Build a Student Intervention System

As education has grown to rely more and more on technology, more and more data is available for examination and prediction. Logs of student activities, grades, interactions with teachers and fellow students, and more are now captured through learning management systems like Canvas and Edmodo and available in real time. This is especially true for online classrooms, which are becoming more and more popular even at the middle and high school levels.

Within all levels of education, there exists a push to help increase the likelihood of student success without watering down the education or engaging in behaviors that raise the likelihood of passing metrics without improving the actual underlying learning. Graduation rates are often the criteria of choice for this, and educators and administrators are after new ways to predict success and failure early enough to stage effective interventions, as well as to identify the effectiveness of different interventions.

Toward that end, your goal as a software engineer hired by the local school district 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. You being a clever engineer decide to implement a student intervention system using concepts you learned from supervised machine learning. Instead of buying expensive servers or implementing new data models from the ground up, you reach out to a 3rd party company who can provide you the necessary software libraries and servers to run your software.

However, with limited resources and budgets, the board of supervisors wants you to find the most effective model with the least amount of computation costs (you pay the company by the memory and CPU time you use on their servers). In order to build the intervention software, you first will need to analyze the dataset on students' performance. Your goal is to choose and develop a model that will predict the likelihood that a given student will pass, thus helping diagnose whether or not an intervention is necessary. Your model must be developed based on a subset of the data that we provide to you, and it will be tested against a subset of the data that is kept hidden from the learning algorithm, in order to test the model's effectiveness on data outside the training set.

Your 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. Classification vs Regression

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

Answer:

This is a classification problem. We are trying to identify students that might need early intervention. Therefore, we train our model to differentiate between two classes of students – students in need of intervention and students not in need of intervention. For each student we will report the more likely class as the “intervention-status”. If we were interesting in predicting a continuous variable like future income or GPA, this would become a regression problem.

2. Exploring the Data

Question: 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)

Use the code block provided in the template to compute these values.

Answer:

- Total number of students: **395**
- Number of students who passed: **265**
- Number of students who failed: **130**
- Graduation rate of the class (%): **67.09%**
- Number of features (excluding the label/target column): **30**

3. Preparing the Data

Question: Execute the following steps to prepare the data for modeling, training and testing:

- Identify feature and target columns

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

Starter code snippets for these steps have been provided in the template.

Answer:

This was done in the notebook.

4. Training and Evaluating Models

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

- What are the general applications of this model? What are its strengths and weaknesses?
- Given what you know about the data so far, why did you choose this model to apply?
- 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.

Note: You need to produce 3 such tables - one for each model.

Answer:

I choose the following 3 models: a decision tree, a random forest with 10 estimators, and a support vector machine. All three models are appropriate for this problem as they are models for classification problems.

Decision Tree:

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

  Decision trees can be used for both classification and regression problems. One of their advantages is that they are easy to visualize and therefore easy to understand by

humans. One other advantage is that they do not require linear relationships between input and output variables which makes them more robust. Other advantages are that decision trees perform variable selection on their own when determining on which variables to split the data set (most predictive variables gets used for the current split) as well as that they do not require any normalization of the features.

Weaknesses of decision trees are that they are not well suited for regression problems. In order to estimate the continuous output, decision trees have to take a vote or calculate an average of the data points in the leafs. Depending on the implementation this can lead to very broad results because too many data points were included in the vote or to overfitting because we build the tree too far and too many data points got included in the vote. Both cases hurt the performance of the model. Fortunately, since we are facing a classification problem with only two classes, this disadvantage does not apply to us. However, one weakness that does apply to our situation is that a decision tree cannot be updated to reflect new training data. Instead, the decision tree has to be build completely new when additional data becomes available. This leads to reoccurring training time and can have a cost impact.

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

I choose the decision tree because we are facing a classification problem with only a few classes and I think that a decision tree's easy visualization would be helpful for better understanding which attributes make students more likely to need intervention. This way, the findings of the model can be easily explained to the board and teaching staff. By understanding the most important variables that indicate a need for intervention, the teaching staff can then further analyze if these variables indicate the true causes for the need of intervention and if that is the case, come up with solution to the underlying issues.

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

Decision Tree	Training Time	Prediction Time	F1 Score on Training Set	F1 Score on Test Set
Predefined Training Set	0.00209	0.000212	1	0.65
30% Training Set	0.000726	0.000155	1	0.62

50% Training Set	0.001102	0.000176	1	0.67
70% Training Set	0.001462	0.000215	1	0.64

Random Forest:

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

Random forests can be seen as extensions of decisions tree as they require the growing of multiple trees. Being an extension of the decision trees, random forests can be used for both classification and regression problems.

Random forests are less likely to overfit than decision trees, because multiple trees are being grown and used when making predictions. Random Forests also do not consider all variables at every split, but randomly choose a subset of variables when deciding on how to split the data. This helps to prevent overfitting as well. With random forests leveraging individual decision trees, random forests also offer some of the strengths associated with decision trees: they do not require linear relationships between input and output variables making them more robust. They also do not require any normalization of the features.

