Online Applied Machine Learning

Final Project

Arjun S Rao (arsrao@iu.edu)

2000008007

04/29/2017

Evaluation of 7 Machine learning Classification Algorithms on Adult data set and Wisconsin breast-cancer data-set UCI repository

Introduction:

In this project, I evaluate performance of 7 Machine learning Algorithms on Adult data set that has census information from 1994 and on Wisconsin breast cancer data set.

DATA PRE-PROCSSING:

Adult Census data set - There are 14 attributes prescribed to each person:

{income (‘>50K’ or ‘<=50K’), age, workclass, fnlwgt, education, education-num, marital-status, occupation, relationship, race, sex, capital-gain, capital-loss, hours-per-week, native-country}.

For every instance that has a missing value I drop the entire row, also I discterzise the feature set, I keep it this way with all the models, for consistency and to identify which performs the best for one dataset.

The feature I discretize,

```
'workclass', 'marital-status', 'relationship', 'race', 'education-num', 'sex', 'Salary'
category_col = data[['workclass', 'marital-status', 'relationship', 'race', 'education-num', 'sex', 'Salary' ]]
for col in category_col:
b, c = np.unique(data[col],
return_inverse=True)
data[col] = c
```

Using the above line to discretize my features,

The value associated with the columns after discretizing the features,
'workclass': [' Federal-gov', ' Local-gov', ' Private', ' Self-emp-inc', ' Self-emp-not-inc', ' State-gov', ' Without-pay'] [**0, 1, 2, 3, 4, 5, 6]** respectively
'marital-status': [' Divorced' ' Married-AF-spouse' ' Married-civ-spouse' ' Married-spouse-absent' ' Never-married' ' Separated' ' Widowed'] : [**0, 1, 2, 3, 4, 5, 6]** respectively
'relationship': [' Husband' ' Not-in-family' ' Other-relative' ' Own-child' ' Unmarried' ' Wife’] : [**0, 1, 2, 3, 4, 5]** respectively
'race': [' Amer-Indian-Eskimo' ' Asian-Pac-Islander' ' Black' ' Other' ' White'] ; [**0, 1, 2, 3, 4]**
'sex': [' Female' ' Male']; [**0, 1]**
'Salary': [' <=50K' ' >50K']; **[0, 1]** respectively

Wisconsin Breast Cancer Data – There are 11 attributes to each patient:

{Sample code number, Clump Thickness, Uniformity of Cell Size, Uniformity of Cell Shape, Marginal Adhesion, Single Epithelial Cell Size, Bare Nuclei, Bland Chromatin Normal Nucleoli, Mitoses, Class (2 for benign, 4 for malignant) }

Every value in the data set is numerical and is discrete no changes done to the data-set.

Analysis:

Evaluation: Since we are dealing with a binary classification task, to evaluate the performance of the models, I will use accuracy which is (1 – error rate), where

error rate = # of mistakes/ size of data set

Algorithm	Train time (sec)	Test time (sec)	Train Accuracy	Test Accuracy
Logistic Regression	2.78	0.002	0.79	0.80
Naïve Bayes	0.29	0.008	0.76	0.77
KNN	1.02	0.143	0.83	0.83
Decision Trees	1.47	0.006	0.84	0.86
Random Forest	3.02	0.068	0.89	0.90
SVM	14.0	0.017	0.74	0.74
Ada Boost	1.81	0.038	0.80	0.81

Table 1.1 for Adult census data set

Algorithm	Train time (sec)	Test time (sec)	Train Accuracy	Test Accuracy
Logistic Regression	0.042	0.002	0.95	0.95
Naïve Bayes	0.0419	0.0009	0.9	0.96
KNN	0.049	0.004	0.95	0.96
Decision Trees	0.028	0.001	0.95	0.92
Random Forest	0.162	0.003	0.95	0.95
SVM	0.040	0.001	0.95	0.96
Ada Boost	0.17	0.002	0.95	0.94

Table 1.2 Wisconsin breast cancer data set

As seen from **table 1.1, Random forest** has the best accuracy and almost the same training and testing time as Ada boost for 10 trees and minimum split of 10 with gini impurity, this is because RF is an ensemble method where we have 10 decision trees and we are averaging the prediction of 10 decision trees. Because of how decision tree work (select an attribute and the classify based on values) and the way error is calculated this does a better job of classifying accurately as compared to all other algorithms.

Decision tree, we know that decision trees selects attribute that has highest information gain places it on top as a root node, then it keeps splitting recursively until we stop it at an optimal stage, because all the dataset values here are categorical discrete variable, decision trees works pretty accurately and fast, it fits fast and runs fast, also because Random forest is an ensemble method, RF takes average of all the decision tree and then outputs the accuracy, we used 10 decision trees here, it is clear from the table that it takes a little longer than decision tree to fit data but has the highest accuracy.

Model with third best accuracy is **KNN**, the parameters affecting the performance of K-NN are number of nearest neighbors (k), distance function, and weighting function, here our K = 3, given the data-set had all discrete values, we observe that KNN wasn't computationally intensive either and pretty accurate.

Ada boost: Ada boost also adaptive boosting combines set of weak learners is combined into a weighted sum that represents the final output of the boosted classifier. AdaBoost is adaptive in the sense that subsequent weak learners are tweaked in favor of those instances misclassified by previous classifiers,

Here, we use 10 decision trees as weak learner with learning rate of 1.0, because it combines decision trees, the accuracy is not bad at all, changing parameter like number of trees and learning rate would change the performance.

Logistic regression: we use LR with L1 penalty, this performs as well as all the other better performing classifiers, the reason I think this is not performing as well on this data-set could be logistic regression assumes linearity of independent variables and log odds

Naïve Bayes: Naïve Bayes has one of the worst accuracy here, one reason I could think of is the assumption of independence, real life data-set are not – most of the time independent,

SVM: the table shows that training time for SVM is higher than other algorithms, this is because training time for SVMs is quadratic, also 2-class linear SVM requires about "nd" computation for training (times the number of training iterations, which remains small even for large n) and on the order of d computations for classification. So, when the number of training examples is large SVMs are too expensive for training and very expensive for classification, also because we run for just 60000 iteration the accuracy is not as high as we would desire, because we have 30000 samples, SVM proved to be a slow, computationally expensive algorithm for this dataset.