**### 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]**

[1-a-(1)] The ‘naive’ assumption underlying Naive Bayes is the conditional independence assumption which is assuming that all the features are assumed to be independent. This assumption is necessary to make the model more feasible. As it is shown below, this assumption makes the joint model into a ‘countable’ and ‘separated’ probabilistic model.

P(x1, x2, ..., xM|y)P(y) => P(x1|y)P(x2|y)...P(xM|y)P(y)

[1-a-(2)] For instance, both parents’ education and jobs can affect to choose a school and the area where the family lives. Real-world data’s features are related to each other. For another example, travel time must be affected by the distance between their schools and address(home) and it can affect study time as well. Also, these features are contributing to the result (Grade) with different weights. The conditional independence assumption is unrealistic as it ignores the correlations between features.

**- 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.**

[1-b] In my code, “# region [Main]” has “# region [HOLD-OUT]” and “# region [CROSS-VALIDATION]”. If we look at HOLD-OUT Region for this question, it calls

1. load\_data('./Data/student.csv')
2. res = split\_data() :

Since using data that has been used for training to test cannot evaluate the classifier’s performance, I used the hold-out method which separates test and training data. Also I used the commonly used rate 80:20 in the function. Thus the training data range becomes [0-518] and the test data becomes [519-648]. At here, I assumed that the both test and training data sets have the same class distribution.

1. train(res[0])
2. prd\_res = predict(res[1]) :

This function returns predicted class array “prd\_res”. For estimated y, I moved the equation from [P(y) \* P(x|y)] to log space [logP(y) + ∑logP(x|y)]. Even if there is an unseen case, it would not make the whole equation 0 by multiplying it. Therefore, I did not use any smoothing strategy.

1. accr\_res = evaluate(res[1], prd\_res) :

For evaluation, I used accuracy. If this classifier would be used as to accept students for a college, I could consider implementing the Precision method. As calculating A/A+ -> A/A+ as True Positive and A/A+ -> Lower than B as False Positive.

**- 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.**

[1-c]

I got an accuracy of 0.34.

I analyzed instances on an excel file. View online or please find the attached file. <https://docs.google.com/spreadsheets/d/10nAbiyw6Uttmpd9PhWgdBuPxSI1iPDpHMKVaGKjizs8/edit?usp=sharing>

As the file shows, incorrect results tend to show two characteristics. Firstly, if there is a zero counted feature (unseen feature) for a specific attribute, it tends to classified as a class that has the biggest possibility number in that attribute.

For example, if you see the cells [O 27:33] of the excel file, the attribute “failures” has unseen feature A+ and B:

# P(failures=low|A+) = 0, P(failures=low|B) = 0

# P(failures=low|F) = 0.35

In that attribute, P(failures=low|F) = 0.35 has the largest possibility. And this instance is classified as F.

For another example, cells [AC 44:50] show as below. It has an unseen feature and the largest possibility case is A in attribute “health”. And it is classified as A.

# P(health=4|A+) = 0

# P(health=4|A) = 0.22

Secondly, in certain attributes, the wrongly classified class tends to show a large number of observations. And especially if the class has a small number of the total cases, it leads the probability makes a prominent peak (relatively large number / relatively small number). For example,

# P(A) = 50, P(F) = 82

At [B 55:60], these Incorrectly estimated classes have a small total number.

# P(school=MS|F) = P(58/82)

At [B 31:33], P(MS|real class)-P(MS|estimated class) = 0.3

0.3 is a big difference and it gives “F” win.

# P(famsize=GT3|F) = P(66/82)

At [E 31:33], this case has the big difference = 0.11 again.

# [L50] [P50][V50][X50][Y50] cells show that P(estimated case A) – P(real case B) has a big difference and it is estimated as A.

### 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]**

[3-a-(i)] The cross-validation strategy is preferred over the hold-out. It divides N observation into k sets, and each set has N/k observation. A single set is to be a test data set and k-1 sets are training data sets. It iterates k times and uses each of the k sets as a test data set and the rest of them as training data sets.

[3-a-(ii)] This strategy uses every instance as a test instance and a training instance. It can help to solve overfitting, selection bias and fluctuation or error by using all the instances as a training and test instance.

- 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.**

[3-b]

I implemented the cross-validation strategy in the main area of my code. The results are as below.

-------------------------------------------------------------------

N\_SPLIT: 5

Running time: 130 sec

[Accuracy Array]: [0.35, 0.53, 0.53, 0.46, 0.51]

[Averaged Accuracy]: 0.48

-------------------------------------------------------------------

N\_SPLIT: 10

Running time: 136 sec

[Accuracy Array]: [0.28, 0.48, 0.55, 0.48, 0.51, 0.45, 0.45, 0.45, 0.54, 0.5]

[Averaged Accuracy]: 0.47

-------------------------------------------------------------------

N\_SPLIT: 65

Running time: 133 sec

[Averaged Accuracy]: 0.49

-------------------------------------------------------------------

Running time: 176

N\_SPLIT: 649

[Averaged Accuracy]: 0.48

-------------------------------------------------------------------

I tried 4 cases with N\_SPLIT = 5,10,65 and 649(leave-one-out). Each case showed accuracy from 0.47 to 0.49 which is relatively similar to each other. And when we compare the result with the Hold-out’s accuracy 0.34, we can see that 0.34 was a biased result just by the selected test set. The N\_SPLIT=10 case explains this problem. The accuracy result [0.28, 0.48, 0.55, 0.48, 0.51, 0.45, 0.45, 0.45, 0.54, 0.5] has 0.28 and it shows that if I just selected that set as a test set for the hold-out strategy, I would get 0.28 for a result and that would be very biased and not an accurate result. In conclusion, **the cross-validation can help to mitigate bias and get more accurate evaluation result by averaging multiple results.**

### 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]

[5-a] Accepting students based on their gender or their parent’s status (since those features affect calculating Grades in this classifier) can cause discrimination. Features “M/Fjob, M/Fedu, guardian, Pstatus” can lead a consequence that the college selects students based on the presence of parents, their financial status and their education level. Also, the feature “sex” can add a gender preference.

**- 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?**

[5-b] At the function predict\_fair() in my code, I removed 7 feature columns as I mentioned at [5-a]. It showed an accuracy result of 0.41 which is 0.08 under compared to the original result 0.49. As removing 7 attributes out of 29, the classifier got a deterioration by 16 per cent ((0.49-0.41)\*100/0.49).

**- 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]

[5-c] First of all, excluding the controversial feature can be another discrimination. For instance, in our dataset, Female has 44 and male has 18 for A and A+ Grade. If we simply get rid of this observation, it would not be fair to female students. Also, it is ignoring real-life data with social bias and it’s not reflecting the truth truly. Secondly, even if we get rid of the problematic attribute, by ‘red-lining effect’[1], classifiers detect other features that correlate with the problematic attribute, learn the pattern and make a similar estimation. In the paper[1], the income classifier showed a similar result even without the sex attribute. The classifier used sex-correlated other features and discriminated indirectly. In conclusion, removing possible-discriminating features cannot solve the unfairness of machine learning completely.

Reference

[1] Toon Calders · Sicco Verwer.

Three naive Bayes approaches for discrimination-free classification.

[Online]: <https://link.springer.com/content/pdf/10.1007/s10618-010-0190-x.pdf>