Student Intervention System

Supervised Learning Project

Ihab Sultan

Contents

1. Introduction

1.1 Project Overview

The goal of this project is to model the factors that predict how likely a student is to pass their high school final exam.
The school district has a goal to reach a 95% graduation rate by the end of the decade by identifying students who need intervention before they drop out of school.

The model will be evaluated on three factors:

- Its F1 score, summarizing the number of correct positives and correct negatives out of all possible cases. In other words, how well does the model differentiate likely passes from failures?
- The size of the training set, preferring smaller training sets over larger ones. That is, how much data does the model need to make a reasonable prediction?
- The computation resources to make a reliable prediction. How much time and memory is required to correctly identify students that need intervention?

1.2 Software and Libraries

The following SW was used in the first part of the project:

- Python 2.7
- NumPy
- scikit-learn
- iPython Notebook

In the last part of this project, R was used as an EDA tool:

- R 3.2.3
- corrplot
- GGally

2. Student Intervention Data Set

2.1 Available Attributes

1. **school**: student's school (binary: "GP" or "MS")
2. **sex**: student's sex (binary: "F" - female or "M" - male)
3. **age**: student's age (numeric: from 15 to 22)
4. **address**: student's home address type (binary: "U" - urban or "R" - rural)
5. **famsize**: family size (binary: "LE3" - less or equal to 3 or "GT3" - greater than 3)
6. **Pstatus**: parent's cohabitation status (binary: "T" - living together or "A" - apart)
7. **Medu**: mother's education (numeric: 0 - none, 1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education)
8. **Fedu**: father's education (numeric: 0 - none, 1 - primary education (4th grade), 2 – 5th to 9th grade, 3 – secondary education or 4 – higher education)
9. **Mjob**: mother's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
10. **Fjob**: father's job (nominal: "teacher", "health" care related, civil "services" (e.g. administrative or police), "at_home" or "other")
11. **reason**: reason to choose this school (nominal: close to "home", school "reputation", "course" preference or "other")
12. **guardian**: student's guardian (nominal: "mother", "father" or "other")
13. **traveltime**: home to school travel time (numeric: 1 - <15 min., 2 - 15 to 30 min., 3 - 30 min. to 1 hour, or 4 - >1 hour)
14. **studytime**: weekly study time (numeric: 1 - <2 hours, 2 - 2 to 5 hours, 3 - 5 to 10 hours, or 4 - >10 hours)
15. **failures**: number of past class failures (numeric: n if 1<=n<3, else 4)
16. **schoolsup**: extra educational support (binary: yes or no)
17. **famsup**: family educational support (binary: yes or no)
18. **paid**: extra paid classes within the course subject (Math or Portuguese) (binary: yes or no)
19. **activities**: extra-curricular activities (binary: yes or no)
20. **nursery**: attended nursery school (binary: yes or no)
21. **higher**: wants to take higher education (binary: yes or no)
22. **internet**: Internet access at home (binary: yes or no)
23. **romantic**: with a romantic relationship (binary: yes or no)
24. **famrel**: quality of family relationships (numeric: from 1 - very bad to 5 - excellent)
25. **freetime**: free time after school (numeric: from 1 - very low to 5 - very high)
26. **goout**: going out with friends (numeric: from 1 - very low to 5 - very high)

27. **Dalc**: workday alcohol consumption (numeric: from 1 - very low to 5 - very high)
28. **Walc**: weekend alcohol consumption (numeric: from 1 - very low to 5 - very high)
29. **health**: current health status (numeric: from 1 - very bad to 5 - very good)
30. **absences**: number of school absences (numeric: from 0 to 93)
31. **passed**: did the student pass the final exam (binary: yes or no)

The very last attribute (*passed*) will be separated as an output label in sklearn dataset.

3. Project Report

3.1 Classification vs Regression

Your goal is to identify students who might need early intervention - which type of supervised machine learning problem is this, classification or regression? Why?

The purpose is to perform classification, as we are trying to predict whether and intervention is needed or not (ie. binary classification).

3.2 Exploring the Data

Can you find out the following facts about the dataset?

