A Comparison of Supervised Learning Algorithms

Sarah Borsotto

Abstract

Over time, the evolution of machine learning models has led to the creation of novel approaches derived from existing ones, leading to improved performances on various classification problems. This study empirically compares three of these algorithms, logistic regression, decision trees, and random forests, within classification tasks across diverse datasets. Their implementations are evaluated based on classification accuracy, a precision metric used to discern the models' strengths and limitations. Prior studies have also assessed these models against each other, but this study extends the analysis to various real-world datasets, highlighting their applicability and trade-offs in different scenarios. The findings can be used to suggest possible methods of analysis for future classification ventures given a dataset with certain characteristics.

Introduction

The study by Caruana and Niculescu-Mizil (2006) provides a foundational exploration into the comparative analysis of diverse classification models, emphasizing the predictive accuracy and interpretability of more advanced machine learning algorithms, such as SVM, against traditional linear models like logistic regression. The strongest models were found to be random forests, neural nets, and bagged trees, while the weakest models were logistic regression, decision trees, and naive bayes. Building upon the study's groundwork, my research seeks to further explore the performance evaluation of logistic regression, decision trees, and random forests. While prior research has shed light on the comparative performance of these models, this study intends to expand on these findings by incorporating newer datasets, exploring varying hyperparameter configurations, and considering the implications of model interpretability and scalability. The differences in achievement for these models are inspected in detail.

There are numerous similarities and distinctions between logistic regression, decision trees, and random forests. Logistic regression predicts binary outcomes by applying a logistic function to a linear combination of input features, thereby drawing a hyperplane that separates classes. Decision trees generate flowchart-like structures by splitting data using feature values as decision boundaries, leading to non-linear relationships that may overfit the data. Random forests aggregate predictions from multiple decision trees, lessening overfitting concerns by combining multiple decision-making strategies and voting on their results. Logistic regression draws linear boundaries, decision trees craft flowcharts, and random forests blend numerous flowcharts, each model offering distinct advantages and limitations when making predictions. While logistic regression is an effective classifier that can be easily interpreted, it may lead to poorer performance when the data does not contain linear relationships between variables. Accordingly, more sophisticated algorithms that can better handle non-linear relationships are preferred in these circumstances, such as decision trees. These decision trees can be further enhanced through random forests, which take into account any variations in decision-making done by the decision trees. These models may perform better on more intricate data, but they are harder to interpret given the complexity of decision trees when classifying data into groups. In order to see this in action, I propose a thorough experiment that directly contrasts these models against each other for three different datasets.

Methodology

In my experiment, I utilized Python's scikit-learn library to compare the performances of three classification algorithms, logistic regression, decision trees, and random forests, for three separate datasets: each extracted from UCI's machine learning repository. First, I explored, cleaned, and prepared each dataset for analysis. I removed null values and converted categorical data into numerical data by binarizing, one hot encoding, and ordinal encoding. I also ensured that all datasets represented two-class classification tasks by investigating the prediction variable of interest and transforming classes into either 0s or 1s. I divided the datasets into training and test sets, partitioning on 20/80, 50/50, and 80/20 training-testing sizes. For each algorithm, I defined a grid of hyperparameters to explore using GridSearchCV, a technique that exhaustively searches through various combinations of hyperparameters using cross-validation. This technique determines the optimal set of hyperparameters that maximizes the model's accuracy on the training data by averaging across three cross-validation sets. Subsequently, I trained each model using the best parameters obtained from the grid search on the training set and evaluated their performance on the test set in order to effectively compare the machine learning models. A summary of the hyperparameters incorporated into the gridsearch for each model is described below.

Logistic Regression: Inspected regularization strength C in intervals of 0.001, 0.01, 0.1, 1, and 10 to regulate the balance between fitting the training data and overfitting. Included l1 and l2 penalty for the type of regularization applied, as well as liblinear, lbfgs, newton-cg, sag, and saga for the solver parameter, which differ in behaviors based on the data's size.

Decision Trees: Selected low values for max depth to prevent overfitting, including none, 5, 10, and 15. Hindered overfitting by choosing 2, 5, and 10 for minimum samples split and 1, 2, and 4 for minimum samples leaf. Lastly, used gini and entropy for the criterion.

Random Forests: Promoted higher performance with 50, 100, and 200 number of trees. I chose the same values as the decision tree hyperparameters for maximum depth of trees, minimum samples split, minimum samples leaf, and criterion.

