CS464 Machine Learning
Spring 2019
Homework 1

Due: March 24 11:59 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`. If your name is Oğuzhan Karakahya and you are from Section 1 for instance, then you should submit a file with name `CS464_HW1_1_Oguzhan_Karakahya`. Please do not use Turkish letters in your package name. The code you submit should be in a format easy to run, main script to call other functions. You must also provide us with a README file that tells us how we can execute/call your program.
- 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.).
- If you do not follow the submission routes, deadlines and specifications (codes, report, etc), it will lead to significant grade deduction.

1 The Chess Game [15 pts]

William Gates is about to play a three-game chess match with Steve Jobs, and wants to find the strategy that maximizes his winning chances. Each game ends with either a win by one of the players, or a draw. If the score is tied at the end of the two games, the match goes into sudden-death mode, and the players continue to play until the first time one of them wins a game (and the match). William has two playing styles, timid and bold, and he can choose one of the two at will in each game, no matter what style he chose in previous games. With timid play, he draws with probability 0.65, and he loses with probability 0.35. With bold play. he wins with probability 0.60, and he loses with probability 0.4. William will always play bold during sudden death, but may switch style between other games.

Question 1.1 [3 pts] Find the probability that Gates wins the match if he plays bold in all first three games.

Question 1.2 [3 pts] Find the probability that Gates wins the match if he plays timid in all first three games.

Question 1.3 [3 pts] If Steve is known to be the winner of the first game, find the probability that William played bold.

Question 1.4 [6 pts] After playing a while, Steve starts to predict William's strategy and plays according to his prediction. If he guesses that William plays timid, he plays bold. Otherwise, he plays timid. The probability that Steve correctly predicted William's strategy is 0.75 and if he guesses correctly, he has a 0.85 chance of winning regardless of the strategy. Find the probability that Steve wins the next game.

2 Medical Diagnosis [10 pts]

Assume that, as a clinic worker, you are asked to conduct lab tests for diagnosis of a disease. From experiments, it is known that any person in the population is either has the disease (positive), or has not (negative), i.e. there is no carrier. Over the entire population of people only 0.005 have this disease and the lab test returns a correct positive result in only 97% of the cases in which the disease is actually present and a correct negative result in only 98% of the cases in which the disease is <u>not</u> present. The state of the patient is presented with random variable S and the test results are presented with random variable T.

Question 2.1 [5 pts] Obtain the probabilities $P(S = \mathit{disease})$, $P(S = \mathit{healthy})$, $P(T = \mathit{positive} \mid S = \mathit{disease})$, $P(T = \mathit{negative} \mid S = \mathit{disease})$, $P(T = \mathit{positive} \mid S = \mathit{healthy})$ and $P(T = \mathit{negative} \mid S = \mathit{healthy})$.

Question 2.2 [5 pts] Suppose a new patient comes to the clinic and the test returns a positive result. Show whether the patient should be diagnosed as having the disease or not.

3 MLE and MAP [15 pts]

The Poisson distribution is a useful discrete distribution which can be used to model the number of occurrences of something per unit time. If X is Poisson distributed, i.e. $X \sim \mathit{Poisson}(\lambda)$, its probability mass function takes the following form:

$$\mathbf{P}\left(X = x \mid \lambda\right) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Assume now we have n identically and independently drawn data points from $\mathit{Poisson}(\lambda) : \mathcal{D} = \{x_1, \ldots, x_n\}$

Question 3.1 [5 pts] Derive an expression for maximum likelihood estimate (MLE) of λ.

Question 3.2 [5 pts] Assume that prior distribution for λ is $\mathit{Gamma}(x \mid \alpha, \beta)$ where Gamma distribution is defined as $\mathit{Gamma}(x \mid \alpha, \beta) = \frac{\beta^\alpha x^{\alpha-1} e^{-\beta x}}{\Gamma(\alpha)}$ where $\alpha, \beta > 0$, derive an expression for maximum a posterior (MAP) estimate of λ.

