Devin Wu <- replace with your name

**CS 585 Fall 2024 Programming Assignment #03**

Due: **Sunday, November 24, 2024 at 11:59 PM CST**

Points: **200**

**Instructions:**

1. Place **all your deliverables (as described below) into a single ZIP** file named:

LastName\_FirstName\_CS585\_Programming03.zip

1. Submit it to Blackboard Assignments section before the due date **[presentation slides can be added AFTER you presented]**. **No late submissions will be accepted**.

**Objectives:**

1. (100 points) Implement and evaluate a Naïve Bayes classifier algorithm.
2. (100 points) Implement and evaluate a logistic regression classifier algorithm.

**Task:**

Your task is to implement, train, and test two classification algorithms (Naïve Bayes and Logistic Regression) using a publicly available data set.

**Data set:**

Pick a publicly available data (**I provided a list and instructions**) set first and do an initial exploratory data analysis.

**Deliverables:**

Your submission should include:

* Python code file(s). Your py file should be named:

cs585\_P03\_AXXXXXXXX.py

where AXXXXXXXX is your IIT A number (this is REQUIRED!). If your solution uses multiple files, makes sure that the main (the one that will be run to solve the problem) is named that way and others include your IIT A number in their names as well.

* Presentation slides in PPTX or PDF format. **[This will be counted as your Written Assignment].** Name it:

LastName\_FirstName\_CS585\_P03\_Slides.pptx or pdf

* This document with your observations and conclusions. You should rename it to:

LastName\_FirstName\_CS585\_P03.pdf

**Implementation:**

Your task is to implement (**one [Naive Bayes] from SCRATCH, logistic regression: you can use scikit-learn package [[LogisticRegression — scikit-learn 1.5.2 documentation](https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LogisticRegression.html)] | you CAN’T use out-of-the-box Python document vectorization**), train, and test a both classifiers (as outlined in class) and apply them to classify sentences entered using keyboard.

Your program should:

* Accept one (1) command line argument, i.e. so your code could be executed with

python cs585\_P03\_AXXXXXXXX.py ALGO TRAIN\_SIZE

where:

* + cs585\_P03\_AXXXXXXXX.py is your python code file name,
  + ALGO is the algorithm type (0 for Naïve Bayes and 1 for Logistic regression)
  + TRAIN\_SIZE is a number between 50 and 80 defining the size (in percentages) of the training set. For example: 60 would mean **FIRST** (as ordered in the dataset file) 60% of samples. **Note that your test set is always going to be the LAST (as ordered in the dataset file) 20% of samples.**

Example:

python cs585\_P03\_A11111111.py 0 85

If the number of arguments provided is NOT two (none, one, three or more) or the TRAIN\_SIZE argument is out of the specified range (50 to 80 inclusive), assume that the value for TRAIN\_SIZE is 80. Similarly, default ALGO value: 0.

* Load and process input data set:
  + Apply any data clean-up / wrangling you consider necessary first (mention and discuss your choices in the Conclusions section below).
  + Text pre-processing:
    - treat every document in the data set as a single sentence, even if it is made of many (no segmentation needed),
* Train your classifier on your data set:
  + assume that vocabulary V is the set of ALL words in the data set,
  + divide your data set into:
    - training set: FIRST (as they appear in the data set) TRAIN\_SIZE % of samples / documents,
    - test set: LAST 20 % of samples / documents,
  + use **Non-Binary Bag of Words with add-1 smoothing** vector representation for documents,
  + train your classifier (find its parameters. Naïve Bayes HINT: use Python dictionary to store them).
* Test your classifier:
  + use the test set to test your classifier,
  + calculate (and display on screen) following metrics:
    - number of true positives,
    - number of true negatives,
    - number of false positives,
    - number of false negatives,
    - sensitivity (recall),
    - specificity,
    - precision,
    - negative predictive value,
    - accuracy,
    - F-score,
* Ask the user for keyboard input (a single sentence S):
  + use your classifier to decide (Naïve Bayes HINT: use log-space calculations to avoid underflow – but bring it back to linear space after!) which class S belongs to,
  + display classifier decision along with P(CLASS\_A |S) and P(CLASS\_B | S) values on screen

Your program output should look like this:

Last Name, First Name, AXXXXXXXX solution:

Training set size: 80 %

Classifier type: TYPE

Training classifier…

Testing classifier…

Test results / metrics:

Number of true positives: xxxx

Number of true negatives: xxxx

Number of false positives: xxxx

Number of false negatives: xxxx

Sensitivity (recall): xxxx

Specificity: xxxx

Precision: xxxx

Negative predictive value: xxxx

Accuracy: xxxx

F-score: xxxx

Enter your sentence/document:

Sentence/document S: <entered sentence here>

was classified as <CLASS\_LABEL here>.

P(<CLASS\_A> | S) = xxxx

P(<CLASS\_B> | S) = xxxx

Do you want to enter another sentence [Y/N]?

If user responds Y, classify new sentence (you should not be re-training your classifier).

where:

* 80 would be replaced by the value specified by TRAIN\_SIZE,
* TYPE is the classifier selected by the ALGO parameter,
* xxxx is an actual numerical result,
* <entered sentence here> is actual sentence entered y the user,
* <CLASS\_LABEL here> is the class label decided by your classifier,
* <CLASS\_A>, <CLASS\_B> are available labels (SPAM/HAM, POSITIVE/NEGATIVE, etc.).
* This part:

P(<CLASS\_A> | S) = xxxx

P(<CLASS\_B> | S) = xxxx

is only for Naïve Bayes classifer

**Classifier testing results:**

Enter your classifier performance metrics below:

|  |  |
| --- | --- |
| Naïve Bayes / TRAIN\_SIZE set to 80: | Logistic Regression/ TRAIN\_SIZE set to 80: |
| Number of true positives: 1879  Number of true negatives: 1927  Number of false positives: 164  Number of false negatives: 15  Sensitivity (recall): 0.9921  Specificity: 0.9216  Precision: 0.9197  Negative predictive value: 0.9923  Accuracy: 0.9551  F-score: 0.9545 | Number of true positives: 1357  Number of true negatives: 1208  Number of false positives: 883  Number of false negatives: 537  Sensitivity (recall): 0.7164  Specificity: 0.5777  Precision: 0.6058  Negative predictive value: 0.6922  Accuracy: 0.6436  F-score: 0.6565 |

What are your observations and conclusions? When did the algorithm perform better? a summary below

|  |
| --- |
| **Summary / observations / conclusions** |
| It looks to me that naïve bayes works way better than logistic regression here. Not sure why logistic regression performed so poorly. Naïve bayes basically performed better in every case. |