CS464 Introduction to Machine Learning
Spring 2021
Homework 1

Due: March 12, 2021, 5:00 PM

Instructions

- Submit a soft copy of your homework of all questions to Moodle. Add your code at the end of the your homework file and upload it to the related assignment section on Moodle. Submitting a hard copy or scanned files is NOT allowed. You have to prepare your homework digitally(using Word, Excel, Latex etc.).
- This is an individual assignment for each student. That is, you are NOT allowed to share your work with your classmates.
- For this homework, you may code in any programming language you would prefer. In submitting the homework file, please package your file as a gzipped TAR file or a ZIP file with the name `CS464_HW1_Section#_Firstname_Lastname`.

  As an example, if your name is Sheldon Cooper and you are from Section 1 for instance, then you should submit a file with name `CS464_HW1_1_sheldon_cooper`. Do NOT use Turkish letters in your package name.

  Your compressed file should include the following:

  - report.pdf : The report file where you have written your calculations, plots, discussions and other related work.
  - q3main.*: The main code file of your work. It should be in a format easy to run and must include a main script serving as an entry point. The extension of this file depends on the programming language of your choice. For instance, if you are using Python, your code file should end with ".py" extension. If you are using a notebook editor, do not forget to save your file as a Python file at the end. If you are using MATLAB, your file should end with extension ".m". For other programming languages, your file should have the extension of the main executable file format for that language.
  - README.txt : You must also provide us with a README file that tells us how we can execute/call your program. README file should include which parameters are the default values, what is the terminal command to execute your file and how to read the outputs.
- You are NOT allowed to use any machine learning packages, libraries or toolboxes for this assignment (such as scikit-learn, tensorflow, keras, theano, MATLAB Statistics and Machine Learning Toolbox functions, e1071, nnet, kernlab etc.).
- Your codes will be evaluated in terms of efficiency as well. Make sure you do not have unnecessary loops and obvious inefficient calculations in your code.
- If you do not follow the submission routes, deadlines and specifications (codes, report, etc), it will lead to significant grade deduction.
- For any questions regarding this homework, contact to salman.mohammad@bilkent.edu.tr.

1 The White Library [17 pts]

Donald is interested in the number of words in the titles of books and their types as novels, poetry,stories; he aims to form a probabilistic model. He finds that the minimum number of words in the titles of all these types is 1 and the maxima are 2,3 and 3, for novels, poetry, stories, respectively.

Question 1.1 [3 pts] Define the sample space of Donald's experiment.

Question 1.2 [3 pts] Based on your sample space, write the elements of the event
$A = \{$"a novel with a single-word title or a poem book with a 2- or 3-word title but not a story book"$\}$.

Question 1.3 [3 pts] Write down the axioms of probability.

Question 1.4 [8 pts] After collecting some statistics, Donald estimates that, in a random book selection experiment,
$P(\{$a 3-word title story book or a 2-word title novel$\}) = 0.045$
$P(\{$a 3-word tile story book or a 2-word title novel or 2-word title poetry book$\}) = 0.11$
$P(\{$a 2-word title poetry book or a 3-word title story book$\}) = 0.06$
Would you agree Donald on his estimation? Why?

2 Cafe Customers [18 pts]

The total number of people X arriving to Mozart Cafe in a 10 minute-break can be modeled as a Poisson random variable with mean 20, i.e. the probability of k people will arrive to the cafe can be expressed as in Eqn.2.1. Suppose that each person has a probability 0.3 of buying a cup of coffee, independent of the others. Let Y be the number of people that buy coffee from the cafe in a 10-minute break. For this question, assume X and Y are independent.

$$P(X = k) = e^{-20} \times \frac{20^k}{k!} \tag{2.1}$$

Question 2.1 [6 pts] If 10 people arrive to the cafe in a break, what is the probability that less than 3 people will buy coffee?

Question 2.2 [6 pts] What is the probability that a total of 2 people arrive to the cafe in a break and none of them buys coffee?

Question 2.3 [6 pts] Compute the expected value of Y, i.e. $E[Y]$ without computing the probability mass function of Y.

