Escuela Politécnica Superior, Universidad Autónoma de Madrid

Grado en Ciencia e Ingeniería de Datos

Procesamiento de Lenguaje Natural

Práctica 2: Clasificación de texto para análisis de sentimiento

Fecha de comienzo 14 de octubre de 2025

Fecha de entrega 18 de noviembre de 2025 (hasta las 15:59h)

Objetivos

Esta segunda práctica trata la **clasificación automática de texto** orientada al **análisis de sentimiento**, abordando dos objetivos principales:

- Aprender a extraer **características lingüísticas** para generar **representaciones vectoriales dispersas** de textos. Las representaciones vectoriales generadas no serán densas, es decir, no serán *embeddings* numéricos preentrenados como Word2Vec o GloVe.
- Aprender a construir y evaluar **modelos de clasificación supervisada** para la detección de la **polaridad de opinión** en reseñas de usuario.

Para ello, de forma gradual, se codificarán en Python una serie de programas que harán uso de bibliotecas como NLTK o Scikit-learn para la extracción de características lingüísticas de un texto, la representación vectorial asociada, y el entrenamiento y evaluación de modelos de clasificación.

El corpus de reseñas preprocesadas resultante de la Práctica 1 será el material de trabajo fundamental en esta práctica. Si es necesario, el corpus se puede modificar o extender en esta práctica, tanto aumentando el número de reseñas como el número de operaciones de preprocesamiento de texto. Estas operaciones podrían ser:

- Análisis gramatical (PoS *tagging*) de palabras.
- Análisis sintáctico de dependencias de oraciones.
- Eliminación de stopwords.
- *Stemming* o *lemmatization*.

Caso de uso

En la Práctica 1 se construyó un corpus anotado de reseñas en inglés sobre juegos de mesa de BoardGameGeek (BGG), que, al menos, incluye los textos de las valoraciones y su rating o puntuación numérica (entre 1 y 10).

En esta práctica se propone usar este *rating* para determinar las **clases (etiquetas) de polaridad de opinión** de las reseñas. Estas etiquetas serán consideradas en el entrenamiento y evaluación de los modelos de clasificación. Por ejemplo, se podrían[1] considerar las siguientes etiquetas de opinión y sus correspondientes umbrales a partir del *rating* numérico de la reseña.

- **Positiva**: *Ratings* iguales o superiores a 7.
- **Neutra**: *Ratings* iguales a 5 o 6.
- **Negativa**: *Ratings* iguales o inferiores a 4.

[1] Se pueden considerarse otras clases y/o umbrales de valores de ratings asociados.

Se deja a libre elección las etiquetas y umbrales de ratings. Dicha elección puede basarse en la distribución de ratings en el corpus, y, en cualquier caso, deberá estar explicada en la memoria de la práctica.

Es obligatorio realizar los pasos necesarios para obtener un **conjunto de datos balanceado** con las etiquetas consideradas, ya sea mediante una selección cuidadosa de los juegos de mesa a incluir en el corpus, o mediante la aplicación de técnicas de submuestreo o sobremuestreo.

Estructura

La práctica consta de los siguientes **cinco ejercicios** a realizar en el orden establecido:

1. Extracción de características lingüísticas de las reseñas.
2. Generación de representaciones vectoriales de las reseñas con las características lingüísticas extraídas.
3. Creación de ficheros de datos para entrenamiento, validación (si procede) y test de modelos de clasificación, a partir de las representaciones vectoriales generadas.
4. Construcción de modelos de clasificación con los ficheros de datos de entrenamiento y validación creados.
5. Evaluación de los modelos de clasificación construidos con los ficheros de test, y elaboración de un informe.

A continuación se proporciona el enunciado de cada uno de estos ejercicios.

Ejercicio 1: Extracción de características lingüísticas de reseñas

Se pide desarrollar un programa `pln_p2_XXXX_YY_e1.py` que obtenga características lingüísticas relevantes para la clasificación de textos atendiendo a su polaridad de opinión. Estas características podrían capturar el sentimiento y la subjetividad, así como la intensidad asociada, de las opiniones expresadas por los usuarios sobre juegos de mesa.

