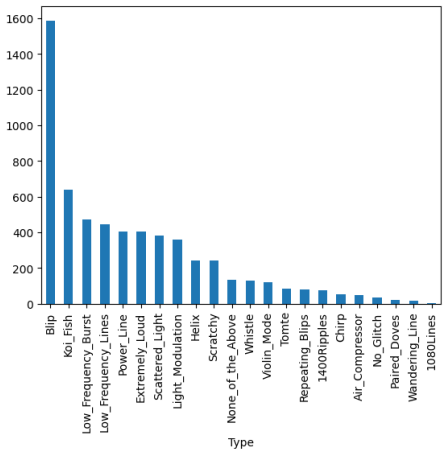

Figure 2: Overview of data

2 Methods

In the project, a comprehensive evaluation of multiple machine learning algorithms was conducted to classify data effectively. Initially, RandomForest, DecisionTree, KNN, Logistic Regression, and

Table 1: Performance Of Different Classifiers Using All Features

Classifier	Precision	Recall	F-measure
Decision Tree	0.699	0.697	0.690
K-Nearest Neighbor	0.645	0.532	0.550
Logistic Regression	0.500	0.437	0.485
Random Forest	0.844	0.794	0.805
Support Vector Machine	0.009	0.104	0.0139

SVM were assessed using the entine set of features. Subsequently, a grid search was performed on each method to fine-tune hyperparameters, aiming to optimize the F-measure, a metric capturing a balance between precision and recall. This iterative process continued until achieving optimal parameter settings, leveraging prior knowledge and recognized important hyper-parameters from similar projects.[2].

Data preprocessing involved removing the 'id' and 'gpstime' features, considering domain knowledge and their negligible contribution to the classification process. The initial grid search was performed on the complete training dataset after excluding the categorical feature 'ifo.' This step aimed to evaluate the models on the original data configuration. Further experimentation involved encoding the 'ifo' feature using One Hot Encoding and performing grid search which noticeably improved the performance. Additionally, due to the high correlation observed between 'bandwidth' and 'peakfreq' shown in figure 1 the correlation matrix, iterations were conducted, systematically dropping one of these features at a time. This process aimed to prevent potential overfitting caused by highly correlated features, thereby enhancing model performance.

Once the best hyper-parameters for the most effective models were determined, the ensemble technique Adaboost was employed to improve overall classification performance.

The github page for the code github

3 Experimental Setup

In the evaluation phase, various metrics like Macro-averaged F1 score, precision, and recall were employed on the test data, which was derived from the training dataset. These metrics were used in determining the most effective model for final evaluation. The dataset was divided into training and validation sets, and the Random Forest classifier emerged as the best-performing model based on the metrics.

In the configuration of the Random Forest classifier, certain parameters were set to specific values to optimize its performance. Notably, the maximum depth (max_depth) of individual trees was constrained to 22, ensuring the model's depth while preventing overfitting. Additionally, a minimum number of samples(min_sample_split) required to split an internal node was set to 2, a measure to maintain performance by preventing further splitting when the sample count is low. The forest's size($n\_estimators$), comprising the number of trees, was established at 110 Lastly, to augment the Random Forest classifier's performance, Adaboost, an ensemble technique, was applied with 50 estimators. Entropy was used as the criterion($cruterion$) to measure the information gain in the decision tree.

The library involving all the classifiers is SK-learn along with pandas for data manipulation and matplotlib and sns for data plotting.The confusion matrix for the best performing model is given below.

Table 2: Performance Of Different Classifiers Using All Features except bandwidth

Classifier	Precision	Recall	F-measure
Adaptive Boosting	0.861	0.795	0.815
Decision Tree	0.714	0.720	0.714
K-Nearest Neighbor	0.623	0.620	0.618
Logistic Regression	0.453	0.437	0.430
Random Forest	0.844	0.799	0.805

4 Results and Discussion

The final result after the prediction on the test data is represented in the below bar plot.
The confusion matrix figure 3 shown below is from the predictions on the validation data using the experimental setup with most successful i.e. Adaptive Boosting. The table above is evaluation metrics of the final model used to make predictions on the test data.

As we can see in the bar plot of overall data figure 2 and bar plot of predictions on test data figure 4, there is similar distribution of the type of glitches . Also the confusion matrix in figure 3 demonstrates our success rate of prediction on the validation data on different classes.

