CS464 Introduction to Machine Learning
Spring 2023
Homework 1

Due: April 02, 2023, 23:59

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 a.yildirim@bilkent.edu.tr.

1 The Online Shopping Case [30 pts]

A student taking CS 464 course at Bilkent University is hired by an online shopping company to infer some probabilities from historical statistics.

The company aims to find the relationship between the number of sales and customer feedback. For this purpose, they categorized the products according to these features. Each product's feedback type is set as positive (F_P) or negative (F_N) based on the customer comments. Besides, they are labeled as popular (P), moderately sold (M), and unpopular (U) regarding their number of sales.

According to the historical data, 95% of the popular products, 60% of the moderately sold products, and 10% of the unpopular products have got positive feedback.

Also, it is observed that 45%, 30%, and 25% of the products have been popular, moderately sold, and unpopular respectively.

The company asks the student the following questions:

Note: It is strongly suggested to use a calculator in the following questions. The results can be rounded to 4 decimal places.

Question 1.1 [10 pts] What is the probability that a product gets positive feedback [$\mathbf{P}(F_P)$]?
Question 1.2 [10 pts] If a new product gets positive feedback, what is the probability that it will be a popular product [$\mathbf{P}(P|F_P)$]?
Question 1.3 [10 pts] If a product does not get positive feedback, what is the probability that it will be a popular product [$\mathbf{P}(P|F_N)$]?

2 Spam Email Detection [70 pts]

As a data scientist working for an email service provider, your task is to develop a model that filters spam emails.

Dataset

For this task, your company provided you a dataset composed of 5172 real emails [1]. Each of these emails are labeled as spam (1) or not spam (0). The dataset is already preprocessed such that each column indicates the number of occurrences of a given word for a given email instance. As a preprocessing stage, we dropped the column indicating instance numbers as they have no use. Additionally, data labels (indicating whether the mail is spam or not spam) are given as a separate file.

The dataset has been split into two subsets: a 4137-email subset for training and 1035-email subset for testing. For this task, treat your test set as a validation set and assume that there is another dataset that is not provided to you as a test set. Your company expects you to report the model that performs best on this given validation set and expects your model to behave similar in the test set that is not provided to you. To prevent any bias occurring from the order of the samples, the train-test split has been performed after shuffling the data.

Using the word frequencies, data files are generated for you. You will use the following files:

- `x_train.csv`
- `y_train.csv`
- `x_test.csv`

- `y_test.csv`

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

In the data files generated for you, each row contains a feature vector specifying the occurrences of vocabulary words. The j^{th} element of the feature vector given in row i indicates the number of occurrences of j^{th} word of the vocabulary in the i^{th} email. There are 3000 words in your vocabulary, representing the most common 3000 words in all emails. For the labels, spam and not spam information is represented by a binary label (label 1 for `spam` mail, label 0 for a `non-spam` mail). In the dataset, there are no missing values both in feature vectors and data labels.

In the `csv` files provided to you, the data is organized in a tabular form. This means that you can access the vocabulary from the index row of files containing the feature vectors (`x_train.csv` and `x_test.csv`). To scan the data files and perform vectorized operations, you can use external libraries (such as Pandas and NumPy libraries for Python). If you do so, specify them in your report and mention to what extent you used them.

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 of D_i being an instance of class y_k as:

$$\mathbf{P}\left(D_i \,|\, Y = y_k\right) = \mathbf{P}\left(X_1 = x_1, X_2 = x_2, .., X_{n_i} = x_{n_i} \,|\, Y = y_k\right) = \prod_{j=1}^{n_i} \mathbf{P}\left(X_j = x_j \,|\, Y = y_k\right) \tag{2.1}$$

In Eq. (2.1), X_j represents the word at 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, let us have two examples of emails (documents):

- The first e-mail (D_1) contains 150 words ($n_1 = 150$). The document might be of positive (spam) e-mail, which corresponds to the class label $y_k = 1$. Additionally, the 17th position in the e-mail might have the word "well" ($x_{17} =$ "well" where $j = 17$).
- The 5^{th} e-mail (D_5) contains 350 words ($n_5 = 350$). This time, the document might be a negative (non-spam) mail, which corresponds to the class label $y_k = 0$. Additionally, the 110th position in the e-mail might have the word "you" ($x_{110} =$ "well" where $j = 110$).

In the above formulation, the length of the feature vector for document i ,$\vec{X}i$, 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 $x_j = c$ if the j-th word in this e-mail is the c-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:

$$\mathbf{P}\left(D_i \,|\, Y = y_k\right) = \prod_{j=1}^{|V|} \mathbf{P}\left(X_j \,|\, Y = y_k\right)^{t_{w_j,i}} \tag{2.2}$$

Here, V is the vocabulary and $|V|$ is the vocabulary size, X_j represents the appearance 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 an example, we might have a vocabulary of size 2300, $|V| = 2300$. The second e-mail might be a spam e-mail ($y_k = 1$). For this document, the word "should" might appear three times in the e-mail and it is the 50^{th} word in the vocabulary ($w_{50} =$ "should"). This means that $t_{w_{50},2} = 3$ where $i = 3$ for the third e-mail. Contemplate on why these two models (Eq. (2.1) and Eq. (2.2)) 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:

$$\mathbf{P}\left(Y = y_k | D_i\right) = \frac{\mathbf{P}\left(Y = y_k\right) \prod_{j=1}^{|V|} \mathbf{P}\left(X_j \mid Y = y\right)^{t_{w_j,i}}}{\sum_k \mathbf{P}\left(Y = y_k\right) \prod_{j=1}^{|V|} \mathbf{P}\left(X_j \mid Y = y_k\right)^{t_{w_j,i}}} \tag{2.3}$$

