Metaheurísticas

Algoritmos genéticos y Differential Evolution en Sentiment Analysis

Resumen

Aplicación de algoritmos genéticos y Differential evolution para la optimización del comportamiento de herramientas de extracción de polaridad

Nuria Rodríguez Barroso
Universidad de Granada
rbnuria@correo.ugr.es

Índice

1. Introducción

1.1. Contexto

En los últimos años, la minería de datos ha obtenido tal relevancia en nuestra sociedad que cada vez se considera más indispensable a la hora de la toma de decisiones en una corporación. Estamos rodeados de todo tipo de datos: afinidades, ubicaciones, opiniones, información personal, etc.

Una corporación o empresa, como sabemos, ofrece un producto al cliente con el fin de satisfacerlo y además lucrarse. Sería de gran interés para esta empresa conocer cuál es la opinión del cliente sobre el producto, cuáles son las mejoras que él añadiría y cuáles son los defectos que él eliminaría.

Actualmente, obtener una opinión de un cliente sobre tu producto es mucho más fácil que pasar una encuesta, gracias a la Web 2.0. Basta con analizar las redes sociales, que están en continuo crecimiento, o analizar otras fuentes de opiniones como pueden ser TripAdvisor, Google, Amazon,etc. Pero, ¿cómo analizaríamos estas opiniones, sobre todo cuando tu producto es comprado y valorado por miles de personas? Efectivamente, hacerlo contratando personal que valore estos comentarios resultaría inviable, tanto por el tiempo que llevaría como por los recursos económicos que habría que invertir.

Sentiment Analysis pretende dar una solución a este problema. Se trata de un campo de la minería de datos que, según [Tan et al., 1999], se centra en la extracción de información útil a partir de textos. Este tema cada vez está obteniendo mayor popularidad en el ámbito de la computación. Como vemos en la figura 1 obtenida a partir de Google Trends, durante los últimos años, Sentiment Analysis ha ido ganado popularidad en las búsquedas, un claro reflejo de que se trata de un campo cada vez más importante y útil.

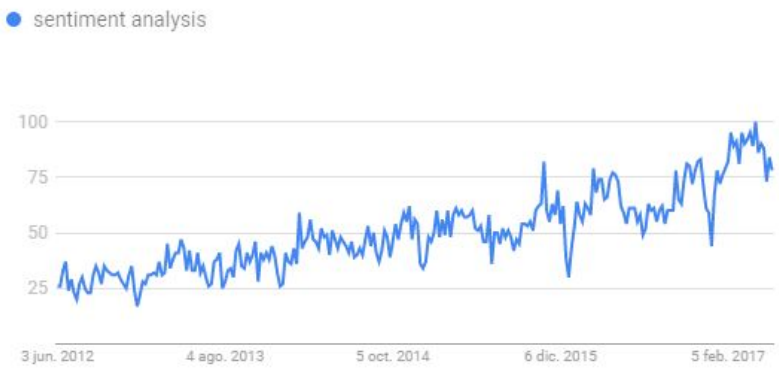

Figura 1: Relevancia del Sentiment Analysis según las búsquedas realizadas en Google

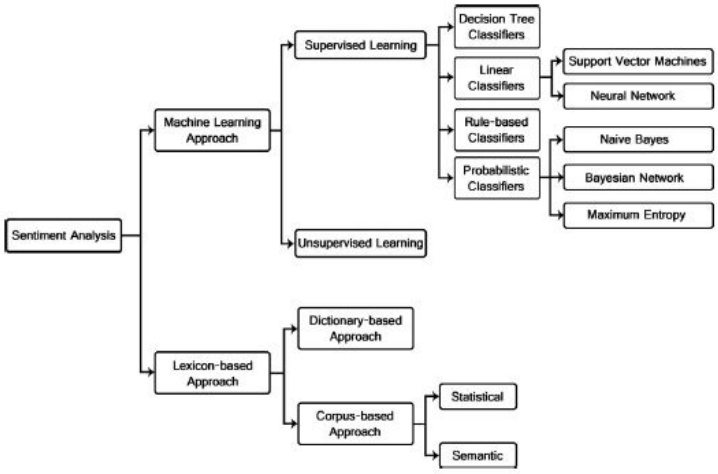

Figura 2: Esquema del problema de Sentiment Analysis

1.2. Sentiment Analysis y descripción del problema

Sentiment Analysis podría definirse como un problema de clasificación (ver 2) en el que el objetivo es identificar el sentimiento que un texto transmite. Se trata de un problema cuya principal dificultad recae en el tratamiento de información difusa, en la que además de factores más objetivos como la ortografía, existen también otros factores subjetivos tales como la ironía o el sarcasmo, a veces siendo estos difíciles de entender incluso por los seres humanos.

