CS 505 – Spring 2021 – Assignment 3 (100 pts Bonus: 10 pts) – Text Classification, Vector Space Model

Problems due 11:59PM EST, March 12.

In this assignment, you will learn about **text classification** and **vector space models**, and use python libraries such as **sklearn** and **gensim**, which are popular in NLP. You have 2 weeks to finish this particular assignment.

Submit in Blackboard by 11:59PM EST, March **12.**
–Please indicate names of those you collaborate with.
–Every late day will reduce your score by 20
–After 2 days (i.e., if you submit on the 3rd day after due date), it will be marked 0.

Submit your (1) code/Jupyter notebook and (2) write up in one zip file.

When necessary, you must show how you derive your answer

Problem 1. *Naive Bayes and Logistic Regression Classification (35 pts)*

1. *Consider the task of learning **Y** from **X**, where the class label **Y** $\in 0, 1$ and each input **X** is represented with n features i.e. **X** $= \langle X_1, X_2, ..., X_n \rangle$, where each X_i is a continuous variable that follows a Gaussian distribution.*
   (a) *(3 pts) List the parameters that you would need to learn to classify an example using a Naive Bayes (NB) classier*
   (b) *(2 pts) What is the total number of parameters you need (in terms of n)?*
   (c) *(5 pts) Formulate $P(Y|X)$ of this Naive Bayes classifier in terms of these parameters and feature variables X_i. Simplify and show that under this Gaussian assumption, $P(Y|X)$ takes the form of Logistic Regression.*
2. *Consider a simple classification problem using Naive Bayes to determine whether a review of a beach resort is positive (1) or negative (0) i.e., $Y : Sentiment \in 0, 1$ given two features: (1) whether the review contains mention of the summer season $X_1 : Summer \in 0, 1$ and (2) whether the review contains mention of the rowdiness of the resort $X_2 : Rowdy \in 0, 1$. From the training data, we estimate that $P(Sentiment = 1) = 0.5$, $P(Summer = 1|Sentiment = 1) = 0.8$, $P(Rowdy = 1|Sentiment = 1) = 0.4$, $P(Summer = 1|Sentiment = 0) = 0.7$, and $P(Rowdy = 1|Sentiment = 0) = 0.5$. Assume that the data satisfies Naive Bayes assumption of conditional independence of the feature variables given the sentiment label.*
   (a) *(2 pts) Write down Naive Bayes formulation of $P(Y|X)$ using both the features $Summer$ and $Rowdy$ and the decision rule using $P(Y|X)$ i.e., how do you decide using $P(Y|X)$ if a review is positive or negative?*
   (b) *(5 pts) What is the expected error rate of your Naive Bayes classifier? i.e., the probability of observations where the label is different than predicted.*
   (c) *(2 pts) What is the joint probability that the sentiment of a review is positive and that the review contains mentions of $Summer$ and $Rowdy$ i.e., $P(Sentiment = 1, Summer = 1, Rowdy = 1)$?*
   (d) *(3 pts) Your boss decides to add another feature to your Naive Bayes classification model that is whether or not the review contains mentions of the Winter season $X_3 : Winter \in 0, 1$. Assume that a review that contains mentions of season can mention either $Summer$ **or** $Winter$ but cannot mention both i.e., it cannot have $Summer = 1$ **and** $Winter = 1$ (and similarly, it cannot have $Summer = 0$ **and** $Winter = 0$). In this case, are any of the NB assumptions violated? Why? What is the joint probability that the sentiment of a review is positive and that the review contains mentions of $Summer$ and $Rowdy$ and does not contain mention of $Winter$? i.e., $P(Sentiment = 1, Summer = 1, Rowdy = 1, Winter = 0$?*
   (e) *(3 pts) What is the expected error rate of your NB classifier using these three features?*
   (f) *(3 pts) Does the performance of your NB classifier improve with this addition of new feature $Winter$? Explain why.*
3. *Imagine that a certain important feature is never observed in the training data e.g., mentions of cleanliness $Clean \in 0, 1$, but it occurs in the test data.*
   (a) *(2 pts) What will happen when your NB classifier predicts the probability of this test instance? Explain why this situation is undesirable.*
   (b) *(5 pts) Will logistic regression have a similar problem? Explain concretely why or why not by looking at the formulation of the weight update in logistic regression.*