- **Total number of students**
- **Number of students who passed**
- **Number of students who failed**
- **Graduation rate of the class (%)**
- **Number of features (excluding the label/target column)**

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%

3.3 Preparing the Data

- **Identify feature and target columns**

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

Feature values:-
  school sex  age address famsize Pstatus  Medu  Fedu     Mjob      Fjob  \
0     GP   F   18       U     GT3       A     4     4  at_home   teacher
1     GP   F   17       U     GT3       T     1     1  at_home     other
2     GP   F   15       U     LE3       T     1     1  at_home     other
3     GP   F   15       U     GT3       T     4     2   health  services
4     GP   F   16       U     GT3       T     3     3    other     other

    ...    higher internet  romantic  famrel  freetime goout Dalc Walc health  \
0   ...       yes       no        no       4         3     4    1    1      3
1   ...       yes      yes        no       5         3     3    1    1      3
2   ...       yes      yes        no       4         3     2    2    3      3
3   ...       yes      yes       yes       3         2     2    1    1      5
4   ...       yes       no        no       4         3     2    1    2      5

  absences
0        6
1        4
2       10
3        2
4        4

[5 rows x 30 columns]
```

- **Preprocess feature columns**
- **Split data into training and test sets**

3.4 Training and Evaluating Models

Choose 3 supervised learning models that are available in scikit-learn, and appropriate for this problem. For each model:

3.4.1 Support Vector Machines (SVM)

- **What are the general applications of this model? What are its strengths and weaknesses?**

  Strengths

  Support vector machines is a general linear classifier, which handles high dimensions efficiently and is able to cope with overfitting by modifying parameters that control complexity-training error tradeoff. SVM performs well when there is a clear marginal separation.

  On top of that, multiple kernel types can be used within SVM framework, which is equivalent to trying multiple classifiers with a change of a single parameter.

  Weaknesses

  SVM is not able to cope with noisy input, where noise might be driving points beyond the boundary to the wrong class. In addition, SVM does not perform very

  well with very large data sets due to training time being cubic with the size of the

data set.

- **Given what you know about the data so far, why did you choose this model to apply?**

  SVMs would perform well with large number of features (dimensions), and has tuning parameters that would allow us to control model complexity. In addition, we don't need to worry about training speed since our data set is small.

- **Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant.**

- **Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.**

	Training set size		
	100	200	300
Training time (secs)	0.001	0.005	0.008
Prediction time (secs)	0.002	0.003	0.007
F1 score for training set	0.941	.913	0.862
F1 score for test set	0.768	0.821	0.848

3.4.2 Decision Trees

- **What are the general applications of this model? What are its strengths and weaknesses?**

  Strengths

  Model is simple, fast to train and test, and results are intuitive and easy to explain since each feature(dimension) is handled separately by decision trees.

  Weaknesses

