Journal of Machine Learning Research 1 (2022) 1-5 Submitted 4/00; Published 10/00

Programming Assignment 1

Peter Rasmussenă PRASMUS3@JHU.EDU
Whiting School of Engineering
Johns Hopkins University
Baltimore MD, USA

Editor: Peter Rasmussen

Abstract

This paper discusses the design and execution of a simple majority predictor along with an analysis of the predictor's performance for regression and classification data sets. We hypothesized that the simple majority predictor is a poor estimator for the six data sets used to train the model. Each data set used in training was pre-processed. Pre-processing steps included replacing non-numeric data with integers, handling categorical data, handing numeric data, discretization of a select number of numeric features, and standardization. We employed cross-validation as a best practice with five folds. We scored model performance on the basis of the class of data set: classification data sets were scored using accuracy and regression data sets were scored using mean squared error. Experimental results support the hypothesis that the simple majority predictor tends to not be an effective estimator for classification and regression data sets. We hypothesize this is because the simple majority predictor has too much bias to be an effective estimator.

Keywords: Simple Majority Predictor, Pre-processing, Classification, Regression

1. Introduction

The simple majority predictor provides a straightforward way to estimate the mean and mode of regression and classification data sets, respectively. The simplicity of the algorithm allows its implementer to focus on building a data pipeline and to assess the limitations of the algorithm. The creation of this pipeline lays the foundation for the implementation of more sophisticated algorithms that will utilize the code developed for this assignment. This paper is organized as follows. Section 2 provides the problem statement. Section 3 summarizes data pre-processing steps. Section 4 presents the experimental approach. Section 5 presents the results. Section 6 discusses the algorithm's behavior. Section 7 concludes.

2. Problem Statement

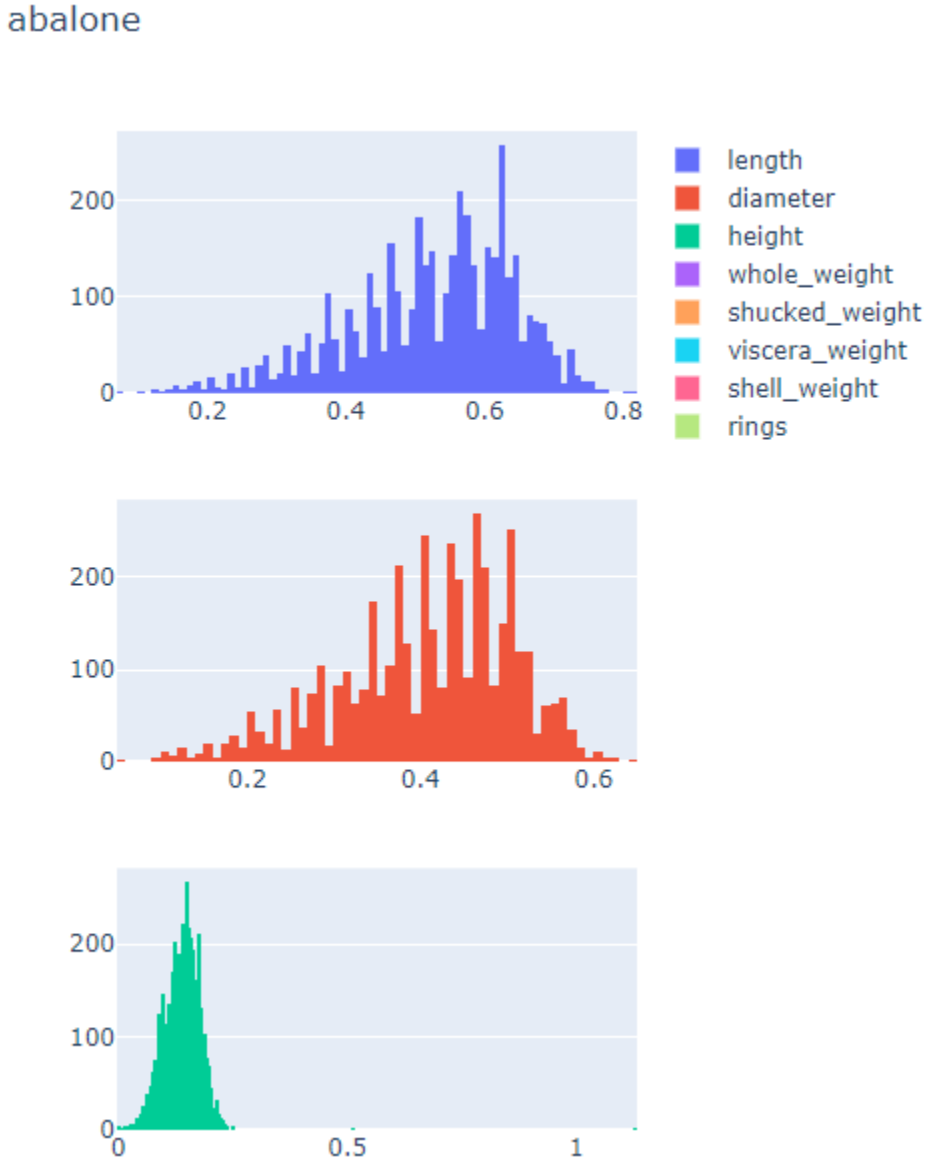
The objectives of this assignment are 1) to teach the student the basics of data processing - through the development of a pipeline - and 2) to assess the performance of a basic learning algorithm, the simple majority predictor. This exercise also familiarized the student with writing this paper and making a video that summarizes the work of the student's assignment.