3 Spam Email Detection [65 pts]

As a computer scientist working for an email service provider, your job is develop a model to filter spam emails.

Dataset

Your dataset is a preprocessed and modified version of Spam Mails Dataset [1]. It is based on 5171 real emails which are either spam or not. Emails have been preprocessed in the following ways:

- **Stop word removal:** Words like "and","the", and "of", are very common in all English sentences and are therefore not very predictive. These words have been removed from the emails.

- **Removal of non-words:** Numbers and punctuation have both been removed. All white spaces (tabs, newlines, spaces) have all been trimmed to a single space character
- **Removal of infrequent words:** Words that occur only once in all data set are removed from emails in order to reduce the size of the data.

The data has been already split into two subsets: a 4085-email subset for training and a 1086-email subset for testing (consider this as your validation set and imagine there is another test set which is not given to you). Features have been generated for you. You will use the following files:

- `x_train.csv`
- y_train.csv
- `x_test.csv`
- `y_test.csv`
- `vocabulary.txt`

The files that start with `x` contain the features and the files starting with `y` contain the ground truth labels.

In the feature files each row contains the feature vector for an email. The j-th term in a row i is the occurrence information of the j-th vocabulary word in the i-th email. The size of the vocabulary is 44020. The label files include the ground truth label for the corresponding email (label 0 is normal mail, and label 1 is spam), the order of the emails (rows) are the same as the features file. That is the i-th row in the files corresponds to the same email. Each email is labeled as either `spam` or `not-spam(normal mail)`.

The file ending with `vocabulary.txt` is the vocabulary file in which the j-th word (feature) in the file corresponds to the j-th feature in both train and test sets.

Bag-of-Words Representation and Multinomial Naive Bayes Model

Recall the bag-of-words document representation makes the assumption that the probability that a word appears in an e-mail is conditionally independent of the word position given the class of the e-mail. If we have a particular e-mail document D_i with n_i words in it, we can compute the probability that D_i comes from the class y_k as:

$$\mathbf{P}\left(D_i \,|\, Y = y_k\right) = \mathbf{P}\left(X_1 = x_1, X_2 = x_2, .., X_{n_i} = x_{n_i} \,|\, Y = y_k\right) = \prod_{j=1}^{n_i} \mathbf{P}\left(X_j = x_j \,|\, Y = y_k\right) \tag{3.1}$$

In Eq. (3.1), X_j represents the j^{th} position in the e-mail D_i and x_j represents the actual word that appears in the j^{th} position in the e-mail, whereas n_i represents the number of positions in the e-mail. As a concrete example, we might have the first e-mail (D_1) which contains 200 words ($n_1 = 200$). The document might be of positive e-mail ($y_k = 1$) and the 15th position in the e-mail might have the word "well" ($x_j =$ "well").

In the above formulation, the feature vector $\vec{X}$ has a length that depends on the number of words in the e-mail n_i. That means that the feature vector for each e-mail will be of different sizes. Also, the above formal definition of a feature vector $\vec{x}$ for an e-mail says that $x_j = k$ if the j-th word in this e-mail is the k-th word in the dictionary. This does not exactly match our feature files, where the j-th term in a row i is the number of occurrences of the j-th dictionary word in that e-mail i. As shown in the lecture slides, we can slightly change the representation, which makes it easier to implement:

$$\mathbf{P}\left(D_i \,|\, Y = y_k\right) = \prod_{j=1}^{V} \mathbf{P}\left(X_j \,|\, Y = y_k\right)^{t_{w_j,i}} \tag{3.2}$$

,where V is the size of the vocabulary, X_j represents the appearing of the j-th vocabulary word and $t_{w_j,i}$ denotes how many times word w_j appears in an e-mail D_i. As a concrete example, we might have a vocabulary of size of 1309, $V = 1309$. The first e-mail (D_1) might be a spam e-mail ($y_k = 1$) and the 80-th word in the vocabulary, w_{80}, is "trade" and $t_{w_{80},1} = 2$, which says the word "trade" appears 2 times in the e-mail D_1. Contemplate on why these two models (Eq. (3.1) and Eq. (3.2)) are equivalent.

