**Mayank Mahajan**

**Comparison of Classifiers for Detection of True Complex Variants**

**1 Introduction**

Complex variants are a relatively newly discovered phenomenon and there is still a significant amount of knowledge yet to be gained regarding these special characteristics. Complex variants are categorized by sites of DNA that have multi-allelic variation, with at least 1 insertion or deletion. Preliminary research has shown that the presence of these complex variants could indicate larger health effects, and it is important to then explore the different factors that may correlate to the existence of a complex variant to assist in the prediction and diagnosis of certain genetic disorders (1).

With the advent of machine learning, it has become much easier to study different features of a dataset to train a model that can accurately classify further data points. Different pieces of the pipeline, like dimensionality reduction and classification, serve to increase the accuracy and decrease the complexity of the resulting model in order to create the most efficient set of methods. This study applied machine learning techniques to the problem of identifying complex variants. Several different methods of dimensionality reduction and classifiers were tested to find the optimal pipeline for determining the presence of complex variants.

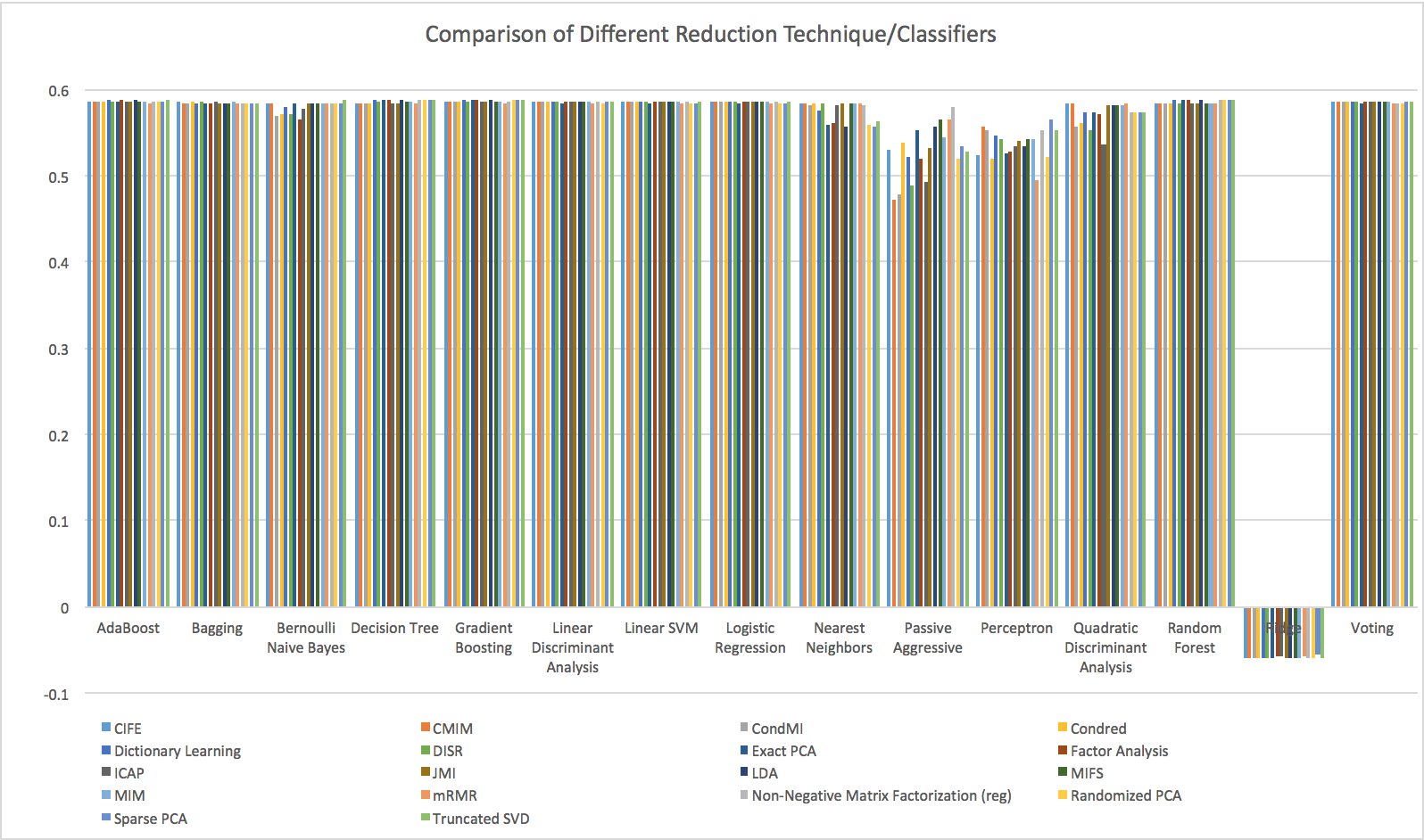
**2 Methods**

The process of dimensionality reduction was performed using already existing Python packages, specifically PyFeast and scikit-learn. Attribute selection techniques, although not used in the final pipeline, were tested using the Weka and JavaML packages, both written in Java. The classification techniques were selected from scikit-learn, and a variety of classifiers were chosen to make for a more comprehensive survey. 10-fold cross validation was used for all test runs.