With random forests leveraging decision trees, random forests also share some of the weaknesses of decision trees (not ideal for regressions problems, model can not be updated but has to be completely retrained). Another weakness of random forests are that the easy visualization and interpretation of decision trees becomes harder because one would have to look at all trees in order to visualize the prediction process.

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

I choose to apply a random forest because we are facing a classification problem and I wanted to see if and/or by how much a random forest could improve the F1 score in comparison to a single decision tree. The idea was that the decision tree might be overfitting because of the high F1 score on the training set and the lower F1 score on the test set. With random forests averaging predictions across multiple trees, a random forest could reduce the overfitting and lead to better predictions on the test set. This

could potentially offset the reduced interpretability if the board is more interested in accurate predictions than understanding the important variables.

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

Random Forest	Training Time	Prediction Time	F1 Score on Training Set	F1 Score on Test Set
Complete Training Set	0.01014	0.000938	0.99	0.71
30% Training Set	0.010695	0.0009	0.99	0.8
50% Training Set	0.011006	0.001055	1	0.78
70% Training Set	0.010692	0.002145	0.98	0.75

Support Vector Machine (SVM):

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

Support Vector Machines are typically used for binary classification problems such as our problems. SVMs can be used for classification problems involving more than two groups as well as for regression problems, but these tasks require some modifications to the approach, e.g., training multiple binary classifiers.

One of the SVM's strengths is that it is easy to implement as an off-the-shelve solution and can also easily be tuned with a few parameters (C and r). Through these parameters users can choose between hard-margin (every data point has to be classified correctly) and a soft-margin (some data points can be classified incorrectly) SVMs which can be helpful for preventing overfitting. SVMs also do not just find a local optima, but solve the equation for the global optima. By using the kernel trick, users can leverage their domain knowledge and can use SVMs to classify non-linearly separable data as well.

The weaknesses of SVMs include that they are not well suited (at least out-of-the-box) for classification problems involving more than two groups. Another weakness of SVMs is the importance of the kernel function. As mentioned above, the kernel function can be used to insert domain knowledge which is an advantage of SVMs. However, one kernel function can lead to very different results for two different problems. Therefore, the user has to experiment with the kernel function in order to maximize accuracy.

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

I choose to apply a SVM because we are facing a binary classification problem and because we have relatively many features in comparison to the number of students in our data set (30 features and 395 students in the complete data set). This could potentially lead to overfitting (course of dimensionality). Because SVMs allow for a relatively easy implementation of regularization through the parameter C, this model seemed liked a good candidate for our data set.

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

SVM	Training Time	Prediction Time	F1 Score on Training Set	F1 Score on Test Set
Complete Training Set	0.005435	0.001416	0.87	0.78
30% Training Set	0.001555	0.001002	0.87	0.67
50% Training Set	0.002699	0.001923	0.89	0.81
70% Training Set	0.004324	0.003287	0.89	0.81

5. Choosing the Best Model

Question: Based on the experiments you performed earlier, 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 recorded to make your case.

Based on the experiments you performed earlier, 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 recorded to make your case.

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).

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?

Answer:

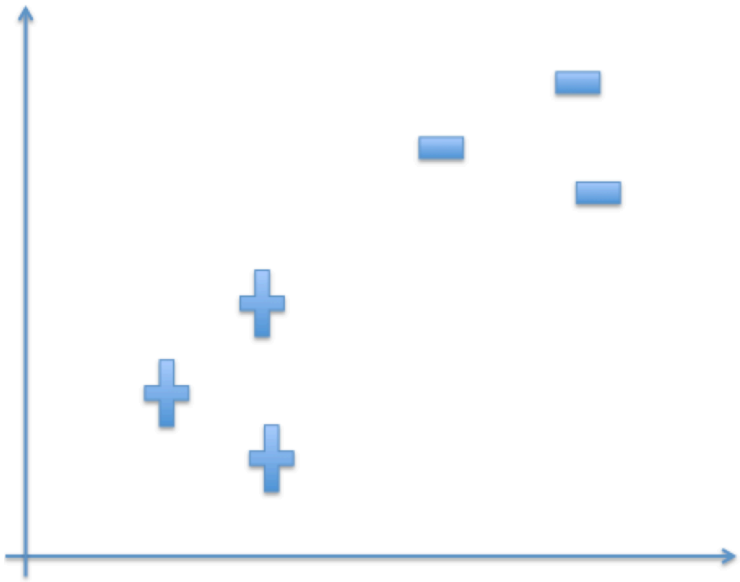
Based on the tests, the support vector machine was chosen as the best model. Except for one training set size (30% of the training data), it consistently showed the best F1 score on the test set. For example, the SVM's F1 score on the test set after training on 70% of the available training data was 0.81. The random forest and decision tree only realized F1 scores of 0.75 and 0.64 respectively in the same scenario.