In the classification problem, we are interested in the probability distribution over the email classes (in this case spam or normal mail) given a particular e-mail D_i. We can use Bayes Rule to write:

$$\mathbf{P}(Y = y_k | D_i) = \frac{\mathbf{P}(Y = y_k) \prod_{j=1}^{V} \mathbf{P}(X_j \,|\, Y = y)^{t_{w_j,i}}}{\sum_k \mathbf{P}(Y = y_k) \prod_{j=1}^{V} \mathbf{P}(X_j \,|\, Y = y_k)^{t_{w_j,i}}} \tag{3.3}$$

Note that, for the purposes of classification, we can actually ignore the denominator here and write:

$$\mathbf{P}(Y = y_k | D_i) \propto \mathbf{P}(Y = y_k) \prod_{j=1}^{V} \mathbf{P}(X_j \,|\, Y = y)^{t_{w_j,i}} \tag{3.4}$$

$$\hat{y}_i = \arg\max_{y_k} \mathbf{P}(Y = y_k \,|\, D_i) = \arg\max_{y_k} \mathbf{P}(Y = y_k) \prod_{j=1}^{V} \mathbf{P}(X_j \,|\, Y = y_k)^{t_{w_j,i}} \tag{3.5}$$

Probabilities are floating point numbers between 0 and 1, so when you are programming it is usually not a good idea to use actual probability values as this might cause numerical underflow issues. As the logarithm is a strictly monotonic function on [0,1] and all of the inputs are probabilities that must lie in [0,1], it does not have an affect on which of the classes achieves a maximum. Taking the logarithm gives us:

$$\hat{y}_i = \arg\max_{y} \left(\log \mathbf{P}(Y = y_k) + \sum_{j=1}^{V} t_{w_j,i} * \log \mathbf{P}(X_j \,|\, Y = y_k) \right) \tag{3.6}$$

, where $\hat{y}_i$ is the predicted label for the i-th example.

Question 3.1 [5 points] If the the ratio of the classes in a dataset is close to each other, it is a called "balanced" class distribution; i.e it is not skewed. What is the percentage of spam e-mails in the `y_train.csv`?.

Is the training set balanced or skewed towards one of the classes? Do you think having an imbalanced training set affects your model? If yes, please explain how it affects and propose a possible solution if needed.

The parameters to learn and their MLE estimators are as follows:

$$\theta_{j\,|\,y=spam} \equiv \frac{T_{j,y=spam}}{\sum_{j=1}^{V} T_{j,y=spam}}$$
$$\theta_{j\,|\,y=normal} \equiv \frac{T_{j,y=normal}}{\sum_{j=1}^{V} T_{j,y=normal}}$$
$$\pi_{y=normal} \equiv \mathbf{P}(Y = normal) = \frac{N_{normal}}{N}$$

- $T_{j,spam}$ is the number of occurrences of the word j in spam e-mails in the training set including the multiple occurrences of the word in a single tweet.
- $T_{j,normal}$ is the number of occurrences of the word j in normal e-mails in the training set including the multiple occurrences of the word in a single e-mail.
- N_{normal} is the number of positive tweets in the training set.
- N is the total number of tweets in the training set.
- $\pi_{y=normal}$ estimates the probability that any particular tweet will be positive.
- $\theta_{j\,|\,y=spam}$ estimates the probability that a particular word in a spam e-mail will be the j-th word of the vocabulary, $\mathbf{P}(X_j \,|\, Y = spam)$
- $\theta_{j\,|\,y=normal}$ estimates the probability that a particular word in a normal e-mail will be the j-th word of the vocabulary, $\mathbf{P}(X_j \,|\, Y = positive)$

For all questions after this point, consider your test set as a validation set and assume that there is another test set that is not given to you.

Question 3.2 (Coding*) [20 points] Train a Multinomial Naive Bayes model on the training set and evaluate your model on the test set given. Find and report the accuracy and the confusion matrix for the test set as well as how many wrong predictions were made.

In estimating the model parameters use the above MLE estimator. If it arises in your code, define $\log 0$ as it is, that is -inf. In case of ties, you should predict "0".

