Questions and Report Structure

1. Classification vs Regression

Its a classification problem since we need to find out whether a student will pass the exam or not.

This is a classic case of Classification where we need to divide students in 2 classes : Pass or Fail

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 : **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 is both the theoretical space complexity to represent the model and the time for the algorithm to make a prediction? You can either provide the big-O notation, or list several of the major features that may affect the algorithm and state if the largest driving factor is constant, linear, logorithmic, polynomical, etc in nature.

Complexity	Space	Time
DecisionTree	O(2^n) : n is the number of features. For decisiontree classifiers, every node can branch out in 2 ways therefore 2 ^n ways.	O(n) : n is the number of features. Given a list of features decision needs to be made about yes or a no.
KNN	O(n) : n is the number of training data or points	O(logn) + k where n is the number of total points or traning data and k is the number of neighbours we need in knn
SVC	1 or constant (it will be a line or a quadratic or polynomial equation which separates space into 2 parts	1 or constant time to predict

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

	Applications	**Strengths**	**Weaknesses**
DecisionTree	1) Classifications, like predicting a simple yes or a no.	1) Simple to understand and visualize	Overfitting, setting max depth is important otherwise can be complicated.

Knn	Housing prices as houses belonging to a particular location have somewhat similar rates and can be easily predicted looking at the neighbours	1) Very simple algorithm. Lazy prediction therefore learning time in constant	Its Lazy, suffers from noise and bias.
SVC		1) Memory efficient, use a subset of points in decision making	Large number of parameter tuning is needed so that the model do not overfit or bias. For non linear planes, difficult to visualize

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

	Reason to choose this model
DecisionTree	1) Simple to visualize. All features are divided into yes or no so suits the input labels perfectly with the DecisionTree
KNN	1) Students with similar features tend to show similar behaviour and pattern, so finding nearest neighbours can be very efficient.
SVC	1) SVC clearly divides the plane into 2 halves or classes. There are a lot of paramters that we can tune up to solve problems of overfitting.

- 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.
    -

DecisionTreeClassifier	Training set size		
	100	200	300
Training time (secs)	1.448	1.52	1.56
Prediction time (secs)	.002	.001	.001
F1 score for training set	.832	.835	.83
F1 score for test set	.77	.76	.8

KNearestNeighbour	Training set size		
	100	200	300
Training time (secs)	1.75	1.832	1.909
Prediction time (secs)	0.004	.007	.02
F1 score for training set	.79	.89	.85
F1 score for test set	.74	.74	.78

SVC	Training set size		
	100	200	300
Training time (secs)	7.9	18.13	37.56
Prediction time (secs)	.003	.005	.009
F1 score for training set	.92	.86	.87

F1 score for test set	.75	.78	.78

○

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 1-2 paragraphs explain to the board of supervisors what single model you chose as the best model. Which model is generally the most appropriate based on the available data, limited resources, cost, and performance?

Based on the available data, limited resources, cost and performance, decision tree seems to be the best model as it gives the best test prediction, as well as provides this F1 score at max depth of 2, hence for this depth, it will not be taking that much of cpu time to learn and then predict.

In 1-2 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 make a prediction).

Decision Tree builds a tree of yes and no with feature as nodes and yes and no or 1 and 0 as edges. It is very simple to visualize. Once the model is well trained, we can pass on the values of the test set and then with a simple decision based find out the 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.

I have used GridSearchCV above in my code with cv as 5, mkscorer as F1-score and parameters based on the classifier.

What is the model's final F1 score?

Final F1 score is provided in the table above