The data itself was comprised of 710,000 labeled samples, totaling over 120 MB, from the Complete Genome Pipeline. For each sample, 42 features were used – these features represented 6 cell lines each annotated with 7 different chromatin features using the ChromHMM software.

The pipeline itself was split into four different Python files for simultaneous execution; the files were split based on the percent of features (10% or 50%) to keep after dimensionality reduction, and the source of the dimensionality technique (PyFeast or scikit-learn). In total, there were 480 runs completed with different combinations of dimensionality reduction techniques, classifiers, and percent of features.

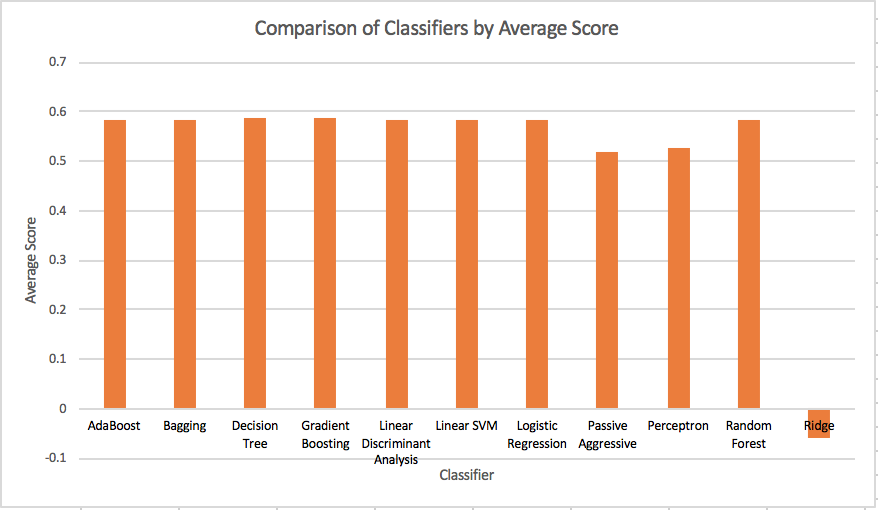
**3 Results**

****

**Fig. 1**. Comparison of Different Reduction Technique/Classifiers.

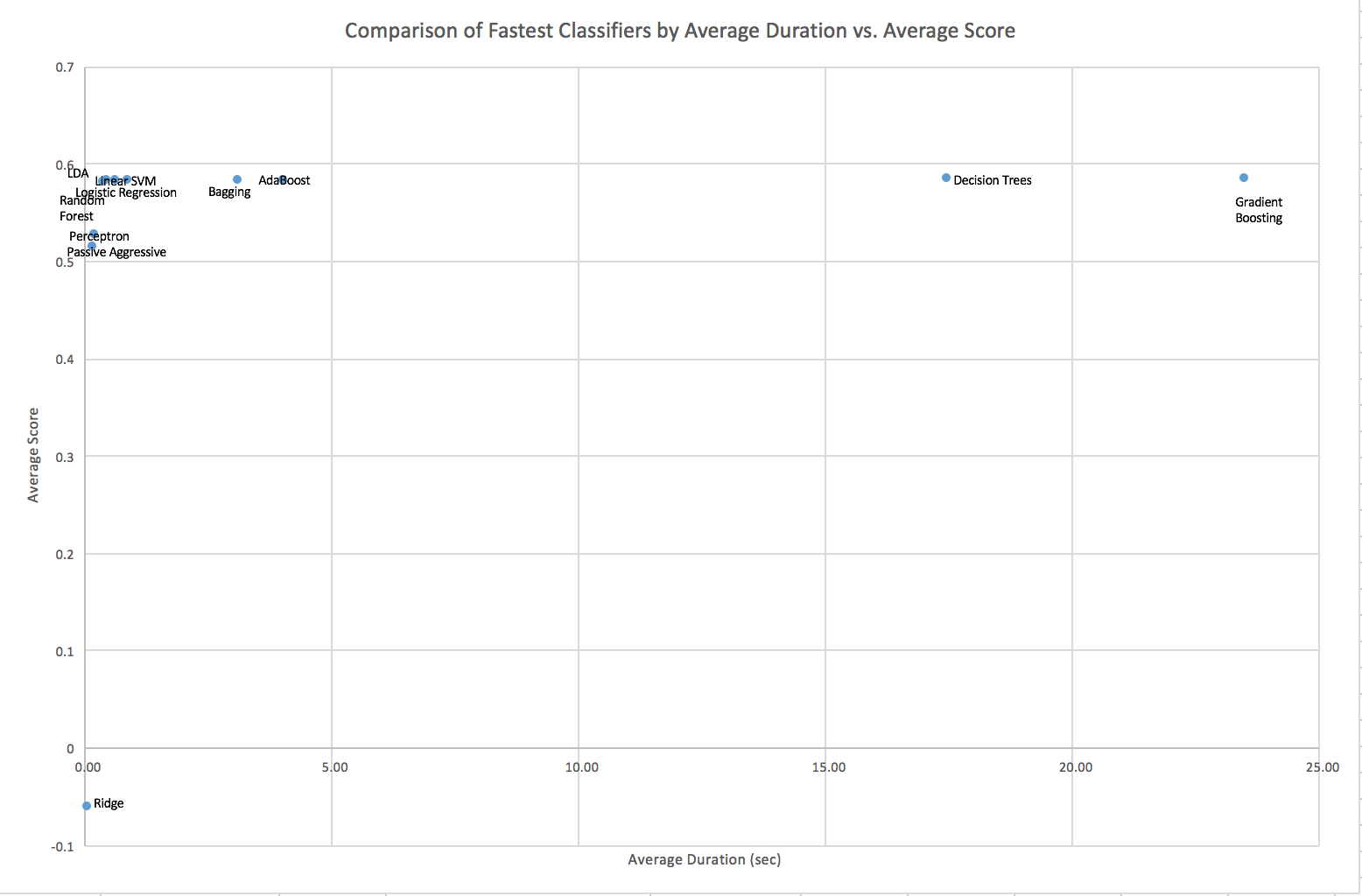
Figure 1 summarizes the results of the test runs. Each bar represents the best score for that combination of dimensionality technique and classifier, based on 10-fold cross validation. The score represents the model’s accuracy in correcting predicting the presence or lack thereof of a complex variant in a DNA fragment. The runs shown are grouped by classifier.

According to the results, neither dimensionality reduction techniques nor classifiers seem to have a strong effect on the model’s ability to predict the existence of a complex variant. With the exception of the ridge classifier, which resulted in negative scores for all runs, the majority of classifiers achieve a similar score at the top end. Some classifiers, like the Passive Aggressive and the Perceptron, are less precise and give varying scores depending on the reduction technique, yet the majority of classifiers consistently receive almost exactly the same accuracy regardless of reduction technique. Most runs achieved scores ranging between 0.55 and 0.6.



