Homework 8

Due: Thursday, March 20, 2025 at 12:00pm **(Noon)**

Programming Assignment

Introduction

In this assignment, you'll implement Naive Bayes and use this algorithm to classify the credit rating (good or bad) of a set of individuals. The textbook section relevant to this assignment is 24.2 on page 347.

Stencil Code & Data

We have provided the following two files:

- `main.py` is the entry point of your program which will read in the data, run the classifiers and print the results. Note that pre-processing has been done for you; feel free to examine the code for what exactly was done.
- `models.py` contains the `NaiveBayes` model which you will be implementing.

You should *not* modify any code in the `main.py`. All the functions you need to fill in reside in `models.py`, marked by `TODO`s. You can see a full description of them in the section below. To feed the data into the program successfully, please do *not* rename the data files and also make sure not to move either the data file or `models.py` from the stencil. To run the program, run `python main.py` in a terminal.

You can find and download the stencil code here: `HW8 on Github`. If you have any problems, please consult the `Download and Handin Guide`.

German Credit Dataset

You will be using the commonly-used German Credit dataset, which includes 1000 total examples. The prediction task is to decide whether someone's credit is good (1) or bad (0). A full list of attributes can be found **here**; note that this includes sensitive attributes like sex, age, and personal status. The specific file we are using comes from **Friedler et.al., 2019**. This data is in the file `german_numerical-binsensitive.csv`.

Data Format

The original feature values in this dataset are mixed—some categorical, some numerical. We have written all the preprocessing code for you, transforming numerical attributes into categories and encoding all attributes as binary features. After preprocessing, there are a total of 69 attributes which take on either 1 or 0. `credit = 1` **corresponds to "good" credit, and** `credit = 0` **corresponds to "bad" credit.**

The Assignment

In `models.py`, there are three functions you will implement. They are:

- `NaiveBayes`:
  - **train()** uses maximum likelihood estimation to learn the parameters (attribute distributions and priors distribution). Because all the features are binary values, you should use the Bernoulli distribution (as described in lecture) for the features. Remember to add Laplace smoothing as you calculate the distributions.

- **predict()** predicts the labels using the inputs of test data. You should return 1-D numpy array.
- **accuracy()** computes the percentage of the correctly predicted labels over a dataset.

Note that there is also a **print_fairness()** method implemented for you in `NaiveBayes`. You should not change this method. Additionally, you are not allowed to use any off-the-shelf packages that have already implemented Naive Bayes, such as scikit-learn; we're asking you to implement it yourself.

Note: Depending on whether or not you assume that the inputs are given in log-space, the autograder may print out a message pointing this out. Know that this is not an error message and should be taken as informational.

Project Report

1. Report the training and testing accuracy of the Naive Bayes classifier. (A correct implementation should have testing accuracy above 70%.) (2 point)

2. What strong assumption about the features/attributes of the data does Naive Bayes make? Comment on this assumption in the context of credit scores. (6 points)

   Solution: Every feature is independent given the class. Probably doesn't hold true in the context of credit scores.

3. This dataset was originally structured as follows:

Month	Credit Amount	Number of credits	...	Credit
6	1169	2	...	1
48	5951	1	...	2
12	2096	1	...	1
9	2134	3	...	1

   For each of the above attributes, describe what transformations to the original dataset would need to occur for it to be usable in a Bernoulli Naive Bayes model. *(hint: every attribute must take on the value of 0 or 1)* (10 points)

   Solution: Discretize "month" and "credit amount"; binarize "month," "credit amount," "number of credits"; switch "credit" encoding to 0-1.

4. Restate the definition of Disparate Impact from lecture (also included in code comments); make sure to notate what each variable (e.g. S) represents. Why might this be a useful measure of model performance? What are some limitations of this measure? (10 points)

   Solution:

$$\frac{P(\hat{Y} = 1 | S = 1)}{P(\hat{Y} = 1 | S = 0)}$$

   $\hat{Y} = 1$ indicates the "good" decision, e.g., "good credit." S is the sensitive attribute, and S = 1 indicates membership in the "disadvantaged" group while S = 0 means "privileged."

   Useful – provides one lens into why a model can be interpreted as discriminatory (outcome-based); often we do care that the outcomes of two groups "match" to an extent. (Also, since this can also be calculated on the data and not just the model output, we have some sense of if the model is improving/worsening the discrimination already in the dataset.)

   Limitations – Doesn't provide the full picture: if the underlying distributions of the two groups are very different, then a "perfect" DI score could actually be very undesirable – e.g. giving one group a lot of false positives and another a lot of false negatives.

5. A different way to think about fairness is based on the errors the model makes. We define the false positive rate (FPR) as $P(\hat{Y} = 1|Y = 0)$, and the false negative rate (FNR) as $P(\hat{Y} = 0|Y = 1)$. Suppose we calculate FPR and FNR for each group. In words, what does the false positive rate and false negative rate represent in the context of credit ratings? What are the implications if one group's FPR is much higher than the other's? What are the implications if one group's FNR is much higher than the other's? (12 points)

   Solution: FPR: fraction of the time someone was given "good credit" when they actually had "bad credit"

   FNR: fraction of the time someone was given "bad credit" when they actually had "good credit"

   FPR disparity: one group gets more access to credit/loans/etc than they should, unfairly rewarding those who are "undeserving" in the group with higher FPR

   FNR disparity: one group gets less access to credit/loans/etc than they should, unfairly punishing those who are "deserving" in the group with higher FNR

Grading Breakdown

We expect the accuracy that `NaiveBayes` reaches should be above 70%, on both training and testing. Note that your results may fluctuate each time you run the program, as there is some stochasticity in the preprocessing of the data. As always, you will primarily be graded on the correctness of your code and not based on whether it does or does not achieve the accuracy target. The autograder does not check for the accuracy threshold, but instead checks for correct implementation and prediction results.

The grading breakdown for the assignment is as follows:

Naive Bayes	60%
Report	40%
Total	100%

Handing in

You will hand in both your project report and code on Gradescope, separately.

Upload your project report and github repo containing all your source code to Gradescope under "Homework 8 Code". For questions, please consult the download/submission guide.

If you have questions on how to set up or use Gradescope, ask on Edstem!

Anonymous Grading

You need to be graded anonymously, so do not write your name anywhere on your handin.

Obligatory Note on Academic Integrity

Plagiarism—don't do it.

As outlined in the Brown Academic Code, attempting to pass off another's work as your own can result in failing the assignment, failing this course, or even dismissal or expulsion from Brown. More than that, you will be missing out on the goal of your education, which is the cultivation of your own mind, thoughts, and abilities. Please review this course's collaboration policy and, if you have any questions, please contact a member of the course staff.