DETECTION OF EXTREME WEATHER EVENTS FROM ATMOSPHERICAL DATA

Hena Dhirajlal Ghonia
20213256
Kaggle Team Name: Hena
Username:henaghonia
https://www.kaggle.com/henaghonia

November 4, 2021

1 INTRODUCTION

This project aims to classify weather events such as i) standard ii) Tropical cyclone and iii) Atmospheric river using set of climate variables for different time, lattitude and longitude. Dataset used was subset of ClimatNet was part of Kaggle competition under course IFT6390. Baseline model was considered as Dummy classifier. Here I used Logistic regression for multiclass classification from scratch. Logistic regression is implemented using Stochastic gradient descent method to calculate weights. To improve performance on Kaggle leaderboard, more features were added and models such as Random forest, gradient boosting and Lightgbm was explored using 5 fold cross validation. Lightgbm being better performing model than other models explored gave 89% of accuracy on train set was achieved using Lightgbm and achieved 0.78415 on private leaderboard.

2 FEATURE DESIGN

2.1 Logistic regression(Baseline)

Data set included for training Logistic regression contained 16 atmospheric features, Lattitude and Longitude. All 18 features were normalized using min-max normalization:

$$x' = \frac{x - min(x)}{max(x) - min(x)}$$

where x is feature column and x' is normalized feature. Due to feature normalization logistic regression gave better results.

2.2 Other Models

New features were prepared by grouping over longitude and calculating minimum, maximum and mean of Z100 column(geopotential Z at 1000 mbar pressure surface [m]) as $Z1000\_min$, $Z1000\_max$, $Z1000\_mean$. Inclusion of these features improved score on Kaggle leaderboad. Reason behind grouping over longitude was observation obtained from SHAP plot. As shown is Figure 3- 'Shap plot of features for lightGBM' Longitude ('lon' column) and Z100 column contributes more towards Class 1, and hence to increase accuracy of Class 1 and 2 new features were constructed. However, aggregation for features such as T500 and PSL over longitude was tried but did not turn to improve model performance. Also, for this section(i.e other models) normalization was not performed since it degraded the results.

3 ALGORITHMS

3.1 Logistic regression(Baseline)

Logistic regression was implemented using Stochastic gradient descent and softmax as decision boundary. Since its a multiclass classification target value-LABELS columns was one hot encoded and weights for each target value was calculated having shape of 18(number of features)x3(number of classes). Initially model was tried for 350 iteration without adding regularizer.

3.2 Other Models

After feature preparation Random forest, LightGBM and gradient boosting was explored using sklearn and lightgbm library. For lightgbm, metric used is multi log loss and objective used was 'multiclass'. Initially for these models parameters were not tuned and were overfitting.

4 METHODOLOGY

4.1 Logistic regression(Baseline)

For model training 3 sets of data were prepared, Training, validation and test(unseen data-submitted to Kaggle). Training and validation data was split into 70:30 ratio inorder to include more samples on test set. Hyperparameter tuning i.e learning rate was tuned on validation set. Metric used was accuracy for logistic regression. Increasing number of iteration improved accuracy of class 1 and class 2. After manually trying values of learning rate as [0.001, 0.002, 0.003, 0.01] for 30 iteration since lower learning rate would take more time and higher learning would overshoot the minimum and fail to converge. As shown in Figure 4, 0.001 smoothly decreases whereas 0.01 has sudden drop for loss value. Hence, Learning rate as 0.002 and 10000 iteration was used for model. L2 Regularizer was added to make model more robust.

4.2 Other Models

5 fold cross validation was performed for Lightgbm, gradient boosting and Random forest and hyperparameter tuning was performed. Dataset was randomly split in 70:30 ratio into training and validation set.
For Random forest, hyperparameter tuning was done for max_depth (more the value more complex the model), max_features, n_estimators and min_samples_leaf(reduce overfitting).
For gradient boosting, hyperparameter tuned were learning_rate, max_depth and min_samples_leaf(reduce overfitting).
For Lightgbm, class_weight(to handle class imbalance) and min_gain_to_split(reduce overfitting) was tuned.

5 RESULTS

5.1 Logistic regression(Baseline)

Accuracy obtained for train set: 82.21% and for validation set: 81.37% As shown 1 Loss value for on train set for logistic regression decreases over time.

