Comparison of the performances of XGBoost, Adaboost, LightGBM, Catboost, Decision Tree and Gaussian Process classifiers in an inter-sentential relation extraction setting

Shraddha Satish Thumsi, Liliana Salas

December 12, 2019

1 Introduction

In Natural Language Processing (NLP), the problem of associating a context to an event has been studied in detail, especially in the realm of cancer research. Much of the literature found in this space involve the context and event mentions lying in the same sentence [4]. However, a non-trivial problem of extracting the relationship between events and contexts in an inter-sentential setting could be more common [5]. Reach is an NLP processor that extracts the events and contexts in the paper and the Machine Learning component in Reach classifies these to be associated as a context or not, i.e., it predicts a true/false value for a given pair of event-context mentions using the linguistic features in the text. Efforts in exploring a suitable classifier for this task have been made by the BioContext team in the University of Arizona [1]. This project report furthers this effort by exploring more classifiers. We also provide an analysis of the time taken by each algorithm compared to the F1 score that it yields.

2 Dataset description

In this project, we use a .csv file wherein the event-context pairs have numerical and boolean values for linguistic features in the text in which they appeared. The "ground truth" for these pairs were the annotations provided to us by domain experts in 2016. Each datapoint is characterized by three strings: paperID (PMCID of the open-access paper), event ID (the sequence of words depicting a BioEvent in Reach) and context ID (the name of a biological entity that may be marked as context, i.e. a BioTextBoundMention). As we mentioned above, we are dealing with an inter-sentential problem, i.e. the event and context mentions may lie many sentences apart in the text of the paper. The feature values for each pair are values of linguistic features such as sentence distance, dependency distance, whether the context mention is the closest one to the event and so on. The features for which we extract values were used in [1]. The data set has 5092 datapoints and each datapoint has 999 non-identifying and non-label features. The labels in the data set are of boolean value marking whether or not the BioTextBoundMention is a true context of the event mention, and this is what our classifiers will predict.

3 Methodology used for comparison of models

Based on the description of the dataset, we chose the pairs from a given paper to be the testing set while the remainder of 21 papers were used as a training set in the cross validation [1]. The number of event-context

pairs were found to differ widely across papers, therefore we used micro-averaging to compare the predicted values to the "ground truth".

4 Overview of classifiers

4.1 XGBoost

"XGBoost is the portmanteau of "extreme gradient boosting". It is an upgrade to the gradient boosting algorithm focusing on computational speed and model performance, including new features like regularization (both L1 and L2). The model supports features of the library Scikit-learn and also libraries in R. Being advantageous for computational speed, its highlights are that it can be used for parallel programming and can be cache-optimized. It is "sparse-aware", making it easy to handle sparse or missing data. This algorithm goes by lots of different names such as gradient boosting, multiple additive regression trees, stochastic gradient boosting or gradient boosting machines" [6]. "Boosting in general is an ensemble model that corrects the errors made by other models in future predictions. Models are added sequentially until no further improvements can be made. Boosting algorithms are useful when there is limited training data, limited training time and little expertise in parameter tuning" [7]. In this analysis, we used the default settings of the hyperparameters for XGBoost [8].

4.2 Adaboost

Adaboost is also an ensemble algorithm. "The output of the other learning algorithms ('weak learners') is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers. AdaBoost is sensitive to noisy data and outliers. In some problems it can be less susceptible to the overfitting problem than other learning algorithms. The individual learners can be weak, but as long as the performance of each one is slightly better than random guessing, the final model can be proven to converge to a strong learner" [9]. It is a part of the *sklearn.ensemble* package. We specified the parameter $n_{estimators}$ (no. of estimators) to be 32, but all others were default parameters [10]. We chose this value to stay close to the number of estimators in the Decision Tree classifier, i.e. square root of the number of numerical features in our dataset (999).

4.3 LightGBM

"LightGBM is a fast, distributed, high performance gradient boosting (GBT, GBDT, GBRT, GBM or MART) framework based on decision tree algorithms used for ranking, classification and many other machine learning tasks [11]". It was developed by Microsoft. As for hyperparameters, we set the "objective" hyperparameter to "binary" since this is a binary classification problem. Additionally, we set the *randomstate* to 5. *randomstate* sets the seed for the algorithm [12].

4.4 Catboost

"Catboost is a high-performing gradient boosting algorithm for decision trees. CatBoost has the flexibility of giving indices of categorical columns so that it can be encoded as one-hot encoding using $one-hot-max-size$ (Use one-hot encoding for all features with number of different values less than or equal to the given parameter value). If you don't pass any anything in the $catfeatures$ argument, CatBoost will treat all the columns

as numerical variables" [7]. We used the following hyperparameter settings: $iterations = 50, depth = 3, learningrate = 0.1, lossfunction = Logloss$.

