Building a Student Intervention System

By : Shmuel Naaman

Supervised Learning

Building a Student Intervention System

1. Classification vs Regression

The goal is to identify students who might need early intervention - In this case we are solving a “classification problem”. Each student in the data set will be categorize according to the parameters as "need early intervention" or does not "need early intervention". The algorithm will fit each student into one of these categories.

2. Exploring the Data

Let's go ahead and read in the student dataset first.

```
Some statistics facts about the dataset
 - Total number of students: 395
 - Number of students who passed: 265
 - Number of students who failed: 130
 - Number of features: 30
 - Graduation rate of the class: 67.09%
```

3. Preparing the Data

In this section, we will prepare the data for modeling, training and testing.

Identify feature and target columns

Separate the data into feature and target columns, and see if any features are non-numeric.
Note: For this dataset, the last column (`'passed'`) is the target or label we are trying to predict.

In [4]:

```
Feature column(s):-
['school', 'sex', 'age', 'address', 'famsize', 'Pstatus', 'Medu', 'Fedu',
'Mjob', 'Fjob', 'reason', 'guardian', 'traveltime', 'studytime', 'failures',
'schoolsup', 'famsup', 'paid', 'activities', 'nursery', 'higher', 'internet',
'romantic', 'famrel', 'freetime', 'goout', 'Dalc', 'Walc', 'health',
'absences']
Target column: passed
```

Preprocess feature columns

As we can see, there are several non-numeric columns that need to be converted! Many of them are simply `yes`/`no`, e.g. `internet`. These can be reasonably converted into `1`/`0` (binary) values.

Other columns, like `Mjob` and `Fjob`, have more than two values, and are known as *categorical variables*. The recommended way to handle such a column is to create as many columns as possible values (e.g. `Fjob_teacher`, `Fjob_other`, `Fjob_services`, etc.), and assign a `1` to one of them and `0` to all others.

These generated columns are sometimes called *dummy variables*, and we will use the `pandas.get_dummies()` function to perform this transformation.

In [5]:

```
Processed feature columns (48):-
['school_GP', 'school_MS', 'sex_F', 'sex_M', 'age', 'address_R', 'address_U',
'famsize_GT3', 'famsize_LE3', 'Pstatus_A', 'Pstatus_T', 'Medu', 'Fedu',
'Mjob_at_home', 'Mjob_health', 'Mjob_other', 'Mjob_services', 'Mjob_teacher',
'Fjob_at_home', 'Fjob_health', 'Fjob_other', 'Fjob_services', 'Fjob_teacher',
'reason_course', 'reason_home', 'reason_other', 'reason_reputation',
'guardian_father', 'guardian_mother', 'guardian_other', 'traveltime',
'studytime', 'failures', 'schoolsup', 'famsup', 'paid', 'activities',
'nursery', 'higher', 'internet', 'romantic', 'famrel', 'freetime', 'goout',
'Dalc', 'Walc', 'health', 'absences']
```

Split data into training and test sets

So far, we have converted all *categorical* features into numeric values. Next, we split the data (both features and corresponding labels) into training and test sets. In addition I set the random state of the train_test_split function so the results will be reproducible.

In [6]:

```
Training set: 300 samples
Test set: 95 samples
```

4. Training and Evaluating Models

Next I choose three supervised learning models that are available in scikit-learn, and appropriate for this problem.

For each model :

- Discussion about the general applications of the model. The strength and weaknesses of the model.
- Reasons for choosing this model.
- Fit the model to the training data, predict labels (for both training and test sets), and measure the F_1 score with different training set sizes (100, 200, 300), keeping test set constant.

Produce a table showing training time, prediction time, F_1 score on training set and F_1 score on test set, for each training set size.

Support Vector Machines Classification O(n3):

General applications:

Given, data points that belong into two clusters, each data point can be represented by a n-dimensional vector. It is possible to find many n-1 dimensional hyper planes that will separate the data points into two clusters. Support Vector Machines Classification will find the hyper plane with maximum separation or margins.

general applications

SVM is helpful in text categorization, images classifications, highly efficient in protein classifications and classification of hand writers' characters.

The advantages of support vector machines are:

- Effective in high dimensional spaces.
- Still effective in cases where number of dimensions is greater than the number of samples.
- Uses a subset of training points in the decision function (called support vectors), so it is also memory efficient.
- Versatile: different Kernel functions can be specified for the decision function. Common kernels are provided, but it is also
- possible to specify custom kernels.

The disadvantages of support vector machines include:

- If the number of features is much greater than the number of samples, the method is likely to give poor performances.
- SVMs do not directly provide probability estimates, these are calculated using an expensive five-fold cross-validation (see Scores and probabilities, below).
- Several specialized algorithms enable quick solution of the quadratic programming problem. Kernel SVM is available in many toolkits. SVM is able to separate feature spaces that are not linearly separated. SVM is effective for problem with high dimensional space. SVM is effective for cases where number of dimensions is greater than the number of samples. SVM is Memory efficient. [http://scikit-learn.org/stable/modules/svm.html]
- Parameters of a solved model are difficult to interpret [https://en.wikipedia.org/wiki/Support_vector_machine]. The classification is into two class and the solution is un-celebrated (we cannot estimate the degree of certainty of a given solutions). All input data must be labeled (no unsupervised learning). If the number of features is much greater than the number of samples, the method is likely to give poor performances

Reasons for choosing this model.