Question 3.3 [5 pts] Show that MLE estimate of λ and MAP estimate of λ with uniform prior $\lambda \sim U(a, b)$ is the same for any a and b.

4 Sentiment Analysis on Tweets [60 pts]

As a computer scientist working in an airline company, your job is to analyze online data to measure customer satisfaction for the airlines in the US. The questions summarize the model, therefore, please read all questions before starting coding.

Dataset

Your dataset is a preprocessed and modified subset of the Twitter US Airline Data Set [1]. It is based on over 14000 real tweets about airlines in US. Tweets 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 tweets.
- **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 tweets in order to reduce the size of the data.

The data has been already split into two subsets: a 11712-tweet subset for training and a 2928-tweet subset for testing (consider this as your validation set and imagine there is another test set which is not given to you). The features have been generated for you. You will use the following files:

- `question-4-train-features.csv`
- `question-4-train-labels.csv`
- `question-4-test-features.csv`
- `question-4-test-labels.csv`
- `question-4-vocab.txt`

The files that ends with `features.csv` contains the features and the files ending with `labels.csv` contains the ground truth labels.

In the feature files each row contains the feature vector for a tweet. The j-th term in a row i is the occurrence information of the j-th vocabulary word in the i-th tweet. The size of the vocabulary is 5722. The label files include the ground truth label for the corresponding tweet, the order of the tweets (rows) are the same as the features file. That is the i-th row in the files corresponds to the same tweet. Any tweet is labeled as either `positive`, `negative` or `neutral`.

The file ending with `vocab.txt` is the vocabulary file in which the first element in j-th is the word that j-th feature represents and the second element is the term frequency of the word in the all data set (training and test sets) regardless of the classes (positive, neutral and negative).

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 tweet is conditionally independent of the word position given the class of the tweet. If we have a particular tweet document D_i with n_i words in it, we can compute the probability that D_i comes from the class y_k as:

$$\mathbf{P}\left(D_i \,|\, Y = y_k\right) = \mathbf{P}\left(X_1 = x_1, X_2 = x_2, .., X_{n_i} = x_{n_i} \,|\, Y = y_k\right) = \prod_{j=1}^{n_i} \mathbf{P}\left(X_j = x_j \,|\, Y = y_k\right) \tag{4.1}$$

In Eq. (4.1), X_j represents the j^{th} position in tweet D_i and x_j represents the actual word that appears in the j^{th} position in the tweet, whereas n_i represents the number of positions in the tweet. As a concrete example, we might have the first tweet (D_1) which contains 200 words ($n_1 = 200$). The document might be of positive tweet ($y_k = 1$) and the 15th position in the tweet 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 tweet n_i. That means that the feature vector for each tweet will be of different sizes. Also, the above formal definition of a feature vector $\vec{x}$ for a tweet says that $x_j = k$ if the j-th word in this tweet 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 tweet i. As shown in the lecture slides, we can slightly change the representation, which makes it easier to implement:

$$\mathbf{P}\left(D_i \,|\, Y = y_k\right) = \prod_{j=1}^{V} \mathbf{P}\left(X_j \,|\, Y = y_k\right)^{t_{w_j,i}} \tag{4.2}$$

,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 tweet D_i. As a concrete example, we might have a vocabulary of size of 1309, $V = 1309$. The first tweet (D_1) might be a positive tweet ($y_k = 1$) and the 80-th word in the vocabulary, w_{80}, is "amazing" and $t_{w_{80},1} = 2$, which says the word "amazing" appears 2 times in tweet D_1. Contemplate on why these two models (Eq. (4.1) and Eq. (4.2)) are equivalent.

