CS699 A1 – Spring 2025
Homework 7

Due: 3/31

For Problem 1 and Problem 2, you must use the method that we discussed in the class.

Problem 1 (10 points). This question is about the AdaBoost algorithm that we discussed in the class.

The following table shows a test result for one model:

TID	Weight	Act	Pred
1	0.1	P	P
2	0.1	N	P
3	0.1	N	N
4	0.1	N	N
5	0.1	P	P
6	0.1	N	P
7	0.1	P	P
8	0.1	P	P
9	0.1	N	P
10	0.1	P	P

(1). Calculate the error of this model.

- The error is 0.3 because 3 out of the 10 are wrong with a weight of 0.1. Added all together the error is 0.3.

(2). Calculate the normalized, updated weights.

- To update the weights, we use the AdaBoost formula. For correctly classified instances, the new weight multiplier is 1 divided by 2 times 1 minus 0.3, which equals 0.714, and for misclassified ones, it is 1 divided by 2 times 0.3, which equals 1.667. Applying these multipliers, the updated weights for the correctly classified instances become 0.0714, while the misclassified ones get bumped up to 0.1667. The total sum still equals 1, so everything checks out. So, in the end, the correctly classified instances now have a weight of 0.0714, and the misclassified ones have 0.1667.

Problem 2 (10 points). This problem is also about the AdaBoost algorithm that we discussed in the class.

Suppose that the algorithm built 5 models and each of the 5 models classified an unseen object. The following table shows the errors of the five models and their predictions.

Model	err(Mi)	Pred
1	0.17	N

2	0.05	P
3	0.13	N
4	0.2	N
5	0.08	P

Determine the final classification (the classification of the composite model).

To determine the final classification of the composite AdaBoost model, we use a weighted majority vote, where each model's prediction gets a weight based on its accuracy. The weight, called the alpha value, is calculated using the error rate of each model with the formula:

Alpha i equals one divided by two times the natural logarithm of one minus the error of model i divided by the error of model i.

Here, Alpha i is the weight assigned to model i, and the error of model i is its error rate. A lower error means a higher alpha value, so more accurate models have a bigger influence on the final decision. After calculating the alpha values for each model, we see that Model 1 has a weight of 0.793, Model 2 has 1.472, Model 3 has 0.950, Model 4 has 0.693, and Model 5 has 1.221. Each model adds its weight to its predicted class, so we add up the alpha values for the models predicting N and those predicting P. The total weight for models predicting N is 2.436, while for those predicting P, it’s 2.693. Since the total weight for P is higher than for N, the final classification of the composite model is P, meaning the ensemble model predicts the object belongs to the positive class. This shows how AdaBoost strengthens the overall decision by giving more power to the accurate models while downplaying the weaker ones.

Problem 3 (10 points). This problem is about the A/B test that we discussed in the class.

A bank is sending out a promotion to potential customers encouraging them to open a new checking account. The promotion offers an opening bonus to customers if they open a new account within 3 months and they want to compare two bonus amounts, $100 and $200. More specifically, they want to know whether offering $200 would attract more new customers than offering $100. They randomly selected 600 customers and randomly divided them into two groups. The first group, Group A, was offered $100 and the second group, Group B, was offered $200. After three months, they counted the number of customers who opened a new account in each group and prepared the following table:

	Group A ($100 bonus)	Group B ($200 bonus)
Number of customers who opened a new account	108	90
Number of customers who did not open a new account	192	210
Total	300	300

Conduct the A/B test using the method that we discussed in the class and determine whether offering $200 would attract more new customers or not. You must answer this problem yourself without using any data mining or data analysis software which can do A/B test automatically.

To conduct the A/B test for the bank's promotion, we start by defining our hypotheses. The null hypothesis states that there is no difference in the proportion of customers who opened an account between the two groups, while the alternative hypothesis suggests that the proportion of customers who opened an account in Group B, which got a 200 dollar bonus, is greater than in Group A, which got a 100 dollar bonus. From the data provided, we see that in Group A, 108 customers opened an account out of 300, giving us a proportion of 0.36. In Group B, 90 customers opened an account out of 300, resulting in a proportion of 0.30.

