By: Swarup Mishra

FILE TYPE IDENTIFICATION

Introduction

Used the popular Content-based file type identification method which uses a byte frequency distribution with classical machine learning techniques to classify file types. Almost all algorithms on the internet use the complete file content to get the byte frequency distribution. This technique is not optimised and hence consumes more time. Using a technique(i.e. select a subset of byte values based on the frequency of occurrence) can help in reducing the classification time.

Approach

Existing file type identification approaches generate features from the byte frequency distribution of the complete file but that will slow the classification process to a great extent. So, I use a technique that selects some of the most frequent byte patterns in each file type. I use the Counter class in the collections module that allows me to easily count the occurrence of individual elements in an iterable object, such as the byte contents. Then used a “most frequently occurring byte selection” method that assumes that a few of the most frequently occurring byte patterns are sufficient to represent the file type. Since each file type has a different set of high-frequency byte patterns, classification uses the lists of most frequently occurring byte patterns. Performed data preprocessing, exploratory data analysis to find patterns in data and boxplot to get outliers. Finally performed feature scaling, feature engineering techniques like encoding to get efficient features and then performed feature selection to get the feature correlating the most with our target. Experimental tests of the techniques involve four classification algorithms: k-means algorithm, support vector machine, logistic regression and random forest classifier.

Findings

Since the file extension and magic numbers can easily be changed, file type identification must only rely on the file contents.The results of comparing six file types (ml, kt, rexx, csproj, jenkinsfile and mak) show that SVM(SVC) algorithms achieves an accuracy of about

61% on train, 55% on validation and 52% on test data using only 5 most frequently occurring byte values for each file for each type.

Scope Of Improvements

The data analysis performed to understand the dataset before building any models can be more precise and detailed. Pre-processing the dataset done before the splitting with some data scaling technique and filling all infinite values with NaN and then filling the NaN with mean will surely help to fit the data for the model training. Approaches used for training the model can be fine-tuned to perform better. And hence while validation that would make the accuracy_score of the model better. The hyperparameter tuning would also be a good way to improve models performance. Improve the feature engineering approach by using the domain knowledge to bring some more features along with encoding my data to improve the accuracy. Can write the code to remove outliers using z-score or IQR that I found during data analysis using boxplot. Finally I shall build a perfect Flask Api with some html web works to wrap the model and show the desired output using curl command on the website that will help in productization.

Conclusion

The techniques described above are designed to speed up file type identification. The first “most frequently occurring byte selection” technique performs byte selection and only uses some of the most frequently occurring byte patterns for classification and thus speeds up the classification.

References

1. FAST CONTENT-BASED FILE TYPE IDENTIFICATION Irfan Ahmed, Kyung-Suk Lhee, Hyun-Jung Shin and Man-Pyo Hong.[link]
2. Byte Histogram in Python3 by Justus Perlwitz [link]