Table 3: Performance Of best Classifiers Using All Features except bandwidth

Classifier	Precision	Recall	F-measure
Adaptive Boosting	0.861	0.795	0.815

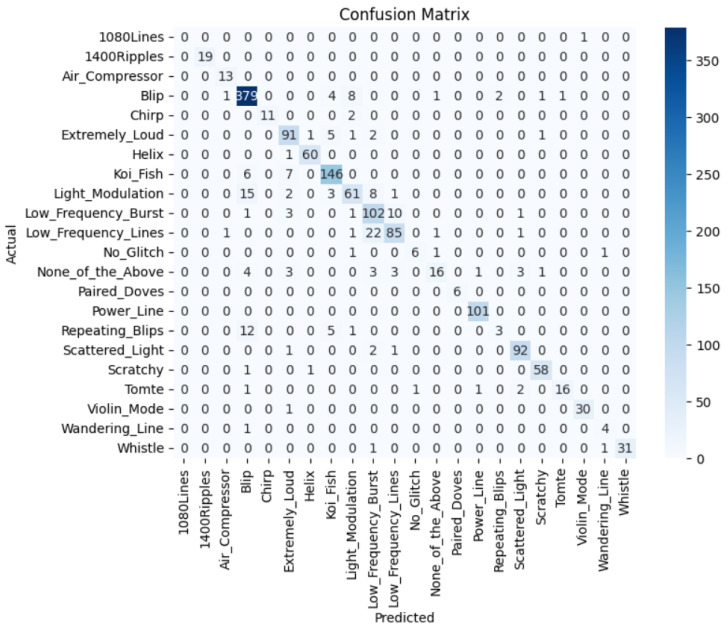

Figure 3: Confusion matrix for adaboost

5 Conclusion

The use of correlation between bandwidth and centralfreq for feature reduction by one for better performance is clearly demonstrated in the results.The distribution of glitches depicted in the bar plot figure4 highlights a substantial imbalance in the data,especially the low number non glitch events as compared to some of the more recurring glitches like blip.

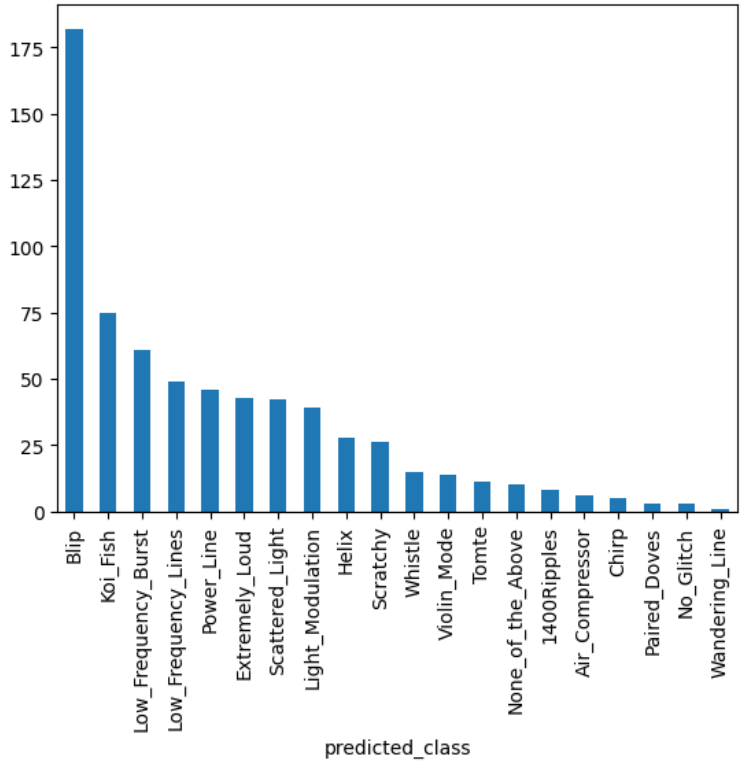

Figure 4: Class distribution in test data prediction

The challenge of creating a highly accurate glitch classification model is difficult due to the complexity inherent in identifying black hole mergers that generate gravitational waves. The rarity of 'No glitch' labels illustrates the infrequency of detection events. Consequently, even re-sampling techniques may not offer substantial insights into these rare detection events.

In conclusion, I believe comprehensive solution for the Gravitational Wave community involves a combitational approach that encompasses both resilient hardware advancements and the continued evolution of ML networks, potentially exploring newer fields like Spiking Neural Networks and Quantum Computation to tackle the complexities associated with gravitational wave detection.

References

[1] https://www.ligo.caltech.edu/page/gw-sources

[2] Robert E.Colgan, K.Rainer Corley, Yenson Lau, Imre Bartos, Imre Bartos, John N. Wright, Zsuzsa Márka and Szabolcs Márka (2020), Physical Review. Efficient gravitational-wave glitch identification from environmental data through machine learning

[3] R.Duda, P.Hart (2001), Pattern Classification, 2nd ed.