Performance metric:

Extracted mean accuracy of the model using sklearn's score attribute. This classification accuracy is a straightforward metric that measures the proportion of correctly classified samples out of the total samples in the dataset. While this can be an accurate representation of the model's performance, issues may arise when there is an imbalance in the data.

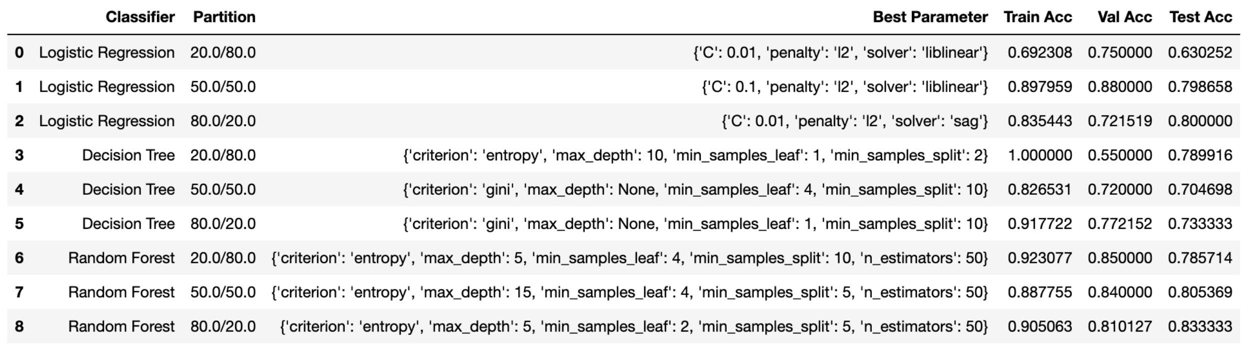
Datasets:

In an attempt to thoroughly investigate the performances of each classifier on various classification tasks, I selected three diverse datasets from UCI's machine learning repository. The first dataset is the Heart Disease dataset, which comprises information from the Cleveland Clinic Foundation, containing 303 rows and 13 features. These attributes include a mix of categorical and numerical features such as age, sex, chest pain type, blood pressure, cholesterol levels, electrocardiographic results, heart rate, and more. The dataset's primary objective is to predict the presence or absence of heart disease (indicated by 1 or 0, respectively) based on the provided patient features. The second dataset is Adult, encompassing approximately 45,222 rows with 14 features. These attributes cover details like age, education, occupation, marital status, and more, with the target variable indicating whether an individual earns above or below $50,000 annually. Lastly, the third dataset is the Credit Approval dataset, containing information on individuals applying for credit and their approval status. It includes numerous attributes such as age, income, employment details, credit history, loan amount, and more. The dataset's primary objective is to predict whether an individual's credit application will be approved or denied based on these features. There are 653 observations and 15 columns with a variety of categorical and continuous variables. While the first and third datasets are mostly balanced, the second dataset is severely imbalanced between classes, with 34014 observations below $50,000 and 11208 observations above $50,000. As a caution for this, I include the class weight attribute as “balanced” in all of the models.

