Karthik Bangalore Mani
A20344597

Phase II Report

Description:

1. **Train:**
   To classify a given product, I have used Naïve Bayes Classifier. According to Naïve Bayes classifier, the probability of a product belonging to a class Ck, given features x1, x2….xn is given by the formula:

$$p(C_k|x_1,\ldots,x_n) = \frac{1}{Z}p(C_k)\prod_{i=1}^{n} p(x_i|C_k)$$

   Where 1/Z is a constant scaling factor dependent only on x1, x2…xn.
   Citation: https://en.wikipedia.org/wiki/Naive_Bayes_classifier

   As seen in the RHS of above formula, to find the probability of class (the LHS), given features, we need to compute 3 things:

   a. **1/Z :**
      After trial and error, I found a good scaling factor, which is equal to the no of the products present in the X matrix of the train function, which is equal to:
      2 ^ (no of features).

      Example: If there are 5 features. The scaling factor will be 2^5 = 32.

      I am storing this scaling factor in an instance variable called self.scalingFactor

   b. **P(Ck)**
      P(Ck) is nothing but the Probability of Excellent products, P(Excellent).

      Example: If there are 1000 products, out which 500 are Excellent, then
      P(Excellent) = 500/ 1000 = 0.5

      I am storing this P(Excellent) in an instance variable called self.probE

   c. **P(Xi | Ck)**
      This is the probability of a single feature Xi, given its class Ck. For our project, Ck will be Excellent, and feature Xi will be either True or False.

      To store these probabilities of each individual features given that the product is Excellent, I have created 2 lists:

      - **trueProbList**: The probabilities of all True features, given that the product is Excellent is stored here. For n features, there will be n entries in trueProbList.

      - **falseProbList**: The probabilities of all False features, given that the product is Excellent is stored here. For n features, there will be n entries in falseProbList.

Example : If there is a feature matrix X and Class vector y, with below values :

X

feature 1	feature 2	feature 3
T	T	T
T	T	F
F	T	T
F	F	F

Y

Class
Excellent
Excellent
Trash
Excellent

Then, the lists will have following probabilities:

	feature 1	feature 2	feature 3
trueProbList	2/3	2/3	1/3
falseProbList	1/3	1/3	2/3

There are **3** Excellent products in class Y, out of which **2** of them have True under feature 1 column of X matrix. Hence, the Probability of feature 1 being True given the product is excellent is **2/3.** Similarly the other remaining 5 true and false probabilities of all 3 features are computed and stored in trueProbList and falseProbList respectively.

2. **Predict_prob_of_excellent:**
Let us assume, we have been given an x vector with the following feature values:
[True, True, False]

So, we have to compute following the probability :
P(Feature = true | Excellent) * P(Feature = true | Excellent) * P(Feature = false | Excellent)

Which is equal to:

P = (2/3) * (2/3) * (2/3).

And finally, I will compute the product of (Probability(Excellent) * P * scalingFactor) and store it in a variable called finalProb.

Because of the kind of scalingFactor I have chosen, I observed that my agent overestimates probabilities. In other words, sometimes, finalProb happens to be more than 1. In that case, I return 1. Else I will return finalProb.

Simulation Results:

SIMULATION RESULTS ON dataset1

Karthik Bangalore Mani
A20344597

Wealth (the larger the better)

Agent_fixed_prob_0.00:	$0.00
Agent_fixed_prob_0.25:	$806,000.00
Agent_fixed_prob_0.50:	$1,265,000.00
Agent_fixed_prob_0.75:	$1,071,000.00
Agent_fixed_prob_1.00:	$30,000.00
Agent_kbangal2:	$1,887,500.00

Prediction Log-loss (the smaller the better)

Agent_fixed_prob_0.00:	11,582.00
Agent_fixed_prob_0.25:	840.28
Agent_fixed_prob_0.50:	693.15
Agent_fixed_prob_0.75:	833.69
Agent_fixed_prob_1.00:	11,443.85
Agent_kbangal2:	744.15

Prediction Error (the smaller the better)

Agent_fixed_prob_0.00:	503
Agent_fixed_prob_0.25:	503
Agent_fixed_prob_0.50:	497
Agent_fixed_prob_0.75:	497
Agent_fixed_prob_1.00:	497
Agent_kbangal2:	173

SIMULATION RESULTS ON dataset2

Wealth (the larger the better)

Agent_fixed_prob_0.00:	$0.00
Agent_fixed_prob_0.25:	$318,000.00
Agent_fixed_prob_0.50:	$45,000.00
Agent_fixed_prob_0.75:	$-637,000.00
Agent_fixed_prob_1.00:	$-2,410,000.00
Agent_kbangal2:	$737,150.00

Prediction Log-loss (the smaller the better)

Agent_fixed_prob_0.00:	5,963.70
Agent_fixed_prob_0.25:	572.22
Agent_fixed_prob_0.50:	693.15
Agent_fixed_prob_0.75:	1,101.75
Agent_fixed_prob_1.00:	17,062.16
Agent_kbangal2:	635.28

Prediction Error (the smaller the better)

Karthik Bangalore Mani
A20344597

Agent_fixed_prob_0.00: 259
Agent_fixed_prob_0.25: 259
Agent_fixed_prob_0.50: 741
Agent_fixed_prob_0.75: 741
Agent_fixed_prob_1.00: 741
Agent_kbangal2: 160

SIMULATION RESULTS ON dataset3

Wealth (the larger the better)

Agent_fixed_prob_0.00: $0.00
Agent_fixed_prob_0.25: $1,278,000.00
Agent_fixed_prob_0.50: $2,445,000.00
Agent_fixed_prob_0.75: $2,723,000.00
Agent_fixed_prob_1.00: $2,390,000.00
Agent_kbangal2: $3,181,450.00

Prediction Log-loss (the smaller the better)

Agent_fixed_prob_0.00: 17,016.10
Agent_fixed_prob_0.25: 1,099.56
Agent_fixed_prob_0.50: 693.15
Agent_fixed_prob_0.75: 574.42
Agent_fixed_prob_1.00: 6,009.75
Agent_kbangal2: 889.09

Prediction Error (the smaller the better)

Agent_fixed_prob_0.00: 739
Agent_fixed_prob_0.25: 739
Agent_fixed_prob_0.50: 261
Agent_fixed_prob_0.75: 261
Agent_fixed_prob_1.00: 261
Agent_kbangal2: 133

SIMULATION RESULTS ON dataset4

Wealth (the larger the better)

Agent_fixed_prob_0.00: $0.00
Agent_fixed_prob_0.25: $748,000.00
Agent_fixed_prob_0.50: $1,120,000.00
Agent_fixed_prob_0.75: $868,000.00
Agent_fixed_prob_1.00: $-260,000.00
Agent_kbangal2: $1,514,550.00

Karthik Bangalore Mani
A20344597

Prediction Log-loss (the smaller the better)

Agent_fixed_prob_0.00:	10,914.25
Agent_fixed_prob_0.25:	808.42
Agent_fixed_prob_0.50:	693.15
Agent_fixed_prob_0.75:	865.55
Agent_fixed_prob_1.00:	12,111.60
Agent_kbangal2:	2,081.92

Prediction Error (the smaller the better)

Agent_fixed_prob_0.00:	474
Agent_fixed_prob_0.25:	474
Agent_fixed_prob_0.50:	526
Agent_fixed_prob_0.75:	526
Agent_fixed_prob_1.00:	526
Agent_kbangal2:	214