Análisis de sentimientos con Python

Adrián Yared Armas de la Nuez

Contenido

1. Actividad

1.2. Enunciado

Resuma el artículo a continuación, destacando los aspectos clave y las conclusiones principales:
https://towardsdatascience.com/sentiment-analysis-concept-analysis-and-applications-6c94d6f58c17

1.3. Resolución

El artículo explica la importancia del análisis de sentimientos, que es una técnica de minería de texto que permite a las empresas comprender la percepción social sobre sus marcas. El autor destaca que el análisis básico de sentimientos, como la clasificación de mensajes en positivos, negativos o neutrales, es superficial y pierde perspectivas más profundas que pueden ser extraídas con tecnologías avanzadas como el aprendizaje profundo y la inteligencia artificial.

Los aspectos claves principales a destacar según mi criterio son las siguientes; la definición de análisis de sentimientos; el análisi de intenciones, que va más allá del análisis de sentimientos, permitiendo identificar la intención del mensaje; la búsqueda css (semántica contextual), esto clasifica los