The University of Melbourne
School of Computing and Information Systems
COMP90049: Introduction to Machine Learning, 2020 Semester 1

Project 1: Understanding Student Success with Naive Bayes

Due:	Wednesday 22 April 11am
Submission:	Python Source code and inline responses in the provided Jupyter Notebook
Marks:	The project will be marked out of 20, and will contribute 20% of your total mark. This will be equally weighted between implementation and responses to the questions. This is an **individual** assignment which you are expected to complete on your own.

Overview

In this project, you will implement a Categorical Naive Bayes classifier. You will explore its inner workings and evaluate its behavior on a data set of student performance in high school, and on this basis respond to some conceptual questions.

Implementation of Naive Bayes

The lectures and workshops contained several pointers for reading in data and implementing your Naive Bayes classifier.

- **You must implement your Naive Bayes classifier from scratch**. Solutions which use non-native Python libraries in the implementation of the Naive Bayes `train()` or `predict()` function(s) will be treated as a **fail**.
- You may, however, use any the following Python libraries of your choice for implementing evaluation metrics and procedures and data processing:
  - `pandas: https://pandas.pydata.org/`
  - `scikit learn: https://scikit-learn.org/`
  - `numpy: https://numpy.org/`
  - `scipy: https://www.scipy.org/`
  - any of Python's native libraries: `https://docs.python.org/3/library/`
- Beyond the above white-list, **no** existing code or libraries may be used.
- Note that you are not required to use any (or all!) of these libraries. This assignment can easily be completed without the use of any Python library whatsoever.

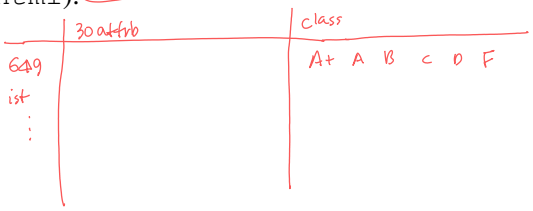
For marking purposes, a minimal submission should define the following functions:

- `load_data()`, which opens the data file, and converts it into a usable format
- `split_data()`, where you split your data sets into a training set and a test set.
- `train()`, where you build a Naive Bayes model from the training data
- `predict()`, where you use a trained model from `train()` to predict a class (or class distribution) for the test data
- `evaluate()`, where you will output the **accuracy** of your classifiers

You may alter the above prototypes to suit your needs; you may write other helper functions as you require. Depending on which answers you choose to respond to, you may need to implement additional functions.

Data

For this project, we have adapted the Student data set available from the UCI machine learning repository (`https://archive.ics.uci.edu/ml/index.html`).



Some critical information

1. File containing the data set: `student.csv`
2. 649 instances
3. 30 nominal attributes. Note that although some attributes may equally be treated as ordinal, for the purpose of this assignment we assume all attributes are nominal.
4. 6 classes, corresponding to predicted final grade: {A+, A, B, C, D, F}
5. `student.txt` explains the attributes and class-labels

Questions

You should respond to **Question 1** and **two additional questions of your choice**. A response to a question should take about 100–250 words, and make reference to the data wherever possible.

Question 1: Naive Bayes Concepts and Implementation

a Explain the 'naive' assumption underlying Naive Bayes. (1) Why is it necessary? (2) Why can it be problematic? Link your discussion to the features of the `students` data set. *[no programming required]*

b Implement the required functions to load the student dataset, and estimate a Naive Bayes model. Evaluate the resulting classifier using the hold-out strategy, and measure its performance using accuracy.

c What accuracy does your classifier achieve? Manually inspect a few instances for which your classifier made correct predictions, and some for which it predicted incorrectly, and discuss any patterns you can find.

Question 2: A Closer Look at Evaluation

a You learnt in the lectures that `precision`, `recall` and `f-1` measure can provide a more holistic and realistic picture of the classifier performance. (i) Explain the intuition behind accuracy, precision, recall, and F1-measure, (ii) contrast their utility, and (iii) discuss the difference between micro and macro averaging in the context of the data set. *[no programming required]*

b Compute `precision`, `recall` and `f-1` measure of your model's predictions on the test data set (1) `separately` for each class, and (2) as a single number using `macro-averaging`. Compare the results against your accuracy scores from Question 1. In the context of the student dataset, and your response to question 2a analyze the additional knowledge you gained about your classifier performance.

