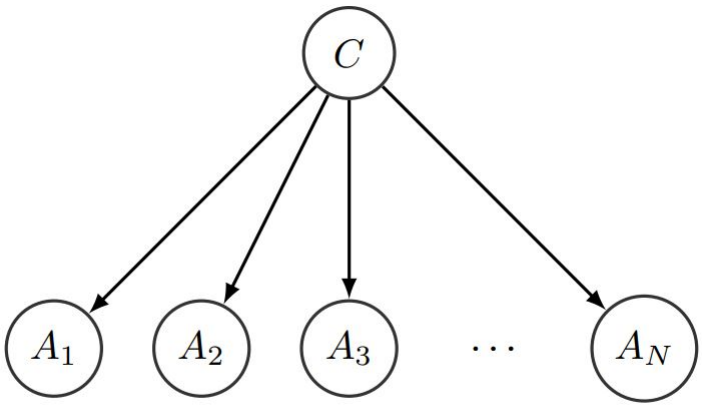
Programming Assignment 1

CMPT 727
Spring 2023

Naive Bayes (NB) classifiers are often competitive classifiers even though their strong independence assumptions may be unrealistic. If C denotes the class variable and $(A_1, ..., A_N)$ the attributes, then a NB model can be represented as a directed graph with these variables as nodes and edges $\{(C, A_i) : 1 \leq i \leq n\}$. The following figure illustrates the graph structure.



In this assignment, you will implement the parameter learning for NB classifiers and use it to compare with logistic regression with LASSO. You will apply these classifiers to predict the party affiliation of either Democrat or Republican of US Congresspeople (the class variable) based on their votes for 16 different measures (the attribute variables) shown in Table 1. Not all congresspeople voted on all 16 measures, so sometimes entries in this dataset will have missing attributes; however, we will still be able to utilize our Bayes Network to accurately classify these examples. To keep things simple, the class and attribute variables are all binary with 0, 1 corresponding to a no and yes vote respectively.

Programming Assignment 1

Attribute	Name	Incomplete Entry 1
A_1	handicapped infants	1
A_2	water project cost sharing	1
A_3	adoption of the budget resolution	0
A_4	physician fee freeze	0
A_5	El Salvador aid	0
A_6	religious groups in schools	0
A_7	anti satellite test ban	0
A_8	aid to Nicaraguan Contras	?
A_9	mx missile	?
A_{10}	immigration	0
A_{11}	synfuels corporation cutback	0
A_{12}	education spending	?
A_{13}	superfund right to sue	?
A_{14}	crime	?
A_{15}	duty free exports	0
A_{16}	export administration act south Africa	0

Table 1: Attribute names for Congressional Voting Records together with an incomplete example that has some voting records missing for a particular Congressman.

We have provided starter code for this assignment [1]. You can download it here.

What to submit

Please submit the following files to CourSYS:

- nb.py – Your completed NB implementation. Do not change the signature of the functions that you were supposed to implement.
- lasso.py – Your completed LASSO implementation using the scikit-learn package.
- report.pdf – A pdf file answering all the questions in this assignment.

Collaboration policy

You may freely discuss this coding assignments with other students. All writing must be your own; it is not acceptable to copy/paste or verbatim transcribe others' text, code or LaTeX source.

(8 points) Question 1

Implement a NB classifier that both learns the parameters from the training data and can use these parameters to score and classify examples in the training data.

[1] This assignment is adapted from Stanford CS 228.

Programming Assignment 1

When training the models, some of the parameters may not have enough examples for accurate estimation. To mitigate this, we will perform Laplace smoothing with a pseudocount of $\alpha = 0.1$ for every value of an attribute given its parents when learning the parameters.

In order to evaluate the performance of our classifiers on the dataset, we will use 10-fold cross-validation. Under 10-fold cross-validation, the dataset is first partitioned into 10 equally sized partitions. Of these 10 partitions, one of them is used for the test set while the rest of the data are used as the training set to compute test error on this partition. This process is repeated for the other nine partitions, and we can take the average of the resulting test errors to obtain the 10-fold CV test error. We have implemented this procedure for you in the function `evaluate`.

What is your test error rate using 10-fold cross-validation?

*Note: you can use the **evaluate** function in the nb.py, but leave the optional argument train subset to its default value until question 3.*

(1 points) Question 2

Using the LASSO classifier from scikit-learn package (set learning rate *alpha* = 0.001), implement 10-fold cross-validation and compute the associated test error. What is your test error rate using 10-fold cross-validation? Which classifier gives a better result?

*Note: We have implemented this for you. Use the **lasso_evaluate** function in the lasso.py, but leave the optional argument train subset to its default value until question 3.*

(2 points) Question 3

To investigate the effect of the amount of training data we have, let's instead use smaller subset of the training data during 10-fold cross-validation. Start with 5 to 10 samples and then increase this number by 10 for each run until we reach 100.

Compute the training error and test error of your trained classifiers. Plot NB training error, NB test error, LASSO test error, and LASSO training error against sample size on the same plot.

What do you observe? Briefly explain the patterns and why we expected them.

*Note: set the arguments train_subset=True, subset_size = x when calling **evaluate** so that the classifiers are trained on data size of x.*

(4 points) Question 4

To evaluate whether our models can learn to base their predictions on the right features, we will create synthetic data. We imagine that bills 1-4 are partisan, with Democrats voting yes 95% of the time and Republicans voting yes 5% of the time. We imagine that bills 5-16 are

nonpartisan, with all members voting either way 50% of the time. We will simulate equal numbers of Democrats and Republicans.

Call **generate_q4_data** function in data_helper.py to create 4000 synthetic congresspeople. Apply LASSO and NB respectively to this data set. Start with 400 samples and increase this number by 400 for each run until we reach 4000.

We will say that LASSO correctly ignores a nonpartisan bill if the associated parameter is 0; we will say that NB does so if the difference between $P(Yes|Democrat)$ and $P(Yes|Republican)$ is less than 0.1. Plot the fraction of nonpartisan bills that are ignored as a function of the training set size. What do you observe?

(2 points) Question 5

In general, working with data where the values of attributes and labels are missing is difficult when learning model parameters. However, we can still use our generative model from a fully trained Bayes Network to classify examples in which some of the attributes may be unobserved. Suppose A_i is unobserved. We can still compute $P(C|A_1, \ldots, A_{i-1}, A_{i+1}, \ldots, A_N)$ by computing $P(C|A_1, \ldots, A_{i-1}, A_i, A_{i+1}, \ldots, A_N)$ and marginalizing over A_i.

Update your NB implementation to handle the case where some attributes may have missing values and use this new implementation to classify Incomplete Entry 1 in Table 1 . Given the observed attributes, what is the marginal probability this Congressperson is Democrat (C = 1) given the votes we did observe? Can you predict how this Congressperson voted on education spending (A12)?

Run LASSO, using the commonly-used strategy of replacing unknown values with 0 ("no"). What is the marginal probability this Congressperson is Democrat (C = 1)?

Note: You should train your classifier on the original dataset.

Question 6

How many hours did you spend on this assignment?