report

January 20, 2017

1 Modeling the Madelon Data Set

1.0.1 Domain

The goal in this project was to create a data analysis pipeline. The pipeline will have consistent elemental steps used to read data from a remote SQL data base, initial benchmarking, feature selection, model selection, and validation.

1.0.2 Data

In this project I worked with the Madelon data set, a synthetic data set with many variables and a high degree of non-linearity. It contains 500 features and a binary classification label (-1,1), which I rescale to (0,1). There are a total of 2000 entries, divided evenly between the two labels. According to the source website (https://archive.ics.uci.edu/ml/datasets/Madelon), the data set has a high degree of non-linearity.

1.0.3 Problem Statement

My goal in this project was to produce a model which accurately predicts the labels in the Madelon data set. As the data is highly non-linear, this required significant feature selection and model selection. I implemented three separate analysis pipelines, corresponding to an initial benchmark using logistic regression, feature selection using a logistic regression with lasso regularization, and the final model selection.

1.0.4 Solution Statement

In constructing the pipeline, I constructed four key wrapper functions.

load_data_from_database Accesses the database and saves the data from the 'dsi' table in 'data'.

make_data_dict Generates features and labels, then splits into training and validation sets. The default split is 70% training/30% validations, and a random seed was used throughout to ensure the same split in the notebook used for each step.

general_transformer Performs an arbitrary transformation on the training set, then applies that transformation to the validation set.

general_model Fits and score an arbitrary model, with any model inputs defined in the model before it is passed to the function.

The first two functions have the same inputs/outputs in all three steps, but the last two function have different inputs/outputs in each step:

Step 1: Benchmarking general_transformer is used for normalization, general_model takes in an unregularized logistic regression

Step 2: Feature Selection general_transformer is used for normalization, general_model takes in a series of logistic regressions with different Lasso regularization weights

Step 3: Model Selection general_transformer is used for normalization and selecting the 'k' best features, general_model takes in a set of grid search objects corresponding to l2-regularized logistic regressions, k-nearest neighbors classifiers, and SVC classifiers

Ultimately, the benchmark in step 1 will be used as a baseline for the feature selection in step 2. The best results for the reduced number of features from step 2 will be used for the SelectKBest reduction in step 3.

1.0.5 Metric

I considered the accuracy as the significant metric for this project. Since the data is equally split between two labels, the baseline accuracy is 50% so no other metric is inherently well-suited in comparison to accuracy. Moreover, since the data set is synthetic, there is no obvious metric which is inherently desireable for field-specific reason. In this scenario, accuracy is the easiest metric to extract meaning from, so I will use it throughout.

1.0.6 Benchmark

The initial benchmark was performed using logistic regression. Logistic regression was also used for feature selection and determining the number of salient features, while the final model allowed for logistic regression, k-nearest neighbors, and SVC classification.

1.0.7 Results : Benchmarking

The initial benchmark in step 1 was produced with an unregularized logistic regression using the average accuracy over a 10-fold cross-validation on the training set. At this point no results were obtained for the validation set. The accuracy obtained at this point was 53.6%, only marginally higher than the baseline accuracy of 50%.

2 Results : Feature Selection

Feature selection was performed using a logistic regression with Lasso regularization. The value of the 'C' parameter (inverse of the regularization weight) was chosen in the set [1,0.5,0.2,0.1,0.05,0.04,0.03,0.02,0.01]. Better results were obtained for smaller weights, with the best accuracies of 61-62% at higher weights.

From this step, accuracies of greater than 60% were obtained for selected models which had 2, 8, and 30 features with non-zero coefficients. I carried these values forward to the next section, using them as inputs to the SelectKBest transformation in step 3.

3 Results - Step 3

Construction a final model was performed using a restricted set of features as determined in the previous section, then performing grid searches for three types of classification algorithms. These algorithms and their inputs were were:

Logistic Regression L2 regularization weight : C = [1,0.5,0.2,0.1,0.05,0.04,0.03,0.02,0.01]

k-Nearest Neighbors Number of neighbors : n_neighbors = [1,3,5,11,21,51], Relative point weighting : weights = ['uniform','distance'], and Metric minkowski/euclidian : p = [1,2]

SVC Penalty parameter : C = [1e3,3e2,1e2,30,10,3,1,3e-1,1e-1,3e-2,1e-2,3e-3,1e-3] and Kernel : kernel = ['rbf','sigmoid']

Of these, the best performance was obtained for a k-nearest neighbors model considering the 8 best features, using the 5 closest neighbors weighted according to the euclidian distance between points. The ultimate accuracy was 82% on the cross-validated training set, and 79.5% on unblinded validation set. The other metrics showed similarly strong scores.

4 Results - Comparison of Models

Of the three models used, logistic regression has the worst performance. While reducing the number of features improved the fit somewhat, further use of Ridge regularization resulted in no further improvement in accuracy. This is unsurprising, as the Madelon data set is highly non-linear and logistic regression is a linear model. Both k-nearest neighbors and SVC had similar performance to logistic regression on the 2-feature model, which suggests that not enough information is contained in those two features for an accurate prediction. For 8 and 30 features, in comparison, both k-nearest neighbors and SVC had superior performance to linear regression. In both cases the performance was best for 8 features, with accuracies of 82% and 75% for k-nearest neighbors and SVC respectively.

5 Conclusion

In this project I considered classification algorithms on the Madelon data set. I set up a modular pipeline used for benchmarking, feature selection, and model selection. From a benchmark accuracy of 53% on a data set with baseline accuracy of 50%, I identified a model using k-nearest neighbors classification which has an 82% accuracy on the cross-validated training set and 79% accuracy on the validation set. I also discussed the relative performance for logistic regression, SVC, and k-nearest neighbors classifiers.

There are several ways this process might be improved to find a more accurate model, most of which are beyond the computationa power I have available. There is a significant change in accuracy for the three values of number of features used, so a more granular scan could find a

better number of features. Similarly, expanding the list of internal model parameters to search over and improving the granularity of the scan. More classifiers could also be considered, such as decision trees or perceptrons. For logistic regression, polynomial features might improve the fits, though that depends greatly on how the non-linearity in the synthetic data set was generated.