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

$$\mathbf{P}\left(Y = y_k | D_i\right) \propto \mathbf{P}\left(Y = y_k\right) \prod_{j=1}^{|V|} \mathbf{P}\left(X_j \mid Y = y\right)^{t_{w_j,i}} \tag{2.4}$$

$$\hat{y}_i = \underset{y_k}{\arg\max}\, \mathbf{P}\left(Y = y_k \mid D_i\right) = \underset{y_k}{\arg\max}\, \mathbf{P}\left(Y = y_k\right) \prod_{j=1}^{V} \mathbf{P}\left(X_j \mid Y = y_k\right)^{t_{w_j,i}} \tag{2.5}$$

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], using the logarithm of the probability values instead of the actual probability does not change the decision of which class gives a better score for the given document D_i. Taking the logarithm gives us:

$$\hat{y}_i = \underset{y}{\arg\max} \left(\log \mathbf{P}\left(Y = y_k\right) + \sum_{j=1}^{|V|} t_{w_j,i} * \log \mathbf{P}\left(X_j \mid Y = y_k\right) \right) \tag{2.6}$$

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

Question 3.1 [8 points] If the ratio of the classes in a dataset is close to each other, it is called "balanced" class distribution; i.e it is not skewed. Regarding the class imbalance problem, answer the following questions:

1. What is the percentage of spam e-mails in the `y_train.csv`?
2. 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 can affect the model briefly.
3. If your dataset is skewed towards one of the classes, how does this affect your reported accuracy? If yes, to what extent the reported accuracy is misleading in such an unbalanced dataset?

We provided you the estimators for the parameters of the Multinomial Naive Bayes Model as follows. Here, the positive samples are labeled as spam and negative non-spam examples are labeled as normal:

$$\theta_{j \mid y=spam} \equiv \frac{T_{j,y=spam}}{\sum_{j=1}^{V} T_{j,y=spam}}$$

$$\theta_{j \mid y=normal} \equiv \frac{T_{j,y=normal}}{\sum_{j=1}^{V} T_{j,y=normal}}$$

$$\pi_{y=normal} \equiv \mathbf{P}\left(Y = normal\right) = \frac{N_{normal}}{N}$$

- $T_{j,spam}$ 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.

- $\theta_{j \mid y=spam}$ estimates the probability that a particular word in a spam e-mail will be the j-th word of the vocabulary, $\mathbf{P}\left(X_j \mid Y = spam\right)$
- $\theta_{j \mid y=normal}$ estimates the probability that a particular word in a normal e-mail will be the j-th word of the vocabulary, $\mathbf{P}\left(X_j \mid Y = positive\right)$

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. You will assess your models depending on how it performs on the validation set.

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 estimator functions. If it arises in your code, define $\log 0$ as it is, that is -inf. In case of ties, you should predict "0".
Hint: To simulate the behavior of the number '-inf', you can assign an arbitrarily small number to this value (like -10^{12}), to handle overflow issues. If you make such an assumption, indicate it in your report.

Question 3.3 (Coding*) [16 points] Extend your classifier so that it can compute an 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.

$$\theta_{j \mid y=spam} \equiv \frac{T_{j,y=spam}+\alpha}{\sum_{j=1}^{V} T_{j,y=spam+\alpha * V}}$$

$$\theta_{j \mid y=normal} \equiv \frac{T_{j,y=normal}+\alpha}{\sum_{j=1}^{V} T_{j,y=normal}+\alpha * V}$$

$$\pi_{y=normal} \equiv \mathbf{P}\left(Y = normal\right) = \frac{N_{normal}}{N}$$

For this question set $\alpha = 5$. 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 and interpret on the effect of the Dirichlet prior α.

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 exists in the i-th e-mail then the related term is set to 1, and 0 otherwise, that is, $t_j = 1$ when the word exists and $t_j = 0$ otherwise (a binary attribute, rather than the frequency of the given word in vocabulary). The formula for the estimated class is given in Eq. (2.7). In estimating the model parameters use the estimator equations, that are provided below. If it arises in your code, define $0 * \log 0 = 0$ (note that $a * \log 0$ is as it is, that is -inf), again you can use an arbitrarily small number instead of the value '-inf'. In the case of ties, you should predict "normal" (label 0). Report your test set accuracy in your written report. What did your classifier end up predicting? Compare your results with the Multinomial Model. How does Bernoulli Naive Bayes model differ from Multinomial Naive Bayes? Discuss your findings and observations explicitly.

$$\hat{y}_i = \arg\max_{y} \left(\log \mathbf{P}\left(Y = y_k\right) + \log\Big(\prod_{j=1}^{V} t_j * \mathbf{P}\left(X_j \mid Y = y_k\right) + (1 - t_j) * (1 - \mathbf{P}\left(X_j \mid Y = y_k\right))\Big) \right) \tag{2.7}$$

, 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 estimator functions are as follows:

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

- $S_{j,spam}$ 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-mail 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, $\mathbf{P}\left(X_j \mid Y = normal\right)$

Question 3.5 [6 points] Using the confusion matrices that you obtained together with the accuracy values, make a brief comparison regarding your results. How does your algorithm behave for different classes? For a real-world application, which algorithm is better and why? Is accuracy deceiving as a performance metric for this task?

References

1. Spam Mails Dataset https://www.kaggle.com/balaka18/email-spam-classification-dataset-csv
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