3. Preprocessing

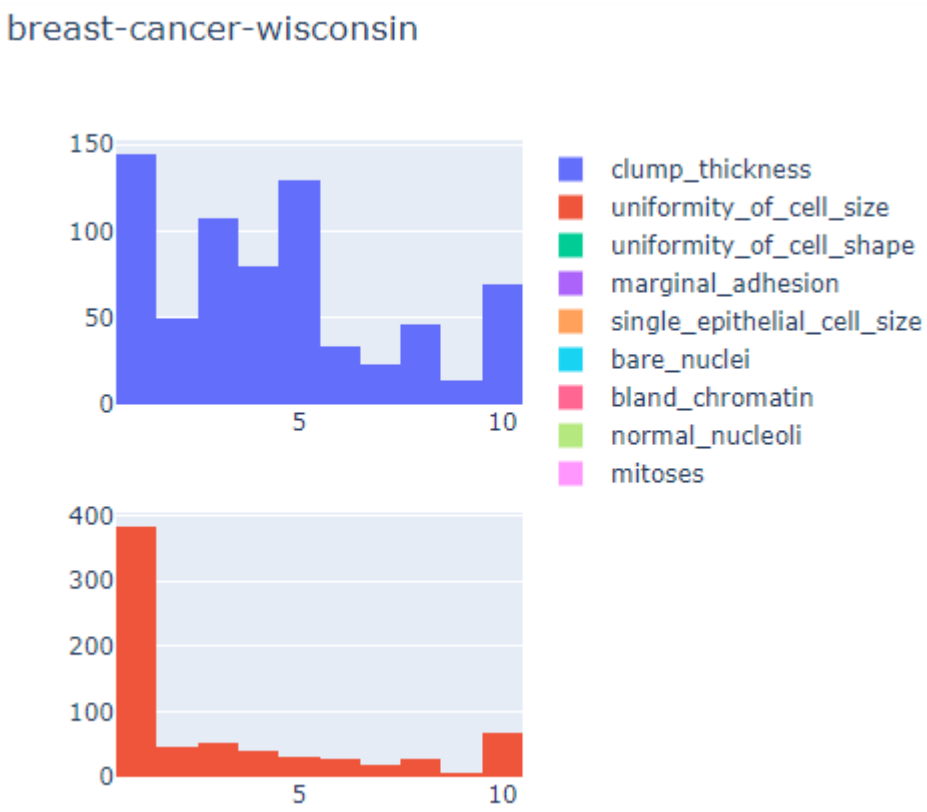
We developed a preprocessing routine that handles both classification and regression data sets that include nominal, ordinal, and numeric features. We subsume the preprocessing into a Preprocessor class so that dataset-specific attributes can be efficiently retrieved to perform each preprocessing step. We make use of a metadata JSON file that provides data-set-specific attributes, such as whether the data is used for classification or regression, data classes (e.g., ordinal, numeric, etc.), missing values, and so forth. A data-set-specific dictionary, created from the JSON parameter file, populates key fields of the preprocessor object.