Entre otras, las siguientes son posibles características a considerar:

- **Palabras de opinión/sentimiento** (existentes en *lexicones*): Palabras cargadas de polaridad como adjetivos, adverbios o verbos.
  - Ejemplos sencillos: *"The game is **amazing**!"* (positivo); *"This game is an **awful** mess."* (negativo).
  - Ejemplos complejos: *"I **enjoy** the theme but the mechanics **frustrate** me."*; *"It's a **decent** filler game."*.
  - Estas características se basan en la presencia o el recuento de *PoS tags* específicos y existencia en *lexicones* de opinión como SentiWordNet o VADER.
- **Negaciones:** Es importante detectar la negación, ya que puede invertir la polaridad de una palabra.
  - Ejemplos sencillos: *"The theme is **not good**."* (convierte un positivo potencial en negativo); *"The rules are **not difficult**."* (convierte un negativo potencial en positivo).
  - Ejemplos complejos: *"I **fail to see** the appeal."* (implica negación de positivo); *"It **lacks** replayability."* (uso de verbo que implica negación).
  - Estas características se basan en la presencia de ciertos patrones sintácticos de dependencias o reglas de alcance de la negación.

- **Intensificadores/modificadores:** Adverbios o frases que amplifican (intensificadores), mitigan o incluso invierten la polaridad.
    - Ejemplos (intensificación): *"The strategy is **extremely deep**."*; *"The playtime is **way too long**."*
    - Ejemplos (mitigación/atenuación): *"The components are **sort of nice**."*; *"The replay value is **barely present**."*
    - Estas características se basan en la presencia de ciertos patrones sintácticos de dependencias.
- **Vocabulario de dominio:** En el contexto de BGG, pueden ser relevantes las menciones a elementos específicos del juego en combinación con palabras polarizadas.
    - Ejemplos: La frecuencia con la que aparecen términos de sentimiento cerca de **mecánicas** (*"The **dice roll** system is **clunky**),* **componentes** (*"**Miniatures** quality is **outstanding**."),* **reglas (***"The **rulebook** is **terribly written**.").*
    - Estas características se basan en términos que pueden derivarse de metadatos de juegos existentes en BGG, o por su frecuencia de aparición en el corpus.

Para su posterior acceso, las características obtenidas deben ser almacenadas en el corpus como se considere oportuno.

Ejercicio 2: Generación de representaciones vectoriales de reseñas

Se pide desarrollar un programa `pln_p2_XXXX_YY_e2.py` que transforme los textos de la reseñas a vectores dispersos.

Se plantea implementar y evaluar las siguientes representaciones vectoriales de forma independiente y conjunta:

1. **Representación(es) basada en n-gramas y frecuencias/pesos:** Generar una representación vectorial basada en la frecuencia o peso de los términos, como TF-IDF. Se debe incluir la posibilidad de probar la representación con n-gramas de palabras, al menos unigramas (n=1). El considerar bigramas (n=2) o incluso trigramas (n=3) podría resultar beneficios, pero incrementa de forma exponencial la dimensión del espacio vectorial.

   Nota sobre la dimensionalidad. Se debe tener en cuenta que la dimensión de los vectores (el número de características) crece drásticamente al incorporar n-gramas y a medida que n aumenta; esto puede conllevar un elevado coste computacional a la hora de crear los modelos de clasificación. Para abordar este problema, se puede llevar a cabo un filtrado de características (p. ej., eliminando *stopwords*, aplicando *stemming* o *lemmatization*, seleccionando las características más relevantes a través del propio TF-IDF para limitar el vocabulario, etc.).

2. **Representación(es) basada en opinión y sentimiento:** Generar una segunda representación que capture información lingüística de alto nivel relevante para minería de opinión, como la polaridad léxica, negación o el uso de modificadores e intensificadores (características extraídas en el ejercicio 1).

En ambos casos, las representaciones vectoriales deben ser almacenadas para su uso en la construcción y evaluación de los modelos de clasificación.