Next, we calculate the pooled proportion, which comes out to 0.33, and the standard error, which is about 0.0384. Using these values, we find the z-statistic, which is around 1.563. For a one-tailed test at a significance level of 0.05, the critical z-value is about 1.645. When we compare our z-statistic to the critical value, we see that 1.563 is less than 1.645, so we fail to reject the null hypothesis.

In conclusion, there’s not enough evidence to support the idea that offering a 200 dollar bonus would attract more new customers compared to the 100 dollar bonus. So, it looks like the 100 dollar bonus is actually more effective in bringing in new customers in this case.

Problem 4 (20 points). Consider the following transactional database.

TID	Items
100	2, 4, 5, 6
200	1, 4, 5, 7
300	2, 4, 5
400	1, 2, 4, 5, 6, 7
500	1, 2, 6

(1) Mine all frequent itemsets using the Apriori algorithm that we discussed in the class. Show all candidate itemsets and frequent itemsets. You should follow the step by step process

that we discussed in the class (i.e., C1 → L1 → C2 → L2 → …). You don't need to show the pruning steps. Minimum support = 30% (or 2 or more transactions). To save your time, L1 is given below:

L1:

Itemset	1	2	4	5	6	7
Count	3	4	4	4	3	2

- L1 (1-itemsets): {1} appears 3 times, {2} appears 4 times, {4} appears 4 times, {5} appears 4 times, {6} appears 3 times, {7} appears 2 times.

- L2 (2-itemsets): {1, 2} appears 2 times, {1, 4} appears 2 times, {1, 5} appears 2 times, {1, 6} appears 3 times, {1, 7} appears 2 times, {2, 4} appears 3 times, {2, 5} appears 4 times, {2, 6} appears 3 times, {2, 7} appears 2 times, {4, 5} appears 4 times, {4, 6} appears 2 times, {4, 7} appears 2 times, {5, 6} appears 2 times, {5, 7} appears 2 times, {6, 7} appears 2 times.

- L3 (3-itemsets): {1, 4, 5} appears 2 times, {1, 4, 6} appears 2 times, {1, 5, 6} appears 2 times, {2, 4, 5} appears 4 times, {2, 4, 6} appears 2 times, {2, 5, 6} appears 2 times, {4, 5, 6} appears 2 times.

- L4 (4-itemsets): {1, 4, 5, 6} appears 2 times, {2, 4, 5, 6} appears 2 times.

This completes the frequent itemset mining using the Apriori algorithm.

(2) Sort all frequent 4-itemsets by their item number. Then, select the first frequent 4-itemset from the sorted list of frequent 4-itemsets and mine all strong rules from this itemset that have the format {W, X} => {Y, Z}, where W, X, Y, and Z are individual items. Assume that minimum confidence = 80%.

To mine strong association rules from the frequent 4-itemset {1, 4, 5, 6}, we first sort the frequent 4-itemsets by their item numbers. The sorted list shows {1, 4, 5, 6} as the first itemset and {2, 4, 5, 6} as the second. We then focus on the first frequent 4-itemset, which is {1, 4, 5, 6}.
From this itemset, we generate all possible combinations of items W, X, Y, Z that can form rules like {W, X} => {Y, Z}. The potential rules include {1, 4} => {5, 6}, {1, 5} => {4, 6}, {1, 6} => {4, 5}, {4, 5} => {1, 6}, {4, 6} => {1, 5}, and {5, 6} => {1, 4}. To figure out how strong these rules are, we calculate their confidence, which is just the support of the combined itemset divided by the support of the antecedent.
For the rule {1, 4} => {5, 6}, the confidence comes out to 100 percent since it appears in the same transactions as the antecedent. We get similar results for the other rules, with confidences of 100 percent for {1, 5} => {4, 6}, {1, 6} => {4, 5}, and {4, 6} => {1, 5}. The rules {4, 5} => {1, 6} and {5, 6} => {1, 4} also have a confidence of 100 percent. Since we set a minimum confidence threshold of 80 percent, all these rules are considered strong.
In summary, the strong association rules mined from the itemset {1, 4, 5, 6} are {1, 4} => {5, 6}, {1, 5} => {4, 6}, {1, 6} => {4, 5}, {4, 6} => {1, 5}, and {5, 6} => {1, 4}, all showing a solid relationship between the items.

Submission:

Include all answers in a single file and name it *LastName_FirstName_HW6.pdf* or *LastName_FirstName_HW6.*docx, and submit it to Blackboard.