Question 3: Training Strategies

There are other evaluation strategies, which tend to be preferred over the `hold-out` strategy you implemented in Question 1.

a Select one such strategy, (i) describe how it works, and (ii) explain why it is preferable over hold-out evaluation. *[no programming required]*

b Implement your chosen strategy from Question 3a, and report the `accuracy` score(s) of your classifier under this strategy. Compare your outcomes against your accuracy score in Question 1, and explain your observations in the context of your response to question 3a.

Question 4: Model Comparison

In order to understand whether a machine learning model is performing satisfactorily we typically compare its performance against alternative models.

a Choose one (simple) comparison model, explain (i) the workings of your chosen model, and (ii) why you chose this particular model.

b Implement your model of choice. How does the performance of the Naive Bayes classifier compare against your additional model? Explain your observations.

Question 5: Bias and Fairness in Student Success Prediction

As machine learning practitioners, we should be aware of possible ethical considerations around the applications we develop. The classifier you developed in this assignment could for example be used to classify college applicants into `admitted` vs `not-admitted` – depending on their predicted grade.

a Discuss ethical problems which might arise in this application and lead to unfair treatment of the applicants. Link your discussion to the set of features provided in the `students` data set. *[no programming required]*

b Select ethically problematic features from the data set and remove them from the data set. Use your own judgment (there is no right or wrong), and document your decisions. Train your Naive Bayes classifier on the resulting data set containing only 'unproblematic' features. How does the performance change in comparison to the full classifier?

c The approach to fairness we have adopted is called "fairness through unawareness" – we simply deleted any questionable features from our data. Removing all problematic features does **not** guarantee a fair classifier. Can you think of reasons why removing problematic features is not enough? *[no programming required]*

Submission

Submission will be made via the LMS, as a single jupyter notebook file.
You will use the iPython notebook `S2020S1-proj1.ipynb`, which summarizes the five core functions, as a template. You will type your answers to the text-based question into cells in the jupyter notebook.

Assessment

5 of the marks available for this project will be assigned to whether the five specified Python functions work correctly. Any other implementation will not be directly assessed (except insofar as it is required to make these five functions work correctly).

15 of the marks will be assigned to accurate and insightful responses to the questions, divided evenly among the questions that you are required to attempt. We will be looking for evidence that you have an implementation that allows you to explore the problem, but also that you have thought deeply about the data and the behavior of the relevant classifier(s).

Late Submission Policy

There will be **no extensions** granted, and **no late submissions** allowed. Submission will close at 11am on Monday, April 22. For students who are demonstratably unable to submit a full solution in time, we offer to reduce the weighting of the mark of this assignment towards the overall course grade (but you will still have to submit your solutions by the deadline).

Changes/Updates to the Project Specifications

If we require any (hopefully small-scale) changes or clarifications to the project specifications, they will be posted on the LMS. Any addendums will supersede information included in this document.

Academic Misconduct

You are welcome — indeed encouraged — to collaborate with your peers in terms of the conceptualization and framing of the problem. For example, what the project is asking you to do, or what you would need to implement to be able to respond to a question.

However, sharing materials — for example, plagiarizing code or colluding in writing responses to questions — will be considered cheating. We will invoke University's Academic Misconduct policy

(http://academichonesty.unimelb.edu.au/policy.html) where inappropriate levels of plagiarism or collusion are deemed to have taken place.

Data references

P. Cortez and A. Silva.
Using Data Mining to Predict Secondary School Student Performance.
In A. Brito and J. Teixeira Eds., Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008) pp. 5-12, Porto, Portugal, April, 2008, EUROSIS, ISBN 978-9077381-39-7
https://archive.ics.uci.edu/ml/datasets/student+performance#