The structure of the data set is a classic case for SVM, each label can be describe with a vector of values. The problem that we want to solve is classification. The fact that the data set is small will make sure that the training will be done in a short time. The data set might not be high dimensional but consider the small size of the data set, seems as SVM will have an advantage with the data set.

In [7]:

AdaBoost Adaptive Boosting Classification

General applications:

A Meta-algorithm that can be used in conjunction with other types of learning algorithms and improve their performance. The output of the other learning algorithms ('weak learners') is combined as a weighted sum to create the boosted classifier. AdaBoost used as a Standard algorithm for Face and object Detection,

Strengths:

AdaBoost is less susceptible to over fitting than other learning algorithms. AdaBoost is the best out-of-the-box classifier. AdaBoost is simple to implement. AdaBoost is can perform feature selection

Weaknesses:

AdaBoost is sensitive to uniform noisy data and outliers. AdaBoost depends on data and weak learner and can fail if weak classifiers are overfit or underfit.

Reasons for choosing this model.

As part of the exploration I wanted to test a predictive model that will be less susceptible to overfitting and since Adaboost is consider the best out-of-the-box classifier, I wanted to check if a boosting model will do better.

In [11]:

Extremely Randomized Trees

General applications:

Ensemble learning method for classification, that construct a multitude decision trees at training, outputting the class of the individual trees. Random decision forests correct for decision trees' habit of overfitting to their training set. The algorithm, inducing random forest "bagging" idea and the random selection of features, in order to construct a collection of decision trees with controlled variance. Random forests use tree learning algorithm that selects, at each candidate split in the learning process, a random subset of the features. Extremely randomized trees or ExtraTrees are trained like in an ordinary random forest, but additionally the top-down splitting in the tree learner is randomized. Random forests can be used to rank the importance of variables in a regression or classification problem in a natural way. Hydrological problem. Computational intensive problems. Modelling of large datasets and input selection.

Strengths:

More complex classifier (a larger forest) getting more accurate nearly monotonically. Combine the predictions of several base estimators built with a given learning algorithm in order to improve generalizability / robustness over a single estimator. effective option to balance accuracy and computational efficiency in data-driven modelling.[http://www.hydrol-earth-syst-sci.net/17/2669/2013/hess-17-2669-2013.pdf]

Weaknesses:

The larger the number of trees in the forest the better, but also the longer it will take to compute. The results will stop getting significantly better beyond a critical number of trees.

Reasons for choosing this model.

The fact that we deal with students and as we know humans are not easy to predict, a model that will include randomization in the process of creating classifiers might captured something that could not be captured by the well formulated models.

SVC

Dataset	Training Time	Prediction Time	Training F1 Score	Test F1 Score
Training 300	0.016	0.004	0.876	0.783
Training 200	0.008	0.002	0.867	0.781
Training 100	0.003	0.002	0.877	0.774

AdaBoostClassifier

Dataset	Training Time	Prediction Time	Training F1 Score	Test F1 Score
Training 300	0.200	0.012	0.863	0.781
Training 200	0.197	0.011	0.892	0.828
Training 100	0.202	0.016	0.948	0.766

ExtraTreesClassifier

Dataset	Training Time	Prediction Time	Training F1 Score	Test F1 Score
Training 300	0.049	0.002	1.0	0.766
Training 200	0.049	0.002	1.0	0.751
Training 100	0.046	0.002	1.0	0.736

5. Choosing the Best Model

- Based on the experiments and the results, a short explaintion about the optimal model of choice. The comparison will consider the available data, limited resources, cost, and performance?

The training time for SVM is 0.01 sec, Adaboost 0.2 and for Extremely Randomized Trees 0.05. Prediction time for SVM is 0.003 sec, Adaboost 0.01 and for Extremely Randomized Trees 0.002. F1 score for the SVM is 0.78, Adaboost 0.79, and for Extremely Randomized Trees 0.75. (Running the model with different random state provides different results, but the difference is not significant).

Considering the above, and the given problem limitations, the most appropriate algorithm is the Adaboost. Adaboost performed best when considering the Prediction time and the performance. The long training time is not significant if we consider the fact that the problem is not real time and the data set is quite small. The school will definitely will prefer more accurate model than faster model.

- A short explenation in layman's terms how the AdaBoost is trained and predict.

AdaBoost uses a number of training samples to pick a number of good 'classifiers'. In this case we consider classifiers as the features that describe each student. Good features (classifiers) will be features that distinguish between students that need intervention to students that do not need interventions. AdaBoost will look at a number of classifiers and find out which is the best predictor of a label (intervention) based on the sample. After it has chosen the best classifier it will continue to find another classifier until some threshold is reached and those classifiers combined together will provide the end result.

Fine-tune the model. Use Gridsearch.

```
----------------------------------------
Training set size: 300
Training AdaBoostClassifier...
Done!
Training time (secs): 0.781
Predicting labels using AdaBoostClassifier...
Done!
Prediction time (secs): 0.050
F1 score for training set: 0.833333333333
Predicting labels using AdaBoostClassifier...
Done!
Prediction time (secs): 0.029
F1 score for test set: 0.802816901408
```

- What is the model's final F_1 score? F1 score for test set: 0.8

Reference

http://wikipedia.org/

http://stackoverflow.com/

http://scikit-learn.org/