Upon loading the data, the preprocessor identifies which columns of the data set are features, which is the label, and which - if any - is an index column. Next, strings, including ordinal strings and excluding data to be dummied, are replaced with numeric values. After that, log transformations are performed on selected columns. The following figure illustrates how some features are not totally normally distributed, and may need to be log transformed

to correct for this.



Following the log transformation, categorical columns are dummied, and the original categorical column is removed. Selected features are then discretized according to specifications provided in the metadata file; these default discretization features can be over-ridden. The following figure shows examples in the Breast Cancer Wisconsin dataset of features that may be good candidates for discretization. None of the features are normally distributed. In fact, the distributions appear bi-modal tri-modal with peaks at either end of the ordinal scale.



The data is then shuffled in-place using a pseudo-random seed. The shuffled observations are then assigned to one of K folds (the default in this assignment is five folds). Standardization actually occurs while looping over each fold, but should be considered a preprocessing step. Standardization parameters mean and standard deviation are computed on the training data. Finally, data is split into test and train-validation sets, and the latter is further split into separate train and validation sets.

4. Experimental Approach

We used K folds cross-validation to mitigate over-fitting of the data. Although over-fitting is not an issue with a simple majority predictor, given its bias and under-fitting, cross-validation will serve the student well when more powerful algorithms, which may over-fit, are trained on the data. To carry out the K folds cross-validation, we first split the data into five folds. Care was taken to stratify fold assignments for the classification data sets, as these data were not balanced. Therefore, we achieved a proportionate number of each class in each fold, and furthermore employed stratification to obtain a proportionate number of each class when the train-validation set was split into separate train and validation sets. Stratification is not necessary for regression data sets, and therefore was not employed for those.

After randomly assigning each observation to a fold, we looped over each of the five folds. For each fold, we first split the data into separate test, training, and validation sets. Fold i was assigned as the test and all other folds were assigned as the training-validation set. The training data was then used to train the simple majority predictor.

For this assignment, we assumed that the classes in the classification data sets were balanced. Doing so allowed us to use accuracy as a measure of model performance. We realize that the classes are in fact imbalanced across each data set, and in future assignments will opt for more powerful evaluation metrics, including taking the harmonic mean of precision and recall, an approach that yields the F1 score. We also assumed mean squared error to be a suitable metric to evaluate performance for the regressions data sets, although other metrics, including root mean squared error, may be employed in the future.

5. Results

As expected, the simple majority predictor did not perform well. The following table summarizes scores by problem class and dataset. The classification datasets, by virtue of being scored on the basis of accuracy and not F1 score, were able to achieve scores greater than 0.6 by simply predicting the mode of the label. The regression datasets' mean squared error, which was not normalized, show poor performance as well. In the future, we will normalize mean squared error to enable a more apples-to-apples comparison across datasets.

		test_score
problem_class	dataset_name	
classification	breast-cancer-wisconsin	0.66
	car	0.70
	house-votes-84	0.61
regression	abalone	2.05
	forestfires	0.15
	machine	2.21

6. Algorithm Behavior

The simple majority predictor is a biased estimator and would score lower if precision and recall were computed. This is because accuracy is not a suitable metric for imbalanced classes, as is the case in for each of the classification datasets. The table below illustrates this point.

dataset_name	frac_positive
breast-cancer-wisconsin	0.34
car	0.70
house-votes-84	0.39

7. Conclusion

The objective of this assignment was to familiarize the student with data processing and to evaluate the performance of a simple estimator, namely the simple majority predictor. Our hypothesis that the simple majority predictor is a biased estimator that will generally exhibit poor performance is supported by the results.

In future assignments, we will be sure to use more appropriate scoring metrics, including precision, recall, and F1 score, which can better handle class imbalances that occur in classification datasets. In addition, we will normalize the mean squared error to allow a more apples-to-apples comparison among regression datasets.