Nombre: Gabriel Muñoz Marcelo Callisaya
CI: 9873103

DAT 261 - Procesamiento del lenguaje natural

Tarea 4.

Ver reseñas del debate vicepresidencial, clasificarlas, crear el modelo y determinar si el debate fue positivo o negativo.

Recolección de reseñas del debate:

Las reseñas se obtuvieron mediante búsquedas en X (Twitter) de conversaciones sobre el debate. Se realizaron búsquedas específicas con operadores avanzados para filtrar por fecha, idioma, y geolocalización centrada en Bolivia.

Se recopilaron un total de aproximadamente 66 tweets, etiquetados manualmente como positivo (1) o negativo (0). Estos resultados se almacenaron en un archivo CSV `tweets.csv` con columnas `text` (el contenido del tweet) y `label` (la clasificación dada).

Las búsquedas de X realizadas fueron:

```
"debate" since:2025-10-04_10:00:00_BOT until:2025-10-06_23:59:00_BOT
lang:es geocode:-17.0,-65.0,600km
```

```
"debate"          claro          since:2025-10-04_10:00:00_BOT
until:2025-10-06_23:59:00_BOT lang:es geocode:-17.0,-65.0,600km
```

```
"debate"          bien           since:2025-10-04_10:00:00_BOT
until:2025-10-06_23:59:00_BOT lang:es geocode:-17.0,-65.0,600km
```

```
"debate"        preparado        since:2025-10-04_10:00:00_BOT
until:2025-10-06_23:59:00_BOT lang:es geocode:-17.0,-65.0,600km
```

```
"debate"          listo          since:2025-10-04_10:00:00_BOT
until:2025-10-06_23:59:00_BOT lang:es geocode:-17.0,-65.0,600km
```

```
"debate"          capaz          since:2025-10-04_10:00:00_BOT
until:2025-10-06_23:59:00_BOT lang:es geocode:-17.0,-65.0,600km
```

```
"debate"        propuesta        since:2025-10-04_10:00:00_BOT
until:2025-10-06_23:59:00_BOT lang:es geocode:-17.0,-65.0,600km
```

Clasificación de las reseñas y creación del modelo:

La clasificación de las reseñas se realizó manualmente analizando el tono y contenido de cada tweet: aquellos que expresaban opiniones favorables, elogios o aspectos constructivos se etiquetaron como positivos (1), mientras que los críticos, negativos o decepcionantes se marcaron como negativos (0).

Para el modelo de clasificación, se crearon dos programas en Python utilizando Máquinas de Soporte Vectorial SVM para clasificación binaria.

Se emplearon librerías como `pandas` para manejar datos, `re` para expresiones regulares, `nltk` para procesamiento de lenguaje natural, `sklearn` para vectorización TF-IDF y el modelo SVM, y `numpy` para manipulaciones numéricas.

El CSV se cargó con pd.read_csv("tweets.csv"), separando textos y etiquetas. Se introdujo una categoría «neutra» para casos ambiguos: en el primer programa, si el porcentaje de positivos está entre 45% y 55%; en el segundo, si la probabilidad positiva está entre 0.45 y 0.55.

Se realiza el preprocesamiento con:

```
def limpiarTexto(texto):
    texto = texto.lower()
    texto = re.sub(r'[\W_]+', ' ', texto)
    texto = ' '.join([palabra for palabra in texto.split() if palabra
not in palabrasVacias])
    return texto
```

Y lo que hace es convertir el texto a minúsculas para estandarizar, eliminar caracteres no alfanuméricos con expresiones regulares para limpiar ruido, y remover palabras vacías (stopwords) usando NLTK para enfocarse en términos relevantes y reducir dimensionalidad.

Luego, se vectoriza el texto con TF-IDF:

```
vectorizador = TfidfVectorizer()
textosVectorizados = vectorizador.fit_transform(datos['text'])
```

Esta parte transforma los textos limpios en vectores numéricos ponderados por frecuencia e importancia (TF-IDF), lo que permite al modelo SVM trabajar en un espacio vectorial.

El modelo SVM se entrena con kernel lineal:

```
modelo = SVC(kernel='linear')
modelo.fit(textosVectorizados, datos['label'])
```

SVM encuentra un hiperplano que separa clases positivas y negativas maximizando el margen, usando coeficientes para identificar palabras influyentes.

Para el primer programa, se calculan porcentajes y palabras top basadas en coeficientes:

```
coeficientes = modelo.coef_.toarray()[0]
palabrasPositivas = [nombresCaracteristicas[i] for i in
np.argsort(coeficientes)[-10:][::-1]]
```

Esto extrae las 10 palabras con coeficientes más altos (positivas) y bajos (negativas), indicando su contribución a cada clase.

El segundo programa usa probabilidades para clasificar inputs nuevos:

```
probabilidadPositiva = modelo.predict_proba(textoNuevoVectorizado)[0]
[1]
if 0.45 < probabilidadPositiva < 0.55:
    resultado = "neutra"
```

Esto permite clasificaciones más matizadas, incluyendo neutral.

Ejecución de los progamas:

Programa uno (determinar si el debate fue positivo o negativo):

```
python3 svm.py
[nltk_data] Downloading package stopwords to /home/zinko/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Porcentaje de reseñas positivas: 40.91%
Porcentaje de reseñas negativas: 59.09%
La clasificación general del debate es: negativo
Las 10 palabras más positivas: ['bolivia', 'larazondigital', 'cabo',
'presidencial', 'claro', 'velasco', 'pasado', 'vicepresidencial',
'lara', 'tse']
Las 10 palabras más negativas: ['perdió', 'ambos', 'visto',
'propuestas', 'balotaje', 'desnudos', 'terminó', 'tahuichi', 'agenda',
'mediática']
```

Programa dos (clasificar nuevos inputs):

```
python3 svmInput.py
[nltk_data] Downloading package stopwords to /home/zinko/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Ingrese un texto nuevo para clasificar: el debate fue claro, velasco y
lara llevaron a cabo un debate vice presidencial correcto
La clasificación del texto es: positivo
```

```
python3 svmInput.py
[nltk_data] Downloading package stopwords to /home/zinko/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Ingrese un texto nuevo para clasificar: las propuestas fueron buenas
La clasificación del texto es: negativo
```

```
python3 svmInput.py
[nltk_data] Downloading package stopwords to /home/zinko/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
Ingrese un texto nuevo para clasificar: bolivia tuvo un horrendo debate,
lo que ha pasado es terrible
La clasificación del texto es: positivo
```

Conclusion

El trabajo demostró que el debate vicepresidencial fue percibido mayoritariamente como negativo (59.09% de reseñas negativas), con una clasificación general negativa.

Los resultados del modelo SVM fueron razonables para un conjunto de datos pequeño y equilibrado, identificando palabras clave asociadas a cada sentimiento.

Sin embargo, en pruebas con el segundo programa, una reseña claramente negativa como `"bolivia tuvo un horrendo debate, lo que ha pasado es terrible"` se clasificó como positiva debido a palabras como `"bolivia"` y `"pasado"` que el modelo asocia con positivos por su frecuencia en contextos favorables.

Esto resalta limitaciones de SVM: como modelo lineal, depende fuertemente de features TF-IDF y puede fallar en capturar contexto semántico o sarcasmo, siendo sensible a desbalances o datos ruidosos. Para mejoras, se podría usar embeddings más avanzados o modelos no lineales.