5.2 Other Models

Metric used for other models was F1 score for each class. F1 score on validation set

Model/F1score	LightGBM	Random forest	Gradient Boosting
Class 0	0.93	0.92	0.93
Class 1	0.75	0.60	0.70
Class 2	0.73	0.58	0.69

F1 score on Training set

Model/F1score	LightGBM	Random forest	Gradient Boosting
Class 0	0.94	0.92	0.95
Class 1	0.81	0.65	0.79
Class 2	0.78	0.62	0.78

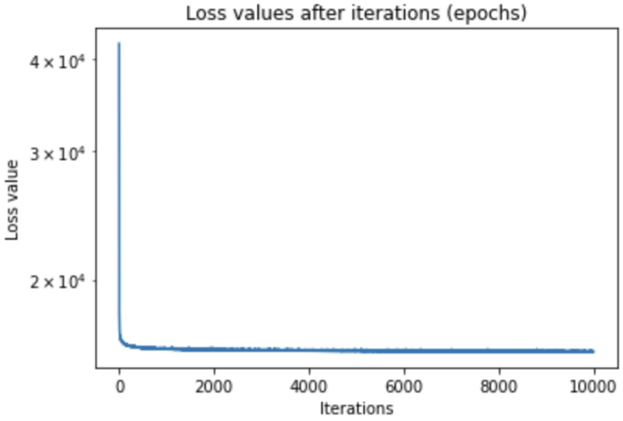
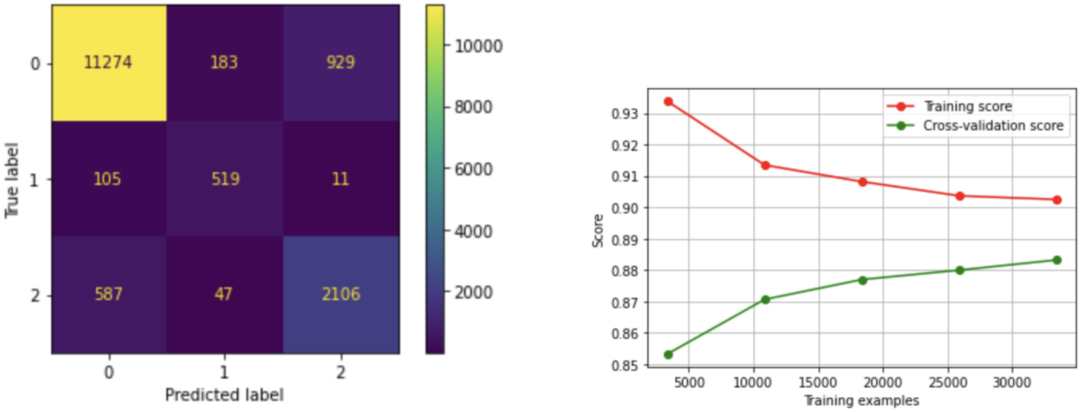


Figure 1: Loss value vs iteration



(a) Confusion matrix

(b) Learning curve for lightgbm 2

Figure 2: 2 Model Results for Lightgbm

Confusion Matrix for Lightgbm model and learning curve for Lightgbm. For learning curve on x-axis, number of training examples and on y-axis; F1-score is plotted. In Learning curve large gap between training score and Cross validation score is observed which means there is high variance and data needs to improve or model can be simplified with fewer features.

Importance of feature decreases as move below in below figure. Shap plot in Figure 3 indicates Z100_mean to be most important and contributes more to prediction of Class 2 compared to Class 1 and Class 0. Similarly contribution of feature feature towards class prediction can be observed from SHAP plot

6 Discussion

Pros of Logistic regression:Adding Regularizer gave better performance and since data is less simple model like Logistic regression would perform better than complex model.

Cons: Here stochastic gradient descent took more time, instead mini batch can also be used to achieve similar performance.

Pros for other models: Since creating new features from already existing data gave better performance, more features can be prepared considering different location and inter correlation between variables. After creating more number of features, feature selection method such as chi square, forward or backward selection can be used.

Cons for Lightgbm: It was best performing model compared to others but was overfitting between train and validation was observed. Although increasing min_samples_leaf value which would help reduce overfitting did not improve overall accuracy. Hence tradeoff of observed between F1 score of class 1 and 2 and increasing reducing overfitting.

