**Information Retrieval Project**

**HOW TO RUN**

Download the repository to your local computer using:

git clone <https://github.com/aievangelist/info-retrieval.git>

There are separate Jupyter notebooks for each dataset in the code:

* IR Project-DB World
* IR Project-Legal Case
* IR\_Sentence\_Classification-Final

You can run them individually to see the result of our analysis on each dataset. The datasets have already been uploaded into the repository for your convenience and the paths are referred to relatively, therefore there is no need to change any code.

**NOTES ON CHOOSING DATASETS**

We have chosen three datasets for our project. These are:

* DBWorld Emails (TF Matrix Format)
* Sentence Classification (Text Format)
* Legal Case Reports (XML Format)

The reason for choosing these was that they were the only ones on which labelling was available and we can only run classification tasks such as kNN, Naive Bayes and Rocchhio on pre-labelled datasets. Other three datasets are clustering problems and it was not requirement of this project.

**DETAILS OF ANALYSIS**

Three types of classification tasks have been run on each dataset:

* Naive Bayes Classification
* kNN Classification
* Rocchhio Classification

**DATASET 1 # DBWORLD EMAILS**

<https://archive.ics.uci.edu/ml/datasets/DBWorld+e-mails>

**Dataset and Setup:**

The data can be found in the .mat files in the dataset. There are separate file for subjects and bodies in the data. These have to be read separately with their labels and classification tasks applied to them separately.

**Execution:**

* Read the data from the .mat files using loadmat
* The dataset is preprocessed. Therefore there is no preprocessing to be done.
* However, the data is in the form of a TF Matrix. Therefore we calculate IDF of each term and then multiply with the TF Matrix.
* The dataset is then split into training and test sets using scikitlearn’s training\_test\_split.
* Each classification task is run on the dataset and analysed using precision, recall, F1-Measure and Confusion Matrix.

**DATASET 2 # SENTENCE CLASSIFICATION**

<https://archive.ics.uci.edu/ml/datasets/Sentence+Classification>

**Dataset and Setup:**

* Split the labeled\_articles folder into training\_set and test\_set folder.
* Place 72 files in training\_set folder and 18 files in test\_set folder.
* Place the notebook file in the same folder as the two folders are placed in.
* "stop words" for this problem are given so place stopwords folder in the current folder.
* "unlabelled" folder is not used in this exercise as we want to measure the performance of the system with labelled data.

**Execution:**

* First all filenames are read from training\_set folder and training\_set folder.
* Each file has multiple lines. Each line starts with the "label" and than "sentence" seperated by "tab".
* Files are read and preprocessing is applied.
  + stopwords removed
  + label and sentence separated
* Labels and sentences are saved in separate files after preprocessing
* Sentences are converted into vectors using Sklearn function "TfidfVectorizer()".
* Labels are read as list and used as labels for the classifiers.
* 3 built in SKlearn classifiers are used for this exercise.
  + MultinomialNB (Naive Bayes)
  + NearestCentroid (Rocchio)
  + KNeighborsClassifier
* Files from "training\_set" folder are used for training
* Files from "test\_set" folder are used for testing.
* Results can be seen in the Python notebook.

**DATASET 3 # LEGAL CASE REPORTS**

<https://archive.ics.uci.edu/ml/datasets/Legal+Case+Reports>

**Dataset and Setup:**

Dataset is in xml format however few tags are not written in the proper format. There are three folders in the CORPUS Citation\_class Citations\_summ Fulltext We have used files from Citation\_class folder. There are 2754 files in the folder. tag consists of the label tag consists of the terms used for classification.

**Execution:**

First all filenames are read from “Citation\_class” folder(2754 files). Each file is in XML format. Files are first read as text and correction is made in the “tag”. One example is *”id=* is replaced by *id=”*. Python notebook contains more “tags” in wrong format. Once XML format is corrected than files are read using “LXML” library for XML processing. Preprocessing is applied.

* stopwords removed
* label() and sentence separated
* Lemmatization and stemming applied Labels and sentences are saved in separate files after preprocessing Sentences are converted into vectors using Sklearn function "TfidfVectorizer()". Files are read in “sparse vector format” which takes less memory than storing the whole vector. Therefore it takes less computation time. Labels are read as list and used as labels for the classifiers. Dataset is split into test and training data.

3 built in SKlearn classifiers are used for this exercise.

* MultinomialNB (Naive Bayes)
* NearestCentroid (Rocchio)
* KNeighborsClassifier