Para obtener esta clasificación, existen varias alternativas:

- **Aproximación por Aprendizaje Automático.** El problema se trata con técnicas de aprendizaje automático tales como Support Vector Machine o técnicas basadas en árboles. Pueden ser problemas supervisados o no supervisados. En los primeros el principal problema es que debemos tener un gran conjunto de comentarios ya etiquetados con el fin de aprender a partir de estos comentarios etiquetados. En los segundos, nuestro sentimiento se extrae a partir de funciones heurísticas u otras reglas predefinidas.
- **Aproximación por técnicas basadas en diccionarios.** Estas técnicas se basan en diccionarios elaborados manualmente donde cada palabra tiene una intensidad de sentimiento asociada. Alternativamente a las técnicas basadas en diccionarios tenemos las técnicas basadas en corpus, donde se tienen en cuenta distintos patrones sintácticos tales como palabras que expresan conectividad o por ejemplo contradicción.
- **Aproximación por herramientas de Procesamiento del Lenguaje Natural (NLP).** Estas herramientas emplean técnicas computacionales (Aprendizaje automático, inteligenia artificial, inferencia estadística...) unidas a técnicas basadas en lingüística para darnos la polaridad de un texto. Se puede definir también como un software que integra alguna de las 2 técnicas nombradas anteriormente (o ambas) y que ha sido desarrollado con fines comerciales o de investigación, facilitando al usuario el proceso de Sentiment Analysis (sin que este tenga que desarrollar su propio modelo o su propio diccionario).

Nuestro problema

Nuestro problema consiste en la optimización del comportamiento de 7 herramientas de Procesamiento de Lenguaje Natural (NLP) o herramientas de extracción de polaridad con respecto al etiquetado experto de un total de 20 datasets de comentarios y la comparación de los resultados con otras alternativas tales como la agregación de estas herramientas (entre otras). Esta optimización del comportamiento se realizará mediante la asignación de pesos a las herramientas, siendo estos pesos buscados mediante algoritmos genéticos primero y posteriormente técnicas de Differential Evolution.

Partiremos de las intensidades que cada herramienta nos devuelve para cada comentario y nuestro objetivo es que la combinación lineal de las intensidades de cada herramienta ponderadas con el peso de la solución minimice el error cuadrático, aunque los resultados se mostrarán según el error de clasificación para ser interpretado más fácilmente.

2. Algoritmos genéticos

Para este estudio vamos a considerar dos tipos de algoritmos genéticos, que surgen de la combinación del operador de cruce aritmético con 2 esquemas de evolución diferentes:

- **Esquema generacional con elitismo:** Por un lado, vamos a considerar un esquema de evolución generacional, esto es, se seleccionará una generación de padres del mismo tamaño de la población genética que serán posteriormente cruzados y mutados (algunos de ellos). Para mantener el elitismo, nos aseguraremos de que la mejor solución de la población anterior siga apareciendo en la nueva generación. Para ello, si esto no ocurre sustituiremos la peor solución generada en la nueva población por la mejor de la anterior.
- **Esquema estacionario:** Por otro lado, consideraremos un esquema de evolución estacionario, donde seleccionaremos dos padres que serán posteriormente cruzados y mutados (no siempre, ya lo veremos). Los cromosomas resultado del cruce y mutación de los padres generados, sustituirán a los dos cromosomas más débiles de la población anterior (peor función objetivo) en caso de no ser ellos mismos.

En cuanto al operador de cruce, como ya se ha comentado, usamos un **Operador CA**, u operador de cruce aritmético, que cada dos cromosomas padres, genera un cromosoma hijo fruto de la media aritmética componente a componente de ambos padres. Esto es, considerando los padres anteriores, el hijo sería de la forma:

$$c_{hijo} = (\frac{c_{11} + c_{21}}{2}, ..., \frac{c_{1n} + c_{2n}}{2})$$

Como este operador genera cada dos padres un único hijo, para generar el mismo número de hijos que con el operador BLX, habrá que utilizar el doble de padres. Esta consideración la veremos con detalle más adelante. Detallamos el algoritmo de generación de un hijo con dicho operador en Algorithm **??**.

Referencias

[Tan et al., 1999] Tan, A.-H. et al. (1999), Text mining: The state of the art and the challenges. In Proceedings of the PAKDD 1999 Workshop on Knowledge Disocovery from Advanced Databases, volume 8, pages 65-70.