  Can easily overfit specially in the case where lots of dimensions are used or when the tree is allowed to grow in complexity beyond the optimal depth.

  Applications

  The model is applied when one needs fast and robust classifier, and therefore can be used as a reference before trying other, more complicated classifiers.

- **Given what you know about the data so far, why did you choose this model to apply?**

Decision trees will be used as a reference for classification performance, we don't expect it to be the best performer.

- **Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant.**

- **Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.**

	Training set size		
	100	200	300
Training time (secs)	0.001	0.002	0.004
Prediction time (secs)	0.001	0.001	0.001
F1 score for training set	1.0	1.0	1.0
F1 score for test set	0.745	0.746	0.703

Notice that F1 score for training is 1.0, indicating no false positives or false negatives from model predictions! yet, the performance on test set is clearly lacking, which is a sign of model overfitting.

3.4.3 AdaBoost

- **What are the general applications of this model? What are its strengths and weaknesses?**

  Strengths

  AdaBoost is a model ensemble which utilizes boosting to combine weak learners. It is a powerful classifier which finds good combined hypothesis. In addition, AdaBoost is good at avoiding overfitting.

  During learning process, AdaBoost picks only features that improve prediction of misclassified training samples, which in turn helps in reducing dimensionality.

  Weaknesses

  AdaBoost is sensitive to noisy data and outliers.

- **Given what you know about the data so far, why did you choose this model to apply?**

  We need to verify the performance of ensemble methods on the data set. AdaBoost is a good method which won't suffer from the overfitting issue of decision trees.

- **Fit this model to the training data, try to predict labels (for both training and test sets), and measure the F1 score. Repeat this process with different training set sizes (100, 200, 300), keeping test set constant.**

- **Produce a table showing training time, prediction time, F1 score on training set and F1 score on test set, for each training set size.**

	Training set size		
	100	200	300
Training time (secs)	0.090	0.087	0.111
Prediction time (secs)	0.005	0.005	0.007
F1 score for training set	0.986	0.891	0.867
F1 score for test set	0.739	0.765	0.768

3.5 Choosing the Best Model

in 2-3 paragraphs explain to the board of supervisors what single model you choose as the best model. Which model has the best test F1 score and time efficiency? Which model is generally the most appropriate based on the available data, limited resources, cost, and performance? Please directly compare and contrast the numerical values recored to make your case.

The chosen model is support vector machine(SVM). Among the 5 tested models, it gave the best prediction time and F1 score combination on the test set.
In addition, the model training time is fast and hence can be quickly retrained if more data arrives.
The model took in the order of couple of milliseconds to train and predict, and the F1 score on test set was above .81 in all three training set sizes.

Decision Trees, on the other hand, had slightly better training and prediction times, and high F1 score on training set, but the model suffered from overfitting when applied to test set.
F1 score for decision trees on test set was more than 0.11 less than the score obtained by SVM.

