CS464 Introduction to Machine Learning
Fall 2021
Homework 1

Due: November 7, 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.
  – q2main.* and q3main.*: The main code files of your work for the second and the third questions. 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.
  – If you want to upload separate files for each coding question, add the question number to the file name. For example, q31main.* will be for Question 3.1 and q33amain.* will be your code for Question 3.3.a .
  – 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.
  – For any questions regarding this homework, contact sina.barazandeh@bilkent.edu.tr.
- You are NOT allowed to use any machine learning packages, libraries or toolboxes for this assignment (such as sklearn, 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.

1 Probability [17 pts]

You are playing a game. Assume that you have 2 coins, A and B. The probability of obtaining heads tossing A is $P1$, and the probability of obtaining heads tossing B is $P2$. Also, each time you randomly pick A or B, and the probability of picking A is $P3$. You have 10 chances and at the end of the game, the score you'll get is the number of heads you have obtained.

Question 1.1 [**3 pts**] What is the probability of getting your first (A, heads) pair in your 8th trial?

Question 1.2 [**5 pts**] What is the expected value of your score after 10 trials? The answer should be based on the given parameters.

Question 1.3 Your friend Oliver is great at predicting binary occurrences! This time he will help you play the game, but with a single coin by which the probability of obtaining heads is 0.4. Every time Oliver tells you that you'll obtain heads, then the probability of you getting heads in the next trial is 0.95, and if he tells you that you'll not obtain heads next time, then with a 0.99 probability you will not obtain heads. Also, once in 100 trials, Oliver predicts that you will not obtain heads.

Question 1.3.a [**3 pts**] How are the given probabilities for Oliver's guessing performance measured? How can you examine the validity of the probabilities?

Question 1.3.b [**2 pts**] What is the probability of Oliver predicting 8 heads in a row?

Question 1.3.c [**4 pts**] You toss the coin and you obtain heads. What's the probability that before tossing, Oliver had predicted that you will not obtain heads?

2 kNN Diabetes Classifier [28 pts]

Your job is to program a diabetes classifier using the given dataset [2]. It's believed that the given features can help people evaluate their risk of developing diabetes. For this task, we want use the kNN classification algorithm.

Dataset

You are given a labeled dataset with 769 samples and 8 features. The dataset has been split into training and test sets with the 80 to 20 ratio (20% of the samples are taken for test). You will use the following files:

- `diabetes-train-features.csv`
- `diabetes-train-labels.csv`
- `diabetes-test-features.csv`
- `diabetes-test-labels.csv`

The files that end with `features.csv` contain the features and the files ending with `labels.csv` contain the ground truth labels which are binary (diagnosed with diabetes or not).

Confusion Matrix [3]

After you train a binary classifier, you will use the test set to evaluate the performance of your model. There are 4 values that are informative about the performance of a classifier:

- `True Positives:` The number of test samples that the true labels are positive and the model has also predicted them as positive.
- `False Positives:` The number of test samples that the true labels are negative but the model has predicted them as positive(error).
- `True Negatives:` The number of test samples that the true labels are negative and the model has also predicted them as negative.
- `False Negatives:` The number of test samples that the true labels are positive but the model has predicted them as negative(error).

Confusion matrix is a 2x2 matrix presented as below:

		Actual	
		Positive	Negative
Predicted	Positive	TP	FN
	Negative	FP	TN

Note: Positive and Negative can be any binary class pairs. For instance, in this case positive means that the patient is diagnosed with diabetes, or in spam message classification it means that the given message is a spam.

Question 2.1 **[4 pts]** What distance metric would you use for this task? Briefly explain why you don't use the other options.

Question 2.2 **[2 pts]** In classification in general, we are free to select m features from the present W features. Why do you think we might not want to use all the present features in a dataset?

Question 2.3 (Coding*) **[17 pts]** Train your kNN classifier with k = 9. First, use all the present features in the given dataset, and then use Backward Elimination[4] as your feature selection method to reduce the number of features and train the model with the reduced feature set and repeat. The goal is to remove as many as possible features to achieve higher accuracy, so you should continue backward elimination until it does not serve this purpose. At each step, evaluate your model using the test set and report the accuracy of the model for different steps using tables or plots(also mention the feature column names you remove at each step).

