Department of Computer Engineering
University of Peradeniya

CO 544 Machine Learning and Data Mining

Introduction to text classification

11^{th} of June 2020

Objectives: The aim of the lab is to provide students hand-on experience in text classification using Python.

At the end of the lab, students should be able to,

- Preprocess a given text dataset.
- Develop classification models using different machine learning algorithms and compare performance of the models.

1. Introduction

Text classification is the process of classifying text strings or documents into different categories, depending upon the content of the document.Detecting user sentiment from a tweet, classifying an email as spam or ham, automatic tagging of customer queries,classifying news articles into different categories like Politics, Stock Market, Sports, etc. are some of the real world applications of text classification.

We can complete this task with the use of Natural Language Precessing (NLP) and classifying algoirthms. NLP enables the computers to understand and interpret human languages.

2. Text classification

(a) Importing required modules

```
import re           #importing re module with regular expressions
import nltk         #importing natural language toolkit module

from nltk.corpus import stopwords    #importing collection of stop words in NLT (you can add
                                    related stop words to the problem in addition to these)
nltk.download('stopwords')

from nltk.stem import WordNetLemmatizer    #for lemmatization
nltk.download('wordnet')
```

(b) Loading data

```
movie_data = load_files(r"txt_sentoken")
X, y = movie_data.data, movie_data.target
```

(c) Data preprocessing

' X=movie_data.data' is a list. Here we preprocess data in each row using a loop and later combine results into one list.

```
document= re.sub(r'\W', ' ', str(X[i]))       #removing special characters in i th row

document= re.sub(r'\^[a-zA-Z]\s+', ' ', document)     #removing single characters from
                                                          the beginning

document= re.sub(r'\s+[a-zA-Z]\s+', ' ', document)    #removing all single characters

document= re.sub(r'\d+', '', document)        #removing numbers

document= re.sub(r'\s+', ' ', document, flags=re.I)     #Substituting multiple spaces with
                                                                  single space

document= document.lower()   #converting to lowercase

document= document.split()        #splitting sentences into words

document= [stemmer.lemmatize(word) for word in document]      #Lemmatization
```

TODO 1: Find data preprocessing steps other than mentioned above.

(d) Convert text into numbers.

There're different approaches to convert text into the corresponding numerical form. The Bag of Words Model and the Word Embedding Model are two of the most commonly used approaches.Here I've used the Bag of Words model.

```
from sklearn.feature_extraction.text import CountVectorizer

vectorizer = CountVectorizer(max_features=1500, min_df=7, max_df=0.8,
             stop_words=stopwords.words('english'))

X = vectorizer.fit_transform(documents).toarray()   #'documents' contain all the rows which
                                                          preprocessed
```

TODO 2: Discuss advantages and disadvantages of 'Bag of Words model'

(e) Text Classification

Now classification algorithms can be applied.

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)

# Logistic regression model

from sklearn.linear_model import LogisticRegression
log_reg = LogisticRegression()
log_reg.fit(X_train,y_train)

predictions = log_reg.predict(X_test)
```

TODO 3: Train a Random Forest model, a Support Vector Machine model and a Naive Byesian classfier. Compare the accuracies of four models including the Logistic Regression model.
What is the best model? Justify your answer.