While the SVM showed the most accurate predictions, it was not the fasted model in regards to prediction and training time. It had the worst average predicting time (0.001907 vs. 0.0004595 for random forest and 0.00009175 for decision tree) across all scenarios as well as only the second best average training time across all training set sizes (0.00350325 vs 0.00520875 for random forest and 0.000704 for decision tree).

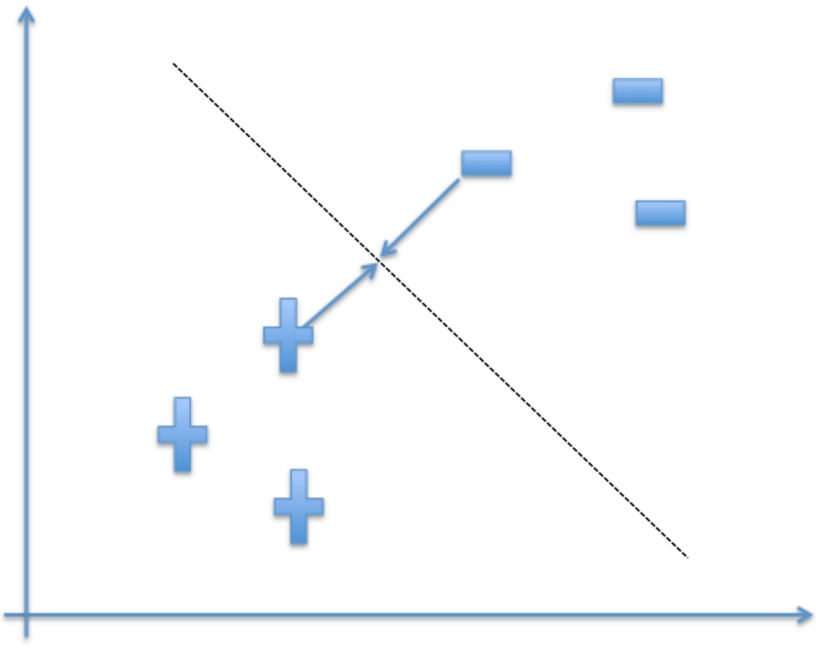
Weighing the accuracy of the predictions on the test set the highest, I would suggest the SVM as the best model to the board of supervisors. The F1 score for the SVM was a lot higher than the score from the other two models while the predicting and training times were not unbearable high.

How does a support vector machine work?

We are trying to classify students as a) students in need of intervention, or b) students not in need of intervention. Since we are only differentiating between two classes (cases a and b), we can represent every student with a plus or minus sign indicating the need for intervention. If we then visualize every student in a chart, we would get a chart similar to the following:



A support vector machine would now try to find a line that separates both classes (the plus and minus signs) from each other. There are many lines that would separate the two classes from each other. A support vector machine finds the line that best separates both classes from each other. The support vector machine does that by choosing the line that maximizes the distance between a class' closest point and the line. This distance is called the support vector.



By adjusting the model's parameters one can specify if the SVM is allowed to misclassify a few data points in order to avoid overfitting, i.e., giving a few outliers too big of an impact on defining the separating line.

Once the SVM is trained, a prediction is made by checking on which side of the separating line the data point (a student in our case) falls. If the student – based on her attributes - gets placed on the left side of the line, she will be classified as a student in need of intervention. If the student gets placed on the left side of the line, she will be classified as a student not in need of intervention.

Fine-Tuning the model

After selecting the SVM as the best model, I performed fine-tuning of the model by using gridsearch. I specified the parameters C, kernel, and gamma to be fine-tuned and defined the values to be tested as:
First option: 'C': [10, 100, 1000], 'kernel': ['linear'],
Second option: 'C': [10, 100, 1000], 'gamma': [0.001, 0.0001], 'kernel': ['rbf']

Grid search identified the following parameters as the best:
{'gamma': 0.0001, 'kernel': 'rbf', 'C': 100}

With the parameters set to these values, the model's final F1 score was 0.82 which is a slight improvement over the previous F1 score of 0.81.

Sources:
Decision Trees:
http://www.simafore.com/blog/bid/62333/4-key-advantages-of-using-decision-trees-for-predictive-analytics
http://stats.stackexchange.com/questions/1292/what-is-the-weak-side-of-decision-trees
Random Forest:
https://en.wikipedia.org/wiki/Random_forest
SVMs:
http://www.cs.rutgers.edu/~mlittman/courses/ml04/svm.pdf
http://stats.stackexchange.com/questions/35276/svm-overfitting-curse-of-dimensionality
http://www.nickgillian.com/wiki/pmwiki.php/GRT/SVM#Disadvantages
https://www.cs.cmu.edu/~ggordon/SVMs/new-svms-and-kernels.pdf
Udacity: Kernel Methods and SVMs Extension (PDF)