

Assignment 2: Naive Bayes Classifier

1 Instructions

- Use python programming language for your implementation.
- Use appropriate approach if you find some attribute is missing in your data.
- Report must contain step-wise description of your implementation and analysis of results. Since data analysis is a crucial task for any machine learning algorithm, report should demonstrate detailed analysis of results and conclusion. It should also clearly mention the steps to run your code.
- Do not use any direct in-built functions of libraries to implement naive bayes classifier algorithm, otherwise -10 will be deducted.

2 Dataset

- Spooky Author Identification, <https://www.kaggle.com/c/spooky-author-identification/overview>.
Each row of the dataset contains three fields: id, text and author. The task is to predict the author from the text. Download the dataset from the given link and proceed.

3 Problem statement: Naive bayes classifier

1. For your dataset, each example is a raw text. In this step, we will extract features from the raw texts. You have to create an $r \times c$ binary feature matrix M where r is the number of examples and c is the size of the vocabulary consisting of distinct words present in the dataset. Each row corresponds to an example of the dataset and each column corresponds to a word in the vocabulary. $M_{ij} = 1$ if and only if the j -th word is present in the text of the i example. **30**

Note the following points while creating the feature matrix:

- (a) Use only the train split (train.csv) of the dataset for your experiment. Which mean you have to create the feature matrix M from train.csv only.

- (b) To read data from train.csv, you can use the python package pandas. Use the function `pandas.read_csv()` to read the data.
 - (c) Since each example is a raw text, split the text into words (this step is called tokenization). You can use the python 're' package for this. Use the function `re.findall("[a-z0-9]+", text.lower())` to split the texts into words.
 - (d) Once you have splitted the texts into words, remove all the uninformative words (also referred to stopwords) like article, prepositions, verbs etc. You can find a list of stopwords at <https://gist.github.com/sebleier/554280>.
 - (e) You may find the matrix M to be very large to fit into the memory. In that case, you can use sparse representation of matrix. You can use either of `scipy.sparse.lil_matrix` or `scipy.sparse.csr_matrix` which ever you find more useful.
2. Randomly split the feature matrix into train split and test split with 70-30 ratio. Train a Naive Bayes classifier on the train split. Compute and report the accuracy of the classifier on the test split. **5+10+10**
 3. Train a Naive Bayes classifier using Laplace correction on the same train split and report the accuracy on the test split. **20+5**
 4. A report explaining the implementation and analysis of the results by providing the 95% confidence interval of the accuracy, precision, f-score, sensitivity and specificity for questions 2 and 3. **20**