Práctica 8: Minería de texto

El objetivo de esta sesión es poner en práctica aplicaciones de minería de texto que se han visto en las clases de teoría (PAs).

Las tareas a realizar son:

1. Elegir una tarea (diferente de "text classification"). En el siguiente enlace, en la sección de "Natural Language Processing" se pueden inspeccionar diferentes tareas de minería de texto:
   https://huggingface.co/tasks

2. Elegir un dataset asociado a dicha tarea. En el siguiente enlace se pueden explorar datasets para diferentes tareas:
   https://huggingface.co/datasets

   La mayoría de datasets del hub de HuggingFace se han utilizado para entrenar grandes modelos. Para evitar manejar datasets grandes, se puede seleccionar un subconjunto de ellos o utilizar un dataset customizado (así también se evita el riesgo de overfitting).

   Otras alternativas:
   Kaggle NLP Datasets
   Datasets for Natural Language Processing - MachineLearningMastery.com

3. Elegir al menos 2 modelos para resolver la tarea elegida.

4. Evaluar sobre el dataset elegido y hacer una comparativa de los modelos. Cada modelo puede estar pre-entrenado con diferentes datasets, por lo que es de interés investigar el impacto tanto del modelo elegido como de los datos con los que fue entrenado.

   En https://huggingface.co/evaluate-metric se pueden comprobar las métricas asociadas a diferentes tareas.

   La evaluación en tareas generativas puede ser más compleja que en otras tareas. Por ejemplo, en clasificación la calidad del modelo se puede evaluar por el porcentaje de aciertos porque la etiqueta esperada es única, sin embargo, en un modelo de generación de texto puede haber varias respuestas válidas. Para esto hay métricas como la perplejidad o la BLEU score. Como alternativa a utilizar estas métricas, el alumno puede actuar como evaluador humano para discutir qué modelo es mejor.

5. Elegir el mejor modelo y crear una demo para desplegarlo.

Nota: la intención de esta práctica no es entrenar ningún modelo desde cero si no reaprovechar el conocimiento de modelos ya entrenados. La mayoría de estos modelos se ha entrenado sobre corpus de texto con miles de datos y tienen millones de parámetros. En consecuencia, es posible que tengan una demanda alta de RAM y también de GPU. Si fuera el caso, se recomienda utilizar alguna opción de cómputo en la nube como Google Colaboratory, AWS Sagemaker o Databricks.