Note: In this question a "step" is where you achieve a higher accuracy in the backward elimination process.

Note: Take note of the training and validation time of your classifier at each step. You'll need them in the next question.

Question 2.4 **[5 pts]** Comment on the training and the validation times you've noted in the previous question. Use plots or tables to support your comment.

3 Spam SMS Message Detection [55 pts]

You are assigned to a project that the goal is to develop a classifier that can detect spam messages.

Dataset

The given dataset is a preprocessed and modified version of an SMS message dataset [1]. It is based on 4891 real messages which are either spam or not. The messages 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 messages.
- **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 less than 3 times in all data set are removed from messages in order to reduce the size of the data.
- **Removal of short messages:** Messages that have less than 3 meaningful terms in them are removed.

The data has been already split into two subsets: a 3912 message subset for training and a 979 message 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:

- `sms_train_features.csv`
- `sms_train_labels.csv`
- `sms_test_features.csv`
- `sms_test_labels.csv`
- `vocabulary.txt`

The files that start with `features` contain the features and the files starting with `labels` contain the ground truth labels. You do not need the `vocabulary.txt` for this specific task and it's only given as part of the dataset.

In the feature files each row contains the feature vector for a message. The j-th term in a row i is the occurrence information of the j-th vocabulary word in the i-th message. The size of the vocabulary is 3458. The label files include the ground truth label for the corresponding message (label 0 is normal message, and label 1 is spam), the order of the messages (rows) are the same as the features file. That is the i-th row in the files corresponds to the same message. Each message 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 message. If we have a particular message 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 message D_i and x_j represents the actual word that appears in the j^{th} position in the message, whereas n_i represents the number of positions in the message. As a concrete example, we might have the first message (D_1) which contains 200 words ($n_1 = 200$). The document might be of positive message ($y_k = 1$) and the 15th position in the message 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 message n_i. That means that the feature vector for each message will be of different sizes. Also, the above formal definition of a feature vector $\vec{x}$ for a message 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 message 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 a message D_i. As a concrete example, we might have a vocabulary of size of 1309, $V = 1309$. The first message (D_1) might be a spam ($y_k = 1$) 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 message 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 message classes (in this case spam or normal mail) given a particular message D_i. We can use Bayes Rule to write:

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

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

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

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

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:

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

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

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 3.1 (Coding*) [20 pts] Train a Multinomial Naive Bayes model on the training set and evaluate your model on the given test set. Report the accuracy and the confusion matrix of your classifier on the test set.

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 3.2 [3 pts] How many parameters do we need to estimate for this model?

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

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

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

Bag-of-Words Representation and Bernoulli Naive Bayes Model

Question 3.3 Train a Bernoulli Naive Bayes classifier using different feature sets. For this purpose, use the Mutual Information[4] as your feature selection method.

Question 3.3.a (Coding*) [23 pts] Select different number of features in steps. First, try 100 features and then use 100 more features at each step up to 600 features (6 steps). Report the accuracy of your model and the training time of the classifier for each step. You may use tables or plots.

Question 3.3.b [4 pts] Do you see a relationship between the time complexity of the algorithm and the number of features used to train the classifier at each step? Explain.

Remember that this time, if the j-th word exist in the i-th message 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).

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

, 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:

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

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

Question 3.4 [5 pts] Compare the results of the Bernoulli Bayes classifier to the Multinomial Bayes classifier.

References

1. SMS Messages Dataset https://www.dt.fee.unicamp.br/~tiago/smsspamcollection/
2. Diabetes Dataset https://www.kaggle.com/pritsheta/diabetes-dataset
3. Machine Learning Metrics https://stanford.edu/~shervine/teaching/cs-229/cheatsheet-machine-learning-tips-and-tricks
4. Text Classification and Naive Bayes https://nlp.stanford.edu/IR-book/pdf/13bayes.pdf
5. Weighted K Nearest Neighbor https://www.csee.umbc.edu/~tinoosh/cmpe650/slides/K_Nearest_Neighbor_Algorithm.pdf