Student Intervention System

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?

- This is an example of a classification supervised learning problem because the output labels are discrete (“yes”, “no”) and not continuous.

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%
- 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:

- 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.

- Model 1: Decision Tree
  - General applications
    - It is heavily used in data mining applications.
    - It can be applied to physics when attempting to detect physical particles.
    - It can also be used as a spam filter.
  - Strengths
    - They are easy to understand.
    - They can be very easy to visualize.
  - Weaknesses
    - Prone to overfitting
  - Based on the data, why I chose this model:
    - Since the output labels in the student data are discrete values ("yes", "no"), decision trees are a reasonable approach because you do not need a large number of leaf nodes after each decision. You can easily follow a path from root to any leaf node which would yield a yes/no result.

Decision Tree Performance Metrics				
Training Set Size	Training Time	Prediction Time	Training: F1 Score	Testing: F1 Score
100	.001	.000	1.000	0.627
200	.001	.000	1.000	0.740
300	.001	.000	1.000	0.610

- Model 2: Support Vector Machine
  - General applications
    - Support vector machines are great at recognizing handwritten characters.
    - It is also useful when trying to label or classify images.
  - Strengths
    - It performs very well when the data can be split with a decent margin in between classified regions.
  - Weaknesses
    - It can be slow on training sets with large amounts of data.
    - It can be very sensitive to outliers. If there is too much noise in the dataset, it will have a lower rate of success.
  - Based on the data, why I chose this model:
    - The size of the data set is relatively small, and support vector machines do not work well with larger data sets.
    - The output of the dataset are a discrete set of labels: support vector machines are great for classification problems. Since there are only two labels in the output, it will be easier to split the data space into two regions.

Support Vector Machines Performance Metrics				
Training Set Size	Training Time	Prediction Time	Training: F1 Score	Testing: F1 Score
100	.001	.001	0.878	0.775
200	.003	.002	0.868	0.781
300	.008	.004	0.876	0.784

- Model 3: K-Nearest Neighbors
    - General applications
        - It can be applied to pattern recognition problems, DNA sequencing, and even for spell checking.
    - Strengths
        - The training time is marginal since it is merely storing the data into memory.
        - It can take advantage of when new data points are added to the training set without re-training the model.
    - Weaknesses
        - The amount of time necessary to make a prediction can be slow since it has to query the entire data set.
        - This algorithm uses up a lot of memory to store all the results.
    - Based on the data, why I chose this model:
        - The size of the data set is small, so the prediction time would not be as much of an issue.

K-Nearest Neighbors Performance Metrics				
Training Set Size	Training Time	Prediction Time	Training: F1 Score	Testing: F1 Score
100	.001	.001	0.806	0.725
200	.001	.002	0.88	0.769
300	.001	.005	0.881	0.780

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 recorded to make your case.

- The Support Vector Machine model had the best F1 score on the test set of 0.784 (with a training size of 300). The Decision Tree model and k-Nearest Neighbors model tied for having

the fastest training times of 0.001 seconds for each training size. The Decision Tree model also had the fastest prediction times of (0.000 seconds for each training size.

- The Decision Tree model had the fastest training times at 0.001 seconds for each training size, but it also had the lowest overall F1 scores on the test set. With a training set size of 300, the Decision Tree model had an F1 test score of 0.61, the Support Vector Machine model had an F1 test score of 0.784, and the k-Nearest Neighbors model had an F1 test score of 0.780.
- K-Nearest Neighbors doesn't make as much sense for this data set because there are a rather large number of features (30) which puts it in a higher dimension space. Dimension reduction would need to be performed prior to applying the K-NN algorithm.
- Based on our experiments we chose the Support Vector Machine algorithm as the best model. It uses up more resources during the training phase, but it gave a good balance between prediction times and F1 scores. Since it is not a lazy learning algorithm, it can free up the data resources after the training phase and rely on the model (opposed to k-nn which needs to store all data points for predictions).

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).

- Let's start with a simple example of using only two features from our data set: number of absences and student's age, and the associated output label (pass/fail label). If we plot these features onto a graph with an x-y axis, our goal is to find a line that splits the data points into a section with pass data points and a section with fail data points. The line we choose should split the data and classify as many data points correctly as possible while also maximizing the distance to the closest data points (this is called the **margin)**. The farther the distance between the separating line and the data points, the more likely we are to make a correct prediction. When making a prediction on a new student, we can plot their age and number of absences on the graph and determine which side of the line they fall on.
- However, we have 30 features to choose from in this dataset which makes it very difficult to graph in a dimension we can comprehend. We are still looking to find a separation of the pass vs. fail data points, but in a higher dimensions (this is called a **hyperplane**). We also want to maximize the distance between the hyperplane and the nearest data points.
- If the data point labels can not be separated by the hyperplane with the current features, we apply something called the **kernel trick.** This allows us to take the data points and map it into a higher dimensional space (additional features) where the data points are separable. We then take the higher dimension solution and map it back to the original feature space where we now have a non-linear separable solution that appropriately separates the labels. We then apply this model on new data points and predict if the student will pass or fail.

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?

- The Support Vector Machine's final F1 score on the test set (using a training sample size of 300) is 0.789.