In the classification problem, we are interested in the probability distribution over the tweet classes (in this case positive, negative and neutral tweets) given a particular tweet D_i. We can use Bayes Rule to write:

$$\mathbf{P}(Y = y_k|D_i) = \frac{\mathbf{P}(Y = y_k)\prod_{j=1}^{V}\mathbf{P}(X_j\,|\,Y = y)^{t_{w_j,i}}}{\sum_k \mathbf{P}(Y = y_k)\prod_{j=1}^{V}\mathbf{P}(X_j\,|\,Y = y_k)^{t_{w_j,i}}} \tag{4.3}$$

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

$$\mathbf{P}(Y = y_k|D_i) \propto \mathbf{P}(Y = y_k)\prod_{j=1}^{V}\mathbf{P}(X_j\,|\,Y = y)^{t_{w_j,i}} \tag{4.4}$$

$$\hat{y}_i = \arg\max_{y_k} \mathbf{P}(Y = y_k \mid D_i) = \arg\max_{y_k} \mathbf{P}(Y = y_k)\prod_{j=1}^{V}\mathbf{P}(X_j\,|\,Y = y_k)^{t_{w_j,i}} \tag{4.5}$$

Question 4.1 [2 points] Explain why the denominator can be ignored in Eq. (4.3).

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:

$$\hat{y}_i = \arg\max_{y}\left(\log \mathbf{P}(Y = y_k) + \sum_{j=1}^{V} t_{w_j,i} * \log \mathbf{P}(X_j\,|\,Y = y_k)\right) \tag{4.6}$$

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

Question 4.2 [5 points] If the the ratio of the classes in a dataset is close to each other, it is a called "balanced" class distribution if not it is skewed. What is the percentage of negative tweets in the `question-4-train-labels.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:

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

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

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

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

- $T_{j,neutral}$ is the number of occurrences of the word j in neutral tweets in the training set including the multiple occurrences of the word in a single tweet.
- $T_{j,positive}$ is the number of occurrences of the word j in positive tweets in the training set including the multiple occurrences of the word in a single tweet.

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

Question 4.3 [2 points] How many parameters do we need to estimate for this model?

Question 4.4 (Coding) [20 points] Train a Naive Bayes classifier using all of the data in the training set (`question-4-train-features.csv` and `question-4-train-labels.csv`). Test your classifier on the test data (`question-4-test-features.txt` and `question-4-test-labels.txt`), and report the testing accuracy 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 $0 * \log 0 = 0$ (note that $a * \log 0$ is as it is, that is -inf). In case of ties, you should predict "neutral". Report your test set accuracy. What did your classifier end up predicting? Why is using the MLE estimate a bad idea in this situation?

Question 4.5 (Coding) [4 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.

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

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

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

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

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. Comment on the results.

Bag-of-Words Representation and Bernoulli Naive Bayes Model

Question 4.6 (Coding) [20 points] Train a Bernoulli Naive Bayes classifier using all of the data in the training set (`question-4-train-features.csv` and `question-4-train-labels.csv`). Test your classifier on the test data (`question-4-test-features.txt` and `question-4-test-labels.txt`, and report the testing accuracy as well as how many wrong predictions were made.

Remember that this time, if the j-th word exist in i-th tweet 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. (4.7). 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 "neutral". Report your test set accuracy in your written report. What did your classifier end up predicting? Compare your results with the Multinomial Model.

$$\hat{y}_i = \arg\max_{y} \left(\log \mathbf{P}\left(Y = y_k\right) + \log\Big(\prod_{j=1}^{V} t_j * \mathbf{P}\left(X_j \,|\, Y = y_k\right) + (1 - t_j) * (1 - \mathbf{P}\left(X_j \,|\, Y = y_k\right))\Big) \right) \tag{4.7}$$

, 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:

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

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

Question 4.7 (Coding) [7 points] Using `question-4-vocab.txt` file, find the most commonly used 20 words in each tweet class in the training set and make comments on them. Do you think the most common words are as expected? Does the models that you have constructed are interpretable?

References

1. Twitter Airline Sentiment dataset. https://www.kaggle.com/crowdflower/twitter-airline-sentiment
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