Finally, AdaBoost had good performance on test set, but performance was still worse than SVM, and the training
prediction times was higher as well, so the model was dropped in favor of SVM.
It can be noted, however, that AdaBoost had improved performance as training set size increases. Had we got a larger training sample set, AdaBoost could have been the winner model.

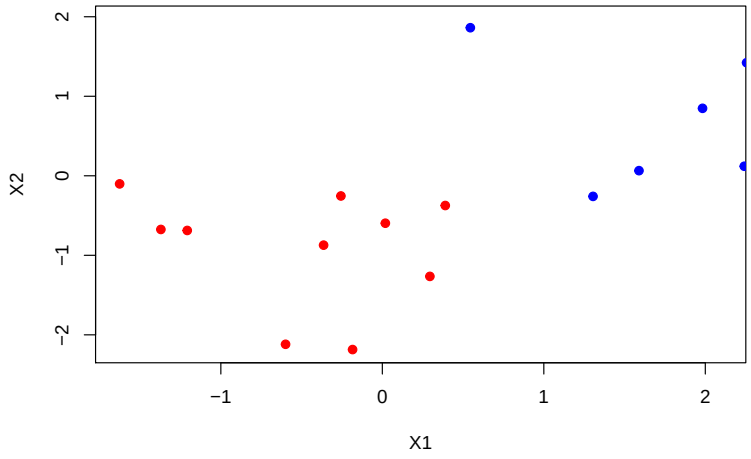
In 1-3 paragraphs explain to the board of supervisors in layman's terms how the final model chosen is supposed to work (for example if you chose a decision tree or support vector machine, how does it learn to make a prediction).

The problem at hand is called a 'classification' problem, meaning that we have some attributes of each student such as his\her weekly study time, health status, number of school absenses, intertent access at home, and many others and want to predict whether the student will pass or fail the final exam.

In such problems, we would search for a model, a black box, which takes all the information of a student and outputs a pass\fail prediction. But in order to build such a model that is responsible for predicting the future, we utilize the current information at hand of all the students who already passed or failed the final exam, we feed all this data to our model, so that it 'learns' how to best separate passing and failing student, this process, as one might expect, is called **'learning'** or **'training'** the model.

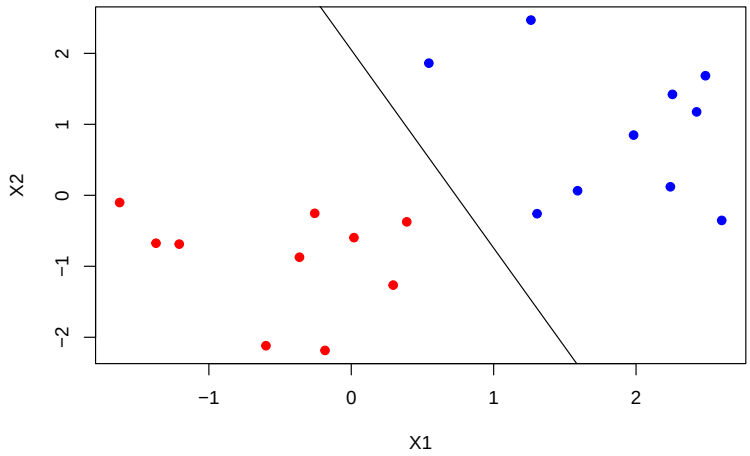
The model we found most capable of doing this prediction is called 'Support vector machine'. This is a powerful model which searches for a line that best separates passing and failing students using their provided information. In order to do that, SVM searches simultaneously for a line which separates passing and failing students as much as possible, while at the same time is able to properly predict whether the student is passing or failing.

To make things more concrete, let's have a look at the following figure:



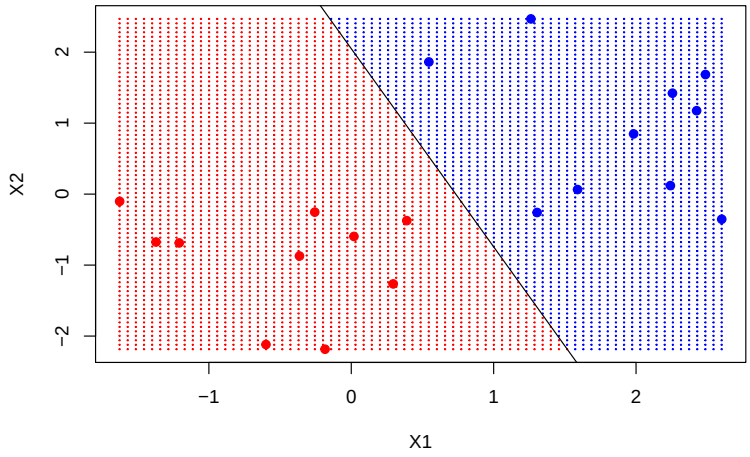
Imagine that blue student are passing final exam, red student are not, and let's assume that the x-axis (labeled x1) represents the study hours, and that y-axis (labeled x2) represents health status during the semester.

SVM looks for a line that separates the two types, let's say something like:



After the model learns how to specify whether a student is passing or failing, it is ready for prime time. A new student is brought forward, all of his\her info is provided to the SVM model, and the model will give us back the expected final results. This process is intuitively called the **'prediction'** process.

In the context of the example above, SVM will try to map each new student to red or blue regions:



Final point for this discussion is: what happens if there is no such straight line that separates passing and failing students?
SVM gives a solution to this in terms of what's called the **'Kernel Trick'**, which basically means that if no straight line is doing a good job, one can try creating a new arrangement of the points, lets say by moving red points outside of the picture plane, and blue points inside, in which case the picture plane itself becomes the line (or more appropriately, the plane) dividing the two types from each other.

Fine-tune the model. Use gridsearch with at least one important parameter tuned and with at least 3 settings. Use the entire training set for this. What is the model's final F1 score?

