CS434 Final Project Report (3-page limit)

Aaron Leondar, Alex Ruef, Ross Shoger

6/13/2018

1 Feature formulation and preprocessing

1.1 Features

To get our features we put together the 30 minute data intervals and then flatten each instance into a vector.

1.2 Preprocessing

The UTC time was preprocessed to simply be an hour value between 1 and 24

2 Learning algorithms

2.1 Algorithms explored

2.1.1 Neural Network

Neural networks are all the rage right now and it is a powerful algorithm so we wanted to try it out.

2.1.2 Decision Tree

We liked decision tree when we used it in previous assignments. We thought our better understanding of this algorithm could boost our success rate.

2.1.3 SVC

Support Vector Clustering seemed like a powerful algorithm to use in order to separate the data into definite groups.

2.1.4 Naive Bayes

Naive Bayes is highly scalable, and therefore has a good probability to handle a large amount of data sets, which would could work well for this project.

2.1.5 Logistic Regression

We chose Logistic Regression due to it being especially good (in fact being the go-to method) for binary classification problems, which is what this project is, since our prediction value is a binary number. 1 = Hypo event, 0 = non-Hypo event.

2.2 Final models

1. Neural Network
2. Decision Tree
3. Logistic Regression

3 Parameter Tuning and Model Selection

3.1 Parameter Tuning

3.1.1 Decision Tree

Originally we were using the Gini criterion for decision tree branching which got us good results on the training set but we were getting no positive predictions on the test set. Changing the criterion to use entropy gave us a slightly higher prediction rate on the training set and gave us some positive predictions on the final test set.

3.1.2 SVC

Tried out different kernels, mostly tested with sigmoid and different polynomials. Any polynomial with degree larger than 3 took too long to finish and we got mostly false positives with polynomial.

3.1.3 Neural Network

We tried adjusting many of the parameters such as the number of hidden layers and max iterations. We found the default setting was best for many of the parameters. We settled on using a relu activation function with an Ibgfs solver for weight distribution. The hard part with Neural Network is we often get no positive predictions and other times it is normal.

3.1.4 Naive Bayes

For Naive Bayes we used a GaussianNB algorithm which didn't have much parameters to change besides priors which didn't apply in our situation. The success rate was so low early on we didn't put much effort into this one.

3.1.5 Logistic Regression

We tried different solver methods like sag and saga but they gave us lower precision rates so we stuck with the default liblinear solver. We also tried lower and increasing the tolerance for the stopping criteria and saw no changes to precision and recall. Reducing the strength of the regularization got us a small boost in precision.

3.2 Model selection

We used hold out validation to test our algorithms. We were really interested in how many true positives our algorithms were getting. Since there are so little positive values in the data set we made the test data set larger than the training set so the algorithms had a chance to find more true positives. With holdout validation we looked for algorithms that had high precision and recall rates to use as our top 3 algortihms.

4 Results

Do you have any internal evaluation results you want to report?

Algorithm	Precision	Recall	Overall Success Rate
Naive Bayes	5%	1%	63%
Neural Network	39%	3%	97%
Decision Tree	25%	2%	96%
SVC	3%	0%	3%
Logistic Regression	21%	1%	93%