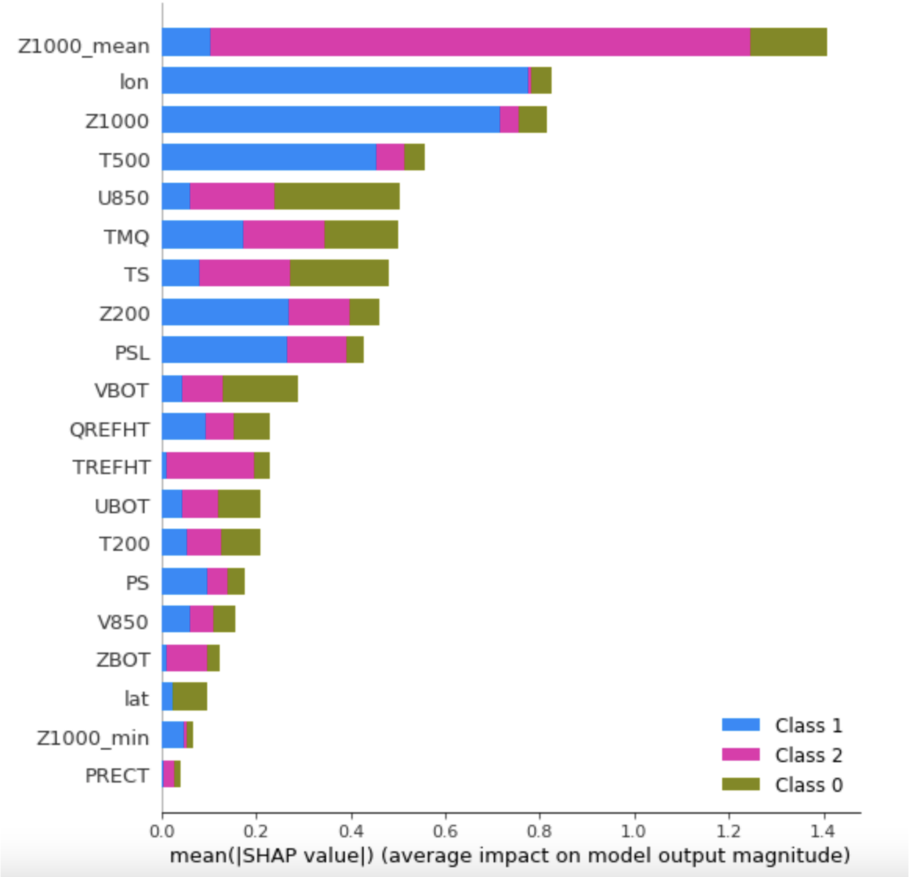


Figure 3: Shap plot of features for lightGBM.

Ideas for improvement: Interpretation of model helped in improving the overall performance and hence it can be iteratively used to determine which model to choose. Moreover, it gave idea about how each feature is related in predicting particular class. More data can be used to improve predictive power of model and techniques to handle imbalance can be explored.

7 Statement of Contributions

I hereby state that all the work presented in this report is that of the author

References

[1] https://www.kaggle.com/c/ift3395-6390-weatherevents/overview

[2] https://en.wikipedia.org/wiki/Feature_scaling

[3] https://lightgbm.readthedocs.io/en/latest/pythonapi/lightgbm.LGBMClassifier.html

[4] https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.GradientBoostingClassifier.html

[5] https://towardsdatascience.com/l1-and-l2-regularization-methods-ce25e7fc831c

[6] https://github.com/slundberg/shap

[7] https://github.com/slundberg/shap

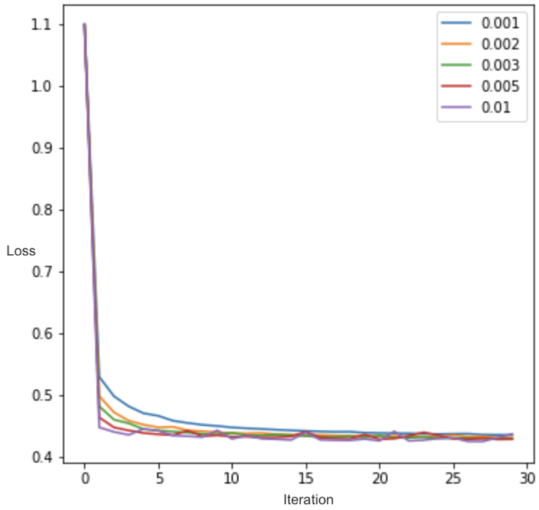


Figure 4: Loss value vs iteration for logistic regression for each learning rate value

8 Appendix

report.html was generated using pandas-profiler library to visualize the correlation, duplicates, and uniqueness in Dataset provided.