Question 3.3 (Coding*) [20 points] Extend your classifier so that it can compute an MAP estimate of θ parameters using a fair Dirichlet prior. This corresponds to additive smoothing. The prior is fair in the sense that it "hallucinates" that each word appears additionally α times in the train set.

$$\theta_{j\,|\,y=spam} \equiv \frac{T_{j,y=spam}+\alpha}{\sum_{j=1}^{V} T_{j,y=spam}+\alpha * V}$$
$$\theta_{j\,|\,y=normal} \equiv \frac{T_{j,y=normal}+\alpha}{\sum_{j=1}^{V} T_{j,y=normal}+\alpha * V}$$
$$\pi_{y=normal} \equiv \mathbf{P}\left(Y = normal\right) = \frac{N_{normal}}{N}$$

For this question set $\alpha = 1$. Train your classifier using all of the training set and have it classify all of the test set and report test-set classification accuracy and the confusion matrix. Explicitly discuss your results.

Bag-of-Words Representation and Bernoulli Naive Bayes Model

Question 3.4 (Coding*) [20 points] Train a Bernoulli Naive Bayes classifier using all of the data in the training set, and report the testing accuracy and the confusion matrix as well as how many wrong predictions were made.

Remember that this time, if the j-th word exist in the i-th e-mail than the related term is set to 1, and 0 otherwise, that is, $t_j = 1$ when the word exists and $t_j = 0$ otherwise. The formula for the estimated class is given in Eq. (3.7). In estimating the model parameters use the below MLE estimator equations. If it arises in your code, define $0 * \log 0 = 0$ (note that $a * \log 0$ is as it is, that is -inf). In case of ties, you should predict "normal". Report your test set accuracy in your written report. What did your classifier end up predicting? Compare your results with the Multinomial Model. Discuss your finding explicitly

$$\hat{y}_i = \arg\max_{y} \left(\log \mathbf{P}\left(Y = y_k\right) + \log\left(\prod_{j=1}^{V} t_j * \mathbf{P}\left(X_j \,|\, Y = y_k\right) + (1 - t_j) * \left(1 - \mathbf{P}\left(X_j \,|\, Y = y_k\right)\right)\right) \right) \tag{3.7}$$

, where $\hat{y}_i$ is the predicted label for the i-th example and t_j indicates whether word j appears in the document.

The parameters to learn and their MLE estimators are as follows:

$$\theta_{j\,|\,y=spam} \equiv \frac{S_{j,y=spam}}{N_{spam}}$$
$$\theta_{j\,|\,y=normal} \equiv \frac{S_{j,y=normal}}{N_{normal}}$$
$$\pi_{y=normal} \equiv \mathbf{P}\left(Y = normal\right) = \frac{N_{normal}}{N}$$

- $S_{j,spam}$ is the number of occurrences of the word j in spam e-mails in the training set NOT including the multiple occurrences of the word in a single tweet.
- $S_{j,normal}$ is the number of occurrences of the word j in normal e-mails in the training set NOT including the multiple occurrences of the word in a single tweet.
- N_{normal} is the number of normal e-mails in the training set.
- N is the total number of e-mails in the training set.
- $\pi_{y=normal}$ estimates the probability that any particular e-mails will be normal.
- $\theta_{j \mid y=spam}$ estimates the fraction of the spam e-mails with j-th word of the vocabulary, $\mathbf{P}\left(X_j \mid Y = spam\right)$
- $\theta_{j \mid y=normal}$ estimates the fraction of the normal e-mails with the j-th word of the vocabulary, $\mathbf{P}\left(X_j \mid Y = normal\right)$

References

1. Spam Mails Dataset https://www.kaggle.com/venky73/spam-mails-dataset
2. "On Discriminative vs. Generative Classifiers: A comparison of logistic regression and Naive Bayes" by Andrew Ng and Michael I. Jordan.
3. Manning, C. D., Raghavan, P., and Schutze, H. (2008). Introduction to information retrieval. New York: Cambridge University Press.
http://nlp.stanford.edu/IR-book/html/htmledition/mutual-information-1.html
4. CMU Lecture Notes.
http://www.cs.cmu.edu/~epxing/Class/10701-10s/Lecture/lecture5.pdf