Clasificación de Texto

Fecha de entrega: 01 de mayo de 2024

Problema a resolver: Desarrollar modelos de machine learning capaces de clasificar mensajes de Telegram en dos categorías: spam y ham.
Instrucciones:

1. **(+1 pt)** Investigación previa

    - **(+1 pt)** Investiga y escribe un breve resumen sobre los algoritmos SVM (Support Vector Machine), Decision Tree y Random Forest.

2. **(+1 pt)** Carga y exploración datos

    - **(+0.0 pts)** Carga el conjunto de datos que acompaña al PDF de esta práctica en un DataFrame de pandas.
    - **(+0.2 pts)** Observa las primeras filas del DataFrame para tener una idea general de los datos y su estructura.
    - **(+0.3 pts)** Utiliza `value_counts()` para verificar si el dataset está balanceado. Para obtener una mejor visualización del balance, usa además un gráfico de barras para mostrar estos datos.
    - **(+0.5 pts)** Al llegar a este paso, habrás notado que el dataset está desbalanceado. Investiga qué técnicas se utilizan para resolver este problema.

3. **(+2 pts)** Preparación de datos

    - **(+0.2 pts)** Limpia los datos eliminando caracteres especiales.
    - **(+0.2 pts)** Limpia los datos convirtiendo los textos a minúsculas para uniformidad.
    - **(+0.5 pts)** Divide los textos en tokens individuales con
    - **(+0.5 pts)** Elimina *stopwords* (palabras comunes que no aportan mucho significado al texto).
    - **(+0.5 pts)** Convierte los textos limpios en vectores numéricos utilizando CountVectorizer o TfidfVectorizer.
    - **(+0.1 pts)** Utiliza `train_test_split` para dividir el conjunto de datos en dos: un conjunto de entrenamiento (80%) y un conjunto de prueba (20%).

4. **(+3 pts)** Entrenamiento de diferentes modelos

   - **(+3 pts)** Entrena los modelos `LogisticRegression()` (logistic regression), `SVC()` (support vector machine),`DecisionTreeClassifier()` (decision tree) y `RandomForestClassifier()` (random forest) con el conjunto de entrenamiento. Configura, según tus criterios, los parámetros básicos de cada modelo (en caso de que así lo veas conveniente).
     Para compensar las clases desbalanceadas, por favor ajusta los pesos de las clases durante el entrenamiento con el parámetro `class_weight='balanced'` (todos los modelos que vamos a usar soportan ese parámetro). Esto le indica al modelo que debe prestarle más atención a la clase minoritaria (spoiler, es spam jsjs). Puedes usar cualquier otra técnica investigada para balancear el dataset, pero creo que esta modificación es más que suficiente.

5. **(+3 pts)** Evaluación de los modelos y reflexión uwu

   - **(+1 pts)** Evalúa los modelos utilizando varias métricas para comprender mejor su funcionamiento. Es importante que incluyas métricas como precisión, recall y el F1-socre (sobre todo porque el dataset está desbalanceado). Incluye también su respectiva matriz de confusión.
   - **(+2 pts)** Usando las evaluaciones, escribe tus conclusiones. Todos los integrantes del equipo deben redactar su conclusión.

Entregables:

- Código fuente del proyecto en un archivo `.ipynb`
- Reporte en `pdf` donde se explique la implementación, la experimentación y el análisis de los resultados. Esto también puede ir en el notebook.

Estos dos archivos deben de estar dentro un de `.zip` con el nombre `equipo`-practica08.

Sobre la entrega:

- La práctica se realizará en equipos de exactamente **5 personas**.
- Ante cualquier duda con la práctica, por favor envía un correo a **taniarubi@ciencias.unam.mx** o manda un mensajito por **telegram** :D
- **En caso de no seguir los lineamientos de entrega, no se calificará la práctica en cuestión.**
- En caso de detectarse copias entre equipos, se evaluará a ambas partes con cero sin realizar indagaciones sobre el asunto.
- En caso de detectarse el uso de IA generativa en la mayor parte de la realización de la práctica, se realizará una entrevista a todo el equipo para determinar la calificación final del trabajo.