Experiment

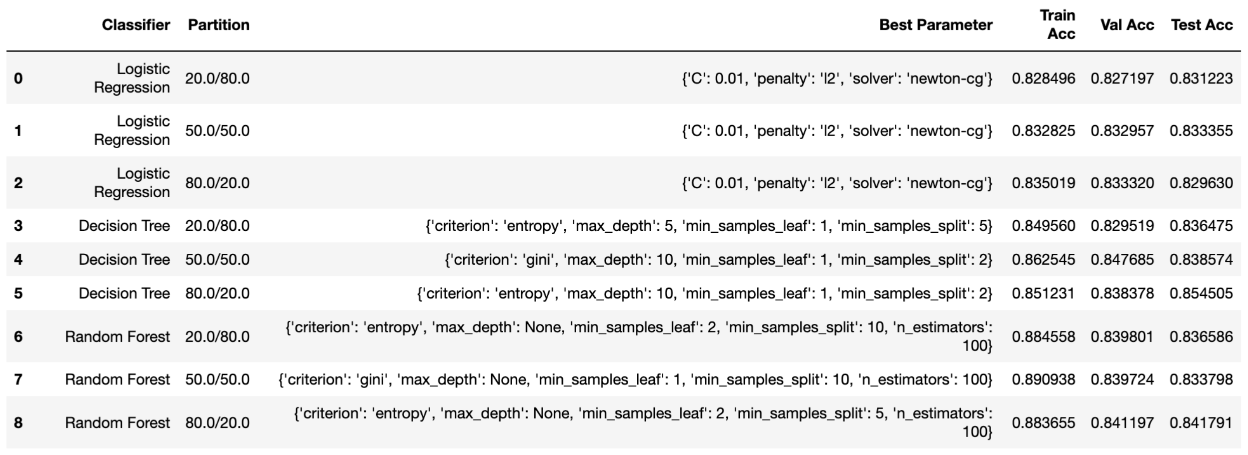
For each dataset, I obtained the variables with the highest correlations to the prediction variable and converted them to numerical variables when necessary. I executed any data cleaning needed to prepare the data for analysis. In the heart disease dataset, I included cp, ca, thal, exang, thalach, and oldpeak without any alterations to the original data. I did, however, convert all dependent values to 0 (no-presence) and 1 (presence). As for the adult dataset, I included age, education-num, marital-status, sex, hours-per-week, and capital-gain, with an explanatory variable of 0 (less than $50,000) and 1 (more than $50,000). The credit-approval dataset did not have any descriptions for the variables, but I incorporated four variables that showed high correlation to the predictor variable, where 0 was not approved and 1 was approved. Once the datasets were prepared, I partitioned the data into three training-testing sizes: 20/80, 50/50, and 80/20. I performed three trails for each hyperparameter in each partition, resulting in three accuracy measurements, training accuracy, validation accuracy, and testing accuracy. Each partition executed a gridsearch consisting of three cross-validation trials to find the best hyperparameters for that classifier and partition. Once the parameters were found, the training data was fitted to each classifier. This process was repeated for each dataset. The results are shown below. Each row represents a separate experiment based on the specified classifier and partition. The best hyperparameter for each partitioned training set is also listed.

Heart Disease:



	Classifier	Partition	Best Parameter	Train Acc	Val Acc	Test Acc
0	Logistic Regression	20.0/80.0	{'C': 0.01, 'penalty': 'l2', 'solver': 'liblinear'}	0.692308	0.750000	0.630252
1	Logistic Regression	50.0/50.0	{'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}	0.897959	0.880000	0.798658
2	Logistic Regression	80.0/20.0	{'C': 0.01, 'penalty': 'l2', 'solver': 'sag'}	0.835443	0.721519	0.800000
3	Decision Tree	20.0/80.0	{'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2}	1.000000	0.550000	0.789916
4	Decision Tree	50.0/50.0	{'criterion': 'gini', 'max_depth': None, 'min_samples_leaf': 4, 'min_samples_split': 10}	0.826531	0.720000	0.704698
5	Decision Tree	80.0/20.0	{'criterion': 'gini', 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 10}	0.917722	0.772152	0.733333
6	Random Forest	20.0/80.0	{'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 50}	0.923077	0.850000	0.785714
7	Random Forest	50.0/50.0	{'criterion': 'entropy', 'max_depth': 15, 'min_samples_leaf': 4, 'min_samples_split': 5, 'n_estimators': 50}	0.887755	0.840000	0.805369
8	Random Forest	80.0/20.0	{'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 50}	0.905063	0.810127	0.833333

Adult:



	Classifier	Partition	Best Parameter	Train Acc	Val Acc	Test Acc
0	Logistic Regression	20.0/80.0	{'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}	0.828496	0.827197	0.831223
1	Logistic Regression	50.0/50.0	{'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}	0.832825	0.832957	0.833355
2	Logistic Regression	80.0/20.0	{'C': 0.01, 'penalty': 'l2', 'solver': 'newton-cg'}	0.835019	0.833320	0.829630
3	Decision Tree	20.0/80.0	{'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 5}	0.849560	0.829519	0.836475
4	Decision Tree	50.0/50.0	{'criterion': 'gini', 'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2}	0.862545	0.847685	0.838574
5	Decision Tree	80.0/20.0	{'criterion': 'entropy', 'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2}	0.851231	0.838378	0.854505
6	Random Forest	20.0/80.0	{'criterion': 'entropy', 'max_depth': None, 'min_samples_leaf': 2, 'min_samples_split': 10, 'n_estimators': 100}	0.884558	0.839801	0.836586
7	Random Forest	50.0/50.0	{'criterion': 'gini', 'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 100}	0.890938	0.839724	0.833798
8	Random Forest	80.0/20.0	{'criterion': 'entropy', 'max_depth': None, 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 100}	0.883655	0.841197	0.841791