**Fig. 2**. Comparison of Classifiers by Average Score

Figure 2 summarizes the average score of each classifier used in this study. The score represents the average ability of the classifier to correctly predict the presence of a complex variant. Figure 2 condenses Figure 1 down to one average measure for each classifier, and it is even clearer from Figure 2 that the choice of classifier does not seem to have a significant effect on accuracy. The average for almost every classifier is around 0.58, with exceptions for the Nearest Neighbor, Perceptron, Passive Aggressive, and Ridge classifiers.



**Fig. 3**. Comparison of Fastest Classifiers by Average Duration vs. Average Score

Figure 3 shows a scatterplot of the fastest classifiers and graphs their average running time against their average accuracy.

Figure 3 indicates a lack of relationship between average runtime and average score for different classifiers. Notably, the LDA, Logistic Regression, and Linear SVM classifiers achieve relatively high average scores with very small running times, indicating that those classifiers show promise as effective classifiers for further machine learning pipelines with complex variant data.

**Acknowledgements**

I acknowledge Md. Sharif Islam Bhuyan for serving as my direct mentor throughout the duration of this project, and for coordinating communications between my professor and I. I also acknowledge Professor Itsik Pe’er for being my research advisor and assigning me this project. Lastly, I acknowledge Aman Chahar for his help with parts of the project relating to PyFeast and other general troubleshooting during the execution of the tests.

**Funding**

This work has no sources of funding.

*Conflict of interest: none declared.*

**References**

1. Bhuyan, M.S.I. and Pe’er, I. (2015) *A Map of Complex Variants in Human*. MS. Department of Computer Science, Columbia University School of Engineering and Applied Science, New York, NY, 10027, USA.