Clasificación de Textos en Lenguaje Natural

Inteligencia Artificial Avanzada

Sergio Tabares Hernández

Universidad de La Laguna

Curso 2020-2021

Indice

Introducción

En este programa se ha llevado a cabo la implementación de un algoritmo de clasificación de textos en lenguaje natural empleando la Simplificación de Markov mediante el modelo probabilístico de unigramas y empleando suavizado laplaciano y un vocabulario abierto.

Preprocesamiento

Para el preprocesamiento del corpus de datos se han realizado varias tareas:

1. Se han convertido todos los caracteres a minusculas.
2. Se han eliminado todos los caracteres diferentes a letras de la A a la Z.
3. Se han eliminado todos los caracteres blancos redundantes.
4. Se han eliminado todas las palabras reservadas (conjunciones, artículos, preposiciones y adverbios).

Librerías utilizadas

Para el desarrollo del programa se han empleado diferentes librerías para diferentes fines:

- itertools: para la unificación de una listas de dos dimensiones a una lista de una sola dimensión.
- math: para el cálculo del logaritmo de las probabilidades de cada palabra y de las clases.
- nltk: para la obtención de las palabras reservadas durante el preprocesamiento.
- re: para la utilización de expresiones regulares durante el filtrado de caracteres en el preprocesamiento.
- time: para el cálculo del tiempo de ejecución de distintas partes del código y poder llevar a cabo tareas de optimización.

Implementación de los programas

El programa ha sido seccionado en diversos ficheros con el fin de aumentar la modularidad y legibilidad del mismo:

- preprocesing.py: funciones para preprocesar los datos de los corpus utilizados.
- dataGetters.py: funciones para obtener los datos de los archivos de texto.
- corpusParser.py: funciones para obtener las distintas partes del corpus.
- vocabularyParser.py: funciones para obtener el vocabulario de las descripciones del corpus.
- classesParser.py: funciones para obtener las diferentes clases de las descripciones del corpus.
- probabilitiesEstimator.py: funciones para la fase de aprendizaje.
- corpusClassifier.py: funciones para la fase de clasificación.
- main.py: función para ejecutar el programa entero de principio a fin.

En cada fichero y función se dispone de una descripción más especifica de su funcionamiento.

Error sobre el corpus de entrenamiento

Tras una serie de pruebas realizando ligeros retoques en la fase de preprocesamiento, se ha llegado a obtener un error del 95.075%. Este valor se ha conseguido simplemente con la realización de las tareas de preprocesamiento antes nombradas y con el valor de umbral de la frecuencia de las palabras establecido con un valor de 0.