Credit Approval:

	Classifier	Partition	Best Parameter	Train Acc	Val Acc	Test Acc
0	Logistic Regression	20.0/80.0	{'C': 1, 'penalty': 'l1', 'solver': 'liblinear'}	0.883721	0.863636	0.858509
1	Logistic Regression	50.0/50.0	{'C': 1, 'penalty': 'l1', 'solver': 'liblinear'}	0.852535	0.853211	0.883792
2	Logistic Regression	80.0/20.0	{'C': 10, 'penalty': 'l1', 'solver': 'liblinear'}	0.859195	0.862069	0.862595
3	Decision Tree	20.0/80.0	{'criterion': 'entropy', 'max_depth': None, 'min_samples_leaf': 4, 'min_samples_split': 2}	0.941860	0.886364	0.810707
4	Decision Tree	50.0/50.0	{'criterion': 'gini', 'max_depth': 5, 'min_samples_leaf': 2, 'min_samples_split': 5}	0.889401	0.853211	0.874618
5	Decision Tree	80.0/20.0	{'criterion': 'entropy', 'max_depth': 5, 'min_samples_leaf': 1, 'min_samples_split': 2}	0.885057	0.867816	0.870229
6	Random Forest	20.0/80.0	{'criterion': 'gini', 'max_depth': 15, 'min_samples_leaf': 4, 'min_samples_split': 2, 'n_estimators': 50}	0.953488	0.818182	0.852772
7	Random Forest	50.0/50.0	{'criterion': 'gini', 'max_depth': 15, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}	1.000000	0.844037	0.886850
8	Random Forest	80.0/20.0	{'criterion': 'gini', 'max_depth': 5, 'min_samples_leaf': 4, 'min_samples_split': 5, 'n_estimators': 50}	0.876437	0.902299	0.931298

Conclusion

In the heart disease dataset, the weakest classifier was logistic regression. It had significantly lower values of accuracy across all set types for the 20/80 partition compared to the other classifiers. While the accuracy did improve in the other partitions, the other classifiers performed better overall, with random forest being the most consistent across set types. Despite the decision tree classifier getting higher training accuracy scores, the testing and validation accuracy is significantly lower than the accuracy exhibited by the random forest classifier. As such, the random forest classifier appears to be the best classifier for the heart disease dataset.

The adult dataset is notably a much larger dataset than the other two datasets. Additionally, the data was severely imbalanced between 0 and 1. This may have contributed to the classifiers' performances. Here, the accuracy scores appear to be very similar across partitions, classifiers, and set types. Perhaps due to the size of the dataset, there is less variation in the classification task. All of the testing accuracies revolve around 0.83 and the validation accuracies seem to follow the same trend as well. The main difference in performance can be seen in the training accuracy, where logistic regression has the lowest scoring, decision tree closely follows, and random forests exhibit the highest accuracy. Yet, these differences are minimal. Considering that the decision tree performs about the same as random forests for such a large dataset, this classifier should be prioritized to lessen the time needed to train the model.

Lastly, the credit approval dataset, a relatively small dataset, reveals similar trends to the other two datasets. The overall performance of the classifiers appears to be the highest in this dataset. Most importantly, the testing and training accuracies are the highest in the random forest classifier trials and lowest in the logistic regression trials.

For all of the datasets, as the training size increases, the testing accuracy also increases. Yet, with a smaller testing set, the testing accuracy might fluctuate because the evaluation may be more sensitive to specific samples in the test set. This fluctuation is seen the most in the smallest dataset, and the least in the largest dataset. Accordingly, getting as much data as possible before analysis is crucial.

Conclusively, logistic regression appears to be the worst classifier, while random forests is the strongest classifier, with decision trees producing similar results. These findings closely resemble the outcomes from Caruana and Niculescu-Mizil's paper, where they discovered that random forests outperformed both decision tree classifiers and logistic regression classifiers.

References

Caruana, R., & Niculescu-Mizil, A. (2006). An empirical comparison of supervised learning algorithms. *Proceedings of the 23rd International Conference on Machine Learning - ICML '06*. https://doi.org/10.1145/1143844.1143865

Welcome to the UC Irvine Machine Learning Repository. UCI Machine Learning Repository. (n.d.). https://archive.ics.uci.edu/