Problem 2. *Twitter Sentiment Classification with sklearn (35 pts, bonus: 10 pts)*

The file: sentiment-train.csv contains 60k tweets annotated by their sentiments (0: negative, 1: positive), which is a sample of a very large sentiment corpus that has been weakly annotated based on the emojis contained in the tweets. See here for the full description of the data and to download the full corpus (Note that the full corpus contains "neutral" tweets, which we do not include in our test set: sentiment-test.csv).

1. *(5 pts) Using sklearn, train a Multinomial Naive Bayes classifier (with default parameters) to predict sentiment on the training data, featurizing the data using CountVectorizer (also in sklearn). Use the default parameters of CountVectorizer, except for the parameter* ***stop_words="english"*** *(to remove stop words from the list of features) and* ***max_features = 1000*** *(to limit the number of bag-of-word features to only the top 1k words based on frequency across the corpus). You should learn more about CountVectorizer parameters and usage here. Report the accuracy of the trained classifier on the test set.*
2. *(3 pts) Instead of CountVectorizer, featurize your data using TFIDFVectorizer in sklearn with the same parameters as before. Using these features, train MultinomialNB classifier with default parameters and report the accuracy of the trained classifier on the test set. Does using TFIDF counts as features improve the classification accuracy?*
3. *(5 pts) Using sklearn, train a logistic regression classifier on your training data, using CountVectorizer to featurize your data (with the same parameters as before). Report the accuracy of the trained classifier on the test set. Which classifier performs better on the test set?*
4. *(2 pts) Train a logistic regression classifier as before, using TFIDFVectorizer (with the same parameter as before) to featurize your data. Report the accuracy of the trained classifier on the test set.*
5. *Use StratifiedKFold in sklearn to split your training data into 5 splits while maintaining label proportions of your training data.*
   (a) *(8 pts) Conduct 5-fold cross validation experiments on your training data: training a Multinomial NB classifier with TFIDFVectorizer using stop_words="english" and different max_features (= 1000, 2000, 3000, or 4000). Report the average accuracies of these different max_features across folds.*
   (b) *(2 pts) Select max_features value that has the highest average accuracy in your cross-validation experiments and train a Multinomial NB classifier on your whole training data using this parameter to featurize your data. Report the accuracy of this trained classifier on the test set.*
6. *Using word2vec as dense features for classification*
   (a) *(3 pts) Use gensim library to learn 300-dimensional word2vec representations from the tokenized and lowercased tweets[1] in your* ***training*** *data (you can use default parameters).*
   (b) *(3 pts) Given the learned word2vec representations, construct a vector representation of each tweet as the average of all the word vectors in the tweet[2]. Ignore words that do not have vector representations – since by default gensim word2vec model only learns vector representations for words that appear at least 5 times across the train set.*
   (c) *(1 pt) Train a logistic regression classifier using the above vector representation of tweets as your features. Report the accuracy of the trained classifier on the test set. Does dense feature representation improve the accuracy of your logistic regression classifier?*
   (d) *(3 pts) Construct a vector representation of each tweet as the average of all the word vectors in the tweet[2] after removing stop words[3] (and ignoring words that do not have vector representations as before). Train a logistic regression classifier on this new representation of tweets as your features. Report the accuracy of the trained classifier on the test set. Does removing stop words improve performance?*
7. *(Bonus: 10 pts). Train a Multinomial NB classifier with CountVectorizer (with the best number of max_features) on the entirety of the 1.6m twitter sentiment training data that you can download from here. To help reduce the burden of processing on CountVectorizer, you can first clean each tweet (tokenize, lowercase, and remove stop words) using NLTK or Spacy. Report the accuracy of this trained classifier on the test set. Does having a huge amount of training data allow this simple classifier like NB with this simple bag-of-words features to perform well on the test set?*