```
parameters = {'kernel':('linear', 'poly', 'rbf'),
              'C': [1e-5, 1e-2, 1e-1, 1, 10],
              'gamma':['auto'],
              'tol': [1e-1, 1e-2, 1e-3, 1e-4, 1e-5]}
clf = svm.SVC()
#use StratifiedShuffleSplit since class labels are unbalanced
cvs = StratifiedShuffleSplit(y_train,n_iter = 50,random_state = 42)
#optimize F1 score
scorer = make_scorer(f1_metric)
#perform grid search
grid_search = grid_search.GridSearchCV(clf,
                                       parameters,
                                       cv = cvs,
                                       scoring = scorer)
grid_search.fit(X_train,y_train)
```

Final F1 score on optimized model using full-set training was (0.853), which is slightly higher than the value obtained with the pre-optimization model (0.848).

Parameters of the gridsearch optimized model were as follows:

- kernel='linear'
- C=0.01
- gamma='auto'
- tol=0.01

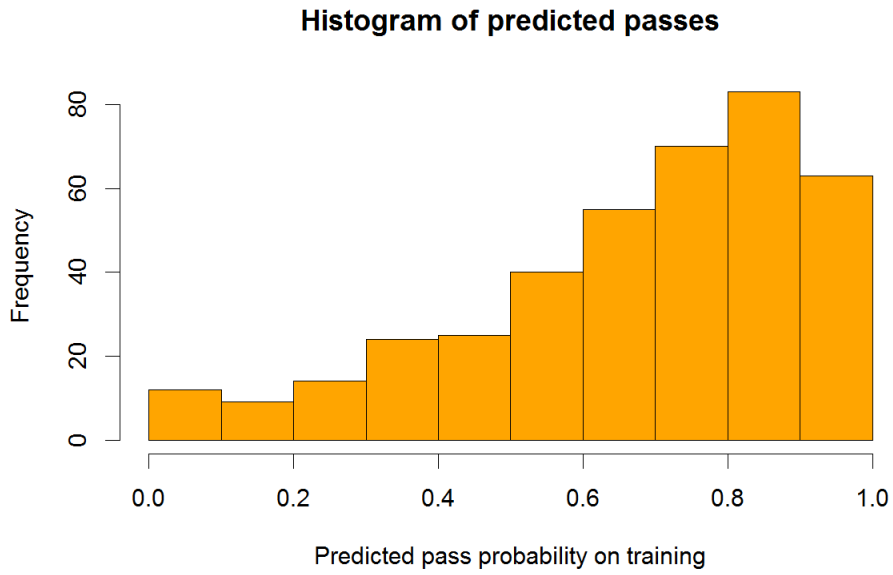
4. Further Explorations with R

This is not part of the project, but an addendum for explorations performed using R statistical package on the students dataset.

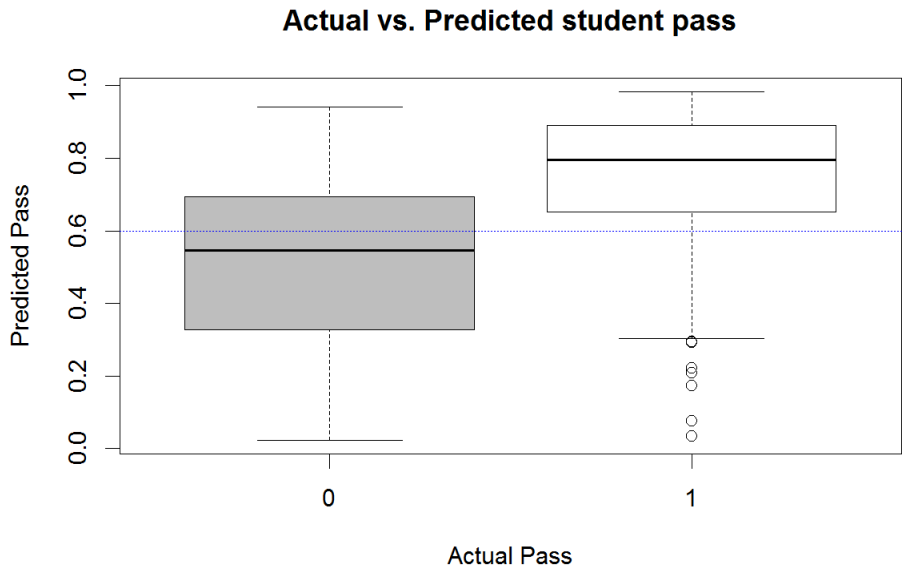
4.1 Logistic Regression

The logistic regression model was used for classification in this section, and performance is compared to our best model from the previous sections obtained using sklearn.

The first plot below shows the result of the logistic regression classification on training set, displayed as a histogram. One of the advantages of logistic regression is that it can be interpreted as a probability distribution over the output classes:



The cutoff parameter used to classify the dependent variable was empirically set to 0.6 to maximize F1 score.
The boxplot below displays the logistic output against the correct classification



With the above setting, the F1 score on *training dataset* was found to be 0.813, which is less than the training error of 0.862 and 0.867 obtained by SVM and AdaBoost, respectively.
The logistic model identified the most significant features of the model below:

$$\begin{aligned}
\textbf{age}, Pr(>|z|) &= 0.048880* \\
\textbf{failures}, Pr(>|z|) &= 1.81e-05*** \\
\textbf{schoolsupyes}, Pr(>|z|) &= 0.017790* \\
\textbf{famsupyes}, Pr(>|z|) &= 0.038396* \\
\textbf{goout}, Pr(>|z|) &= 0.000188**
\end{aligned}$$

So the most important factors according to this model are:

- **Failures**: Number of past class failures.
- **Goout**: Going out with friends.
- **Schoolsupyes**: Extra educational support.
- **Famsupyes**: Family educational support.
- **Age**: Student's age.