Project 2: Supervised Learning - Building a Student Intervention System

by Mingming Guo

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?

Answer: this problem is a classification problem because the response values are categorical labels, while the response values are continuous in regression problem.

2. Exploring the Data

Can you find out the following facts about the dataset?

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

3. Preparing the Data

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.

4. Training and Evaluating Models

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

1) Naive Bayes (sklearn.naive_bayes.GaussianNB())
What are the general applications of this model? What are its strengths and weaknesses?
Answer: the general applications of naive bayes model are for classification problems like spam detection, linguistics and document classification, etc. Its strengths are: running fast, easy to implement, and requiring only a small set of training data to estimate necessary parameters, etc. Its weaknesses are: it can't learn the interactions between features since it assume the independence between pairs of features

Given what you know about the data so far, why did you choose this model to apply?
Answer: From the data, we see most of the features are categorical features. It seems most of features are independent. This is suitable for naive bayes algorithm to calculate the conditional

probability. We also only have 300 training examples and 30 features which is a small dataset. So it's a good fit to try naive bayes for this problem.

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 Size	Training Time (secs)	Prediction Time for Training Data (secs)	Prediction Time for Test Data (secs)	F1-score on Training Set	F1-score on Test Set
100	0.001	0.000	0.000	0.8032	0.7716
200	0.001	0.000	0.000	0.7808	0.6991
300	0.001	0.000	0.000	0.7755	0.7460

2) Support Vector Machine (sklearn.svm.SVC())
What are the general applications of this model? What are its strengths and weaknesses?
Answer: Support Vector Machine is generally used for classification, regression and outliers detection. Its strengths are: it works well in high dimensional spaces, and has different kernel functions to handle different decision function, and it's memory efficient, etc. Its weaknesses are: it does not give probability estimates, and has poor performance when the number of samples is much smaller than the number of features, and it's running slowly given a large dataset, etc.

Given what you know about the data so far, why did you choose this model to apply?
Answer: Given that the dataset is a small one, and the number of samples are much greater than the number of features, support vector machine is a good model to try for applying to this data since it will not like overfitting the data.

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 Size	Training Time (secs)	Prediction Time for Training Data (secs)	Prediction Time for Test Data (secs)	F1-score on Training Set	F1-score on Test Set
100	0.001	0.001	0.001	0.8428	0.8187
200	0.003	0.002	0.001	0.8464	0.8266

Training Size	Training Time (secs)	Prediction Time for Training Data (secs)	Prediction Time for Test Data (secs)	F1-score on Training Set	F1-score on Test Set
300	0.006	0.005	0.002	0.8552	0.8684

3) Logistic Regression Classifier (sklearn.linear_model.LogisticRegression)
What are the general applications of this model? What are its strengths and weaknesses?
Answer: Logistic Regression is used for classification problem. Its strengths are: its robust to the individual features that are not normally distributed, it gives the probability estimates, it's intrinsically simple, running fast, and it has low variance thus less prone to overfit. Its weaknesses: it assumes that there is a smooth linear decision boundary within the given data which may not be true, and its performance decreases if there are features that have collinearity relationship, etc.

Given what you know about the data so far, why did you choose this model to apply?
Answer: Given the data, we see lots of categorical features that are not normally distributed, and logistic regression is robust to this situation. Moreover, we can use logistic regression as the benchmark because it's simple and easy to understand, even if the linear relationship assumption is not hold within the data.

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 Size	Training Time (secs)	Prediction Time for Training Data (secs)	Prediction Time for Test Data (secs)	F1-score on Training Set	F1-score on Test Set
100	0.041	0.000	0.000	0.8760	0.6942
200	0.009	0.000	0.000	0.7816	0.7619
300	0.049	0.000	0.000	0.8125	0.8270

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

5. Choosing the Best Model

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 recored to make your case.

Answer: Comparing the F1 score on test set based on full size training data, we see the Naive Bayes is 0.7460, SVM is 0.8684, and Logistic Regression is 0.8270. So the best F1 score is from Support Vector Machine.

Comparing the training time based on full size training data, we see Naive Bayes is 0.001 seconds, SVM is 0.006 seconds and Logistic Regression is 0.049 seconds. We see Naive Bayes and SVM are pretty fast. The prediction times on test data are 0 seconds, 0.002 seconds and 0 seconds separately which are all fast.