4.5 Decision Tree

"Decision tree learning is a supervised machine learning technique for inducing a decision tree from training data. It is a predictive model which is a mapping from observations about an item to conclusions about its target value. In the tree structures, leaves represent classifications (also referred to as labels), nonleaf nodes are features, and branches represent conjunctions of features that lead to the classifications" [13]. We used 32 estimators and a depth of 8 for the decision tree. All other hyperparameters were the default value as specified by sklearn.

4.6 Gaussian process

"Gaussian Processes (GP) are a generic supervised learning method designed to solve regression and probabilistic classification problems. GaussianProcessClassifier places a GP prior on a latent function , which is then squashed through a link function to obtain the probabilistic classification. The latent function is a so-called "nuisance" function, whose values are not observed and are not relevant by themselves. Its purpose is to allow a convenient formulation of the model, and is removed (integrated out) during prediction. GaussianProcessClassifier implements the logistic link function, for which the integral cannot be computed analytically but is easily approximated in the binary case" [14]. We used the default settings of the hyperparameters provided in the documentation.

5 Performance of models

Each of the six chosen algorithms were compared on Precision, Recall, F1 and Accuracy scores. The individual scores obtained by the models are plotted in Fig 1.

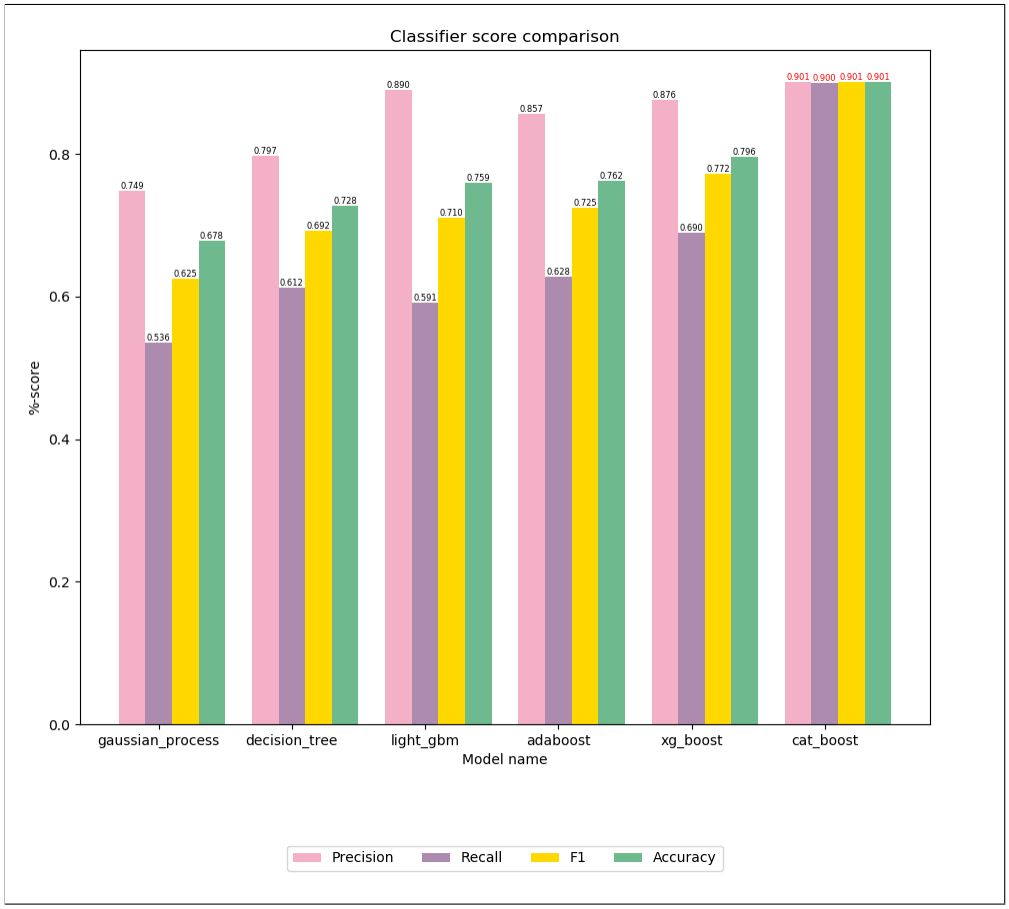

Figure 1: Results from the Reach 2019 dataset that presented six competing classification methods for the BioContext event-context classification experiment

As an additional analysis, we compared the amount of time in minutes taken by each classifier to achieve the F1 score mentioned above. The time v/s F1 graph is presented in Fig 2.

