The University of Melbourne
School of Computing and Information Systems
COMP30027 Machine Learning, 2018 Semester 1

Project 1: What is labelled data worth to Naive Bayes?

Due:	Thurs 29 March 11am
Submission:	Source code (in Python) and (inline) responses
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.
Groups:	You may choose to form a group of 1 or 2. Groups of 2 will respond to more questions, and commensurately produce more implementation.

Overview

In this project, you will implement a supervised Naive Bayes classifier, and an unsupervised Naive Bayes classifier. You will evaluate their behaviour on some standard data sets, and then respond to some conceptual questions.

Naive Bayes classifiers

There are numerous suggestions for implementing your classifiers in the "Project 1 intro" lecture; ultimately, the specifics of your implementation are up to you, and will depend on which question(s) you choose to answer.

For marking purposes, a minimal submission should have a `preprocess()` function, which opens the data file, and converts it into a usable format. It should also define the following functions, for each of the supervised **and** unsupervised NB classifiers:

- `train()`, where you calculate counts (or probabilities) from the training data, to build a model
- `predict()`, where you use the model from `train()` to predict a class (or class distribution) for the test data
- `evaluate()`, where you will output your evaluation metric(s), or sufficient information so that they can be easily calculated by hand

There is a sample iPython notebook `2018S1-proj1.ipynb` that summarises these, which you may use as a template.

You may alter the above prototypes to suit your needs; you may write other helper functions as you require. Depending on what you try to implement, some of the functions may be very similar between the supervised and unsupervised versions.

Data

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

(i) `breast-cancer.csv`, having 286 instances, 9 nominal attributes (some of which contain digits), and two classes: `no-recurrence-events` and `recurrence-events`. A small number of attribute values are missing, and marked with `?`: you may deal with them as you see fit.

(ii) `car.csv`, having 1728 instances, 6 nominal attributes (some of which contain digits), and two classes: `acc` and `unacc`.

(iii) `hypothyroid.csv`, having 3163 instances, 18 nominal attributes, and two classes: `hypothyroid` and `negative`. A small number of attribute values are missing, and marked with `?`: you may deal with them as you see fit.

(iv) `mushroom.csv`, having 8124 instances, 22 nominal attributes, and two classes: `e` (edible) and `p` (poisonous). A large number of attribute values are missing, and marked with `?`: you may deal with them as you see fit.

Questions

The following problems are designed to pique your curiosity when running your classifiers over the given data sets:

1. Since we're starting off with random guesses, it might be surprising that the unsupervised NB works at all. Explain what characteristics of the data cause it to work pretty well (say, within 10% Accuracy of the supervised NB) most of the time; also, explain why it utterly fails sometimes.

2. When evaluating supervised NB across the four different datasets, you will observe some variation in effectiveness (e.g. Accuracy). Explain what causes this variation. Describe and explain any particularly suprising results.

3. Evaluating the model on the same data that we use to train the model is considered to be a major mistake in Machine Learning. Implement a hold–out (hint: check out `numpy.shuffle()`) or cross–validation evaluation strategy. How does your estimate of Accuracy change, compared to testing on the training data? Explain why. (The result might surprise you!)

4. Implement one of the advanced smoothing regimes (add-k, Good-Turing). Do you notice any variation in the predictions made by either the supervised or unsupervised NB classifiers? Explain why, or why not.

5. The lecture suggests that deterministically labelling the instances in the initialisation phase of the unsupervised NB classifier "doesn't work very well". Confirm this for yourself, and then demonstrate why.

6. Rather than evaluating the unsupervised NB classifier by assigning a class deterministically, instead calculate how far away the probabilistic estimate of the true class is from 1 (where we would be certain of the correct class), and take the average over the instances. Does this performance estimate change, as we alter the number of iterations in the method? Explain why.

7. Explore what causes the unsupervised NB classifier to converge: what proportion of instances change their prediction from the random assignment, to the first iteration? From the first to the second? What is the latest iteration where you observe a prediction change? Make some conjecture(s) as to what is occurring here.

If you are in a group of 1, you should respond to question (1), and one other of your choosing. If you are in a group of 2, you should respond to question (1), and 3 others of your choosing. A response to a question should take about 100–200 words, and make reference to the data wherever possible. Note that not all questions are equally difficult. Also note that not all questions are equally interesting. `(-:`

Submission

Submission will be made via the LMS, as a single file or archive of files. Submissions will open one week before the submission deadline.

Assessment

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

10 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 behaviour of the relevant classifier(s).

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 conceptualisation 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 beyond your group — for example, plagiarising 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. taken place.

1 Data references

`breast-cancer` is thanks to:

> Matjaz Zwitter & Milan Soklic (physicians)
> Institute of Oncology
> University Medical Center
> Ljubljana, Yugoslavia

`car` is derived from:

> Zupan, Blaz, Marko Bohanec, Ivan Bratko, and Janez Demsar (1997) Machine Learning by Function Decomposition, in *Proceedings of the International Conference on Machine Learning*, Nashville, USA.

`hypothyroid` is derived from:

> Quinlan, J. Ross (1986) Induction of Decision Trees, in *Machine Learning*, Vol. 1, pp. 81–106.

`mushroom` is derived from:

> Schlimmer, Jeff (1987) *Concept Acquisition through Representational Adjustment*, Technical Report No. 87-19, University of California, Irvine.