Ejercicio 3: Creación de ficheros de entrenamiento, validación y test

Se pide desarrollar un programa `pln_p2_XXXX_YY_e3.py` que implemente las siguientes funcionalidades:

1. **Etiquetado del corpus:** Asignar la etiqueta de polaridad (p. ej., **positiva, neutra, negativa**) a cada reseña basándose en la conversión del *rating* numérico según los umbrales definidos.
2. **Particionado del conjunto de datos**: Dividir el corpus etiquetado en al menos dos conjuntos de datos disjuntos: **entrenamiento** y **test** (p. ej., 80% y 20%). Se recomienda estratificar la división para mantener la proporción de las clases P, N, NEG. Si se considera apropiado para el ajuste de hiperparámetros, se puede crear conjuntos de **validación** a partir de los conjuntos de entrenamiento.
3. **Almacenado de datos:** Almacenar los conjuntos de datos resultantes en un formato adecuado (p. ej., **CSV o JSON**) para que sean utilizados directamente por los algoritmos de clasificación en el ejercicio 4.

Ejercicio 4: Construcción de modelos de clasificación

Se pide desarrollar un programa `pln_p2_XXXX_YY_e4.py` que entrene modelos de clasificación supervisada para predecir la polaridad de las reseñas.

Modelos a implementar. Los modelos podrían incluir:

- Clasificador Bayesiano (p. ej., *Multinomial Naive Bayes*).
- Máquinas de Soporte Vectorial (SVM).
- Random Forest o XGBoost.

Subconjuntos de características. Cada modelo se debería construir atendiendo a las dos representaciones vectoriales del ejercicio 2, considerando diferentes subconjuntos de características, p. ej., solo n-gramas, solo características lingüísticas de sentimiento, o una combinación de ambas.

Ejercicio 5: Evaluación de algoritmos de clasificación. Elaboración de un informe

Se pide desarrollar un programa `pln_p2_XXXX_YY_e5.py` que implemente la evaluación rigurosa de los modelos generados y la elaboración de un informe técnico.

Evaluación:

- Evaluar todos los modelos y sus versiones (subconjuntos de características) sobre el conjunto (o conjuntos) de test del ejercicio 3.
- Realizar ajuste de hiperparámetros, p. ej., mediante *Grid Search* o *Randomized Search*.
- Calcular y reportar las métricas de clasificación, incluyendo al menos *accuracy*, *precision*, *recall* y *F1-score* a nivel global.
- Generar las matrices de confusión para los mejores modelos.

Informe técnico:

Elaborar un informe conciso `pln_p2_XXXX_YY_informe.pdf` que:

- De manera opcional, describa brevemente aspectos relevantes relacionados con el preprocesamiento de texto, y con la construcción y tratamiento de los conjuntos de datos.
- Describa de forma concisa las características lingüísticas utilizadas para construir cada representación vectorial.

- Describa los experimentos realizados para evaluar los modelos de clasificación para las diferentes representaciones vectoriales consideradas.
- Compare y analice los resultados de evaluación obtenidos. Se deberá identificar el modelo óptimo para la tarea de clasificación de polaridad.

Lo anterior se puede complementar con un análisis de la relevancia de las características donde se justifique la elección de los subconjuntos de características y se intente cuantificar el impacto de las diferentes características lingüísticas añadidas.

El informe técnico deberá incluir los nombres y apellidos de los autores.

Entrega

Como resultado de la práctica, cada equipo deberá entregar una carpeta comprimida en un fichero comprimido ZIP.

El nombre de la carpeta y del fichero ZIP será `pln_p2_XXXX_YY.zip`, donde `XXXX` ha de sustituirse por el código del grupo de prácticas (7461 o 7462), e `YY` ha de sustituirse por el identificador de equipo: 01, 02, etc.

El fichero ZIP contendrá:

- El **código Python** (carpeta `src`) de los programas desarrollados en los cinco ejercicios.
- Los **ficheros de datos** o parte de ellos.
- El **informe técnico**.

Planificación sugerida

Se propone la siguiente planificación de trabajo:

- **1ª semana**
  - Comenzar el ejercicio 1.
- **2ª semana**
  - Finalizar el ejercicio 1.
  - Comenzar el ejercicio 2.
- **3ª semana**
  - Finalizar el ejercicio 2.
  - Realizar el ejercicio 3.
- **4ª semana**
  - Realizar el ejercicio 4.
- **5ª semana**
  - Realizar el ejercicio 5.
  - Redactar el informe técnico.