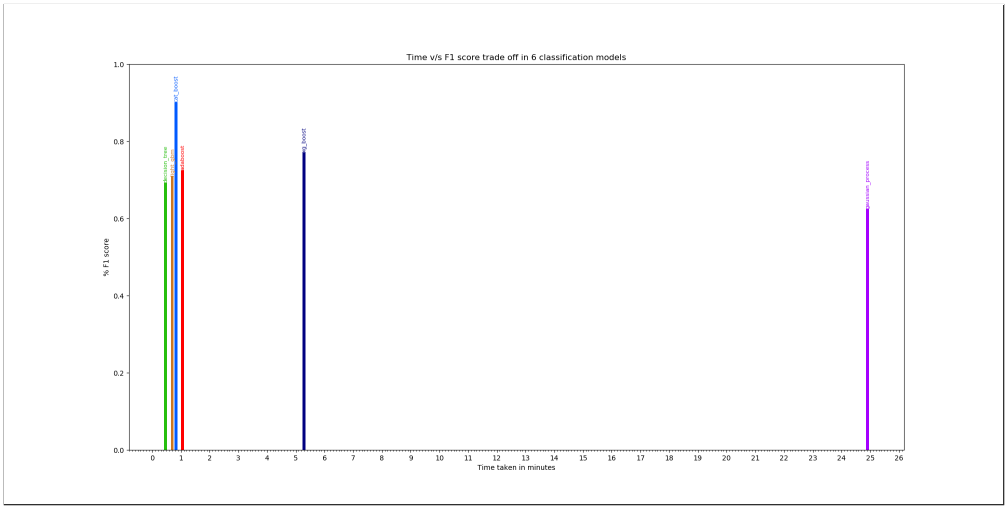

Figure 2: A comparison of the time-F1 tradeoff for each of the 6 classifiers

6 Results

Over the 6 classifiers we compared, Catboost had the highest F1 (0.90). It took about 0.85 minutes to finish testing over all papers in the cross-validation loop. Gaussian Process classifier was the slowest and had the least F1 score (25 minutes and 0.625 respectively). Four out of the six classifiers finished the cross-validation loop in under a minute.

7 Acknowledgements

We want to thank Dr. Clayton Morrison for his support and guidance through the project, both in the scope of the project and in the finer details of it. A special thank you to Paul D Hein for providing us the starter code for the hyperparameters on some classifiers as well as some starter code for the scoring graph. The Sklearn package makes an AI practioner's life easier by providing examples, documentation and code for easy access. I also want to thank Liliana for having allowed me to walk with her through the code as well as the math involved in not only the project, but also the entire course. I have learned more from her than she might have from me.

References

[1] Noriega-Atala, E., Hein, P. D., Thumsi, S. S., Wong, Z., Wang, X., & Morrison, C. T. (2018). Inter-sentence Relation Extraction for Associating Biological Context with Events in Biomedical Texts. arXiv preprint arXiv:1812.06199.

[2] Valenzuela-Escárcega, M. A., Babur, O., Hahn-Powell, G., Bell, D., Hicks, T., Noriega-Atala, E., & Morrison, C. T. (2017). Large-scale automated reading with Reach discovers new cancer driving mechanisms. In *Proceedings of the Sixth BioCreative Challenge Evaluation Workshop*. Bethesda (pp. 201-3).

[3] Kim, Y. (2014). Convolutional neural networks for sentence classification. arXiv preprint arXiv:1408.5882.

[4] N. Bach and S. Badaskar, "A review of relation extraction," *Literature review for Language and Statistics II*, 2007.

[5] K. Swampillai and M. Stevenson, "Extracting relations within and across sentences," *in Proceedings of Recent Advances in Natural Language Processing*, 2011.

[6] https://machinelearningmastery.com/gentle-introduction-xgboost-applied-machine-learning/

[7] https://towardsdatascience.com/catboost-vs-light-gbm-vs-xgboost-5f93620723db

[8] https://xgboost.readthedocs.io/en/latest/parameter.html

[9] https://en.wikipedia.org/wiki/AdaBoost

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

[11] Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree. NIPS.

[12] https://lightgbm.readthedocs.io/en/latest/Parameters.html

[13] Tan, Lin. (2015). Code Comment Analysis for Improving Software Quality. The Art and Science of Analyzing Software Data. 493-517. 10.1016/B978-0-12-411519-4.00017-3.

[14] https://scikit-learn.org/stable/modules/gaussian_process.html