Based on the available data, time efficiency, and performance measured by F1 score, Support Vector Machine is the best and appropriate 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).

Answer: First, we use a simple example to introduce Support Vector Machine mechanism. Given a binary classification problem and two classes can be separated by a linear line. From figure 1, we see two classes, one is labeled with green color and the other is labeled with red color. SVM tries to build two lines to maximize the distance between the two classes. The two lines are called Margins. SVM uses the sample data points near the boundary from both classes which called support vectors to construct the margins with maximal distance. In figure 1, we see the two green points and one red point on the margins are Support Vectors. SVM constructs the decision boundary which is a line paralleling to two margins that already built, but located in the middle of the two margins. Then, if a new data point comes, SVM will use this decision boundary to classify it and predict which class it belongs to by judging which side the new point is located at.

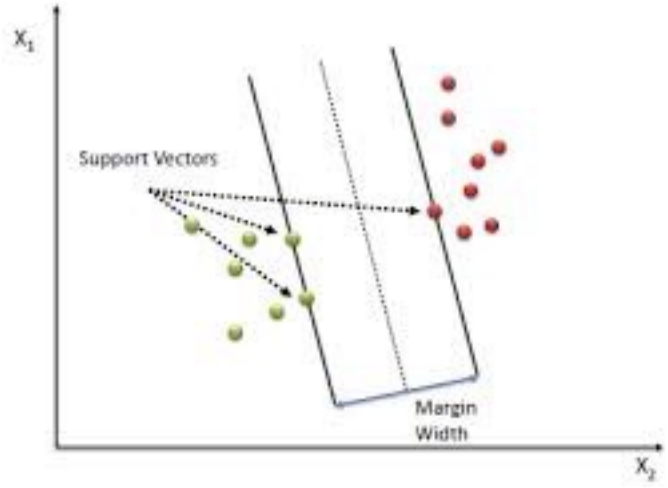

Figure 1

In addition, we will use two figures to illustrate the kernel trick of SVM. For example, if we have two classes that cannot be single separated by a smooth linear line in two dimension. From figure 3, we see there is no a linear line can separate the two classes. SVM will try to project the data into three dimension like figure 2. Then, SVM draw a smooth linear surface to separate the two classes in this three dimensional space. Then, we draw a non-linear line in the two dimensional space according to the linear surface we draw in three dimensional space. We can see this non-linear line can perfectly separate two classes. We can use this line to classify new coming data by judging which side it close to. This is the magic way how SVM works for non-linear classification problem by using the kernel trick. Due to SVM's complexity, it work wells for small or median size of dataset. For a very large dataset containing millions of data points, it's not suitable to use SVM because its running speed is relatively slow.

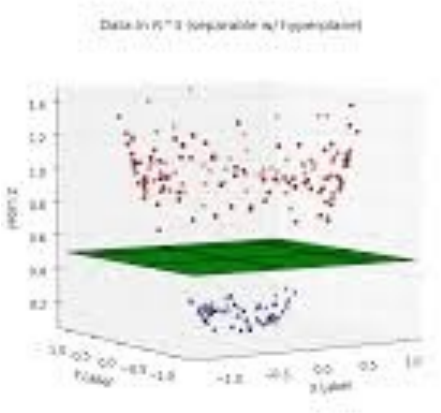

Figure 2

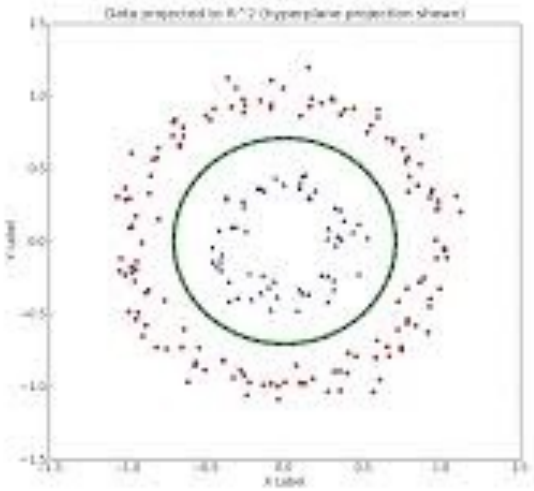

Figure 3

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: the Support Vector Machine learning model's final F1 score is 0.868421052632.