[1] You can use either NLTK or Spacy to tokenize your tweets

[2] There is research that suggests weighted average of these vectors are better, where the vector representation of a sentence v_s is defined as $v_s = \frac{1}{|s|}\sum_{w \in s} \frac{a}{a+p(w)} v_w$ where $a = 10^{-3}$, $p(w)$ is the unigram probability of the word w in the entire corpus and v_w is the vector representation of word w. Optionally, you can choose to use this weighted average whenever you need to create a sentence representation from the words in the sentence

[3] You can use NLTK or Spacy to remove stop words

Problem 3. *Vector Space Models (30 pts)*

The file: will_play_text.csv contains lines from William Shakespeare's plays. The second column of the file contains the name of the play, while the fifth and the sixth contain the name of the character who spoke and what they spoke, respectively. Tokenize and lower case each line in will_play_text.csv using NLTK or spacy. For this question, include generated visualizations in your write up.

1. *(3 pts) Create a term-document matrix where each row represents a word in the vocabulary and each column represents a play. The file vocab.txt lists the words in the vocabulary, while the file play_names.txt lists the names of the plays. Each entry in this matrix represents the number of times a particular word (defined by the row) occurs in a particular play (defined by the column). You can use CountVectorizer in sklearn to help.*

2. *(3 pts) From your term-document matrix, use PCA in sklearn to create a 2-dimensional representation of each play. Visualize these representations to see which plays are most similar to each other. You can follow the tutorial here to create the visualization. What plays are similar to each other? Do they match the grouping of Shakespeare's plays into comedies, histories, and tragedies here?*

3. *(2 pts) Create another term-document matrix where each row represents a word in the vocabulary and each column represents a play, but with TFIDF counts. You can use TFIDFVectorizer in sklearn to help.*

4. *(2 pts) Use PCA again on these TFIDF term-document matrix and visualize the plays. Does using TFIDF give you better grouping of plays?*

5. *(2 pts) Create a word-word matrix where each row (and each column) represents a word in the vocabulary (vocab.txt). Each entry in this matrix represents the number of times a particular word (defined by the row) co-occurs with another word (defined by the column) in a sentence (i.e., line in will_play_text.csv).*

6. *(6 pts) Using the row word vectors, create a representation of a play as the average of all the word vectors in the play[4]. Use these vector representations of plays to compute average pairwise cosine-similarity between plays that are comedies. Compute the same for plays that are histories, and plays that are tragedies. You can use the grouping of plays in here.*

7. *(6 pts) Use gensim to learn 100-dimensional word2vec representation of the words in the play (you can use default parameters but with min_count=1 so you can learn vector representations of all the words in your data). Use the learned word2vec representation to construct vector representations of plays as the average of all the word vectors in the play[4]. Use these vector representations of plays to compute average pairwise cosine-similarity between plays that are comedies. Compute the same for plays that are histories, and plays that are tragedies.*

8. *(3 pts) Use the learned word2vec representation of words to construct the vector representation of each character as the average of all the word vectors the character spoke[4]. Visualize the characters using PCA.*

9. *(3 pts) Mention 3 interesting insights with respect to the grouping of plays and/or characters e.g., what are characters that are most similar/dissimilar to each other? Do the vector representations of female characters differ distinguishably from male ones? Can you find plays that are central to each category (i.e., comedies, histories, tragedies)?*

[4]Optionally as before, you can also try weighted average of these vectors, i.e., by defining the vector representation of the play v_p as $v_p = \frac{1}{|p|}\sum_{w \in p} \frac{a}{a+p(w)} v_w$ where $a = 10^{-3}$, $p(w)$ is the unigram probability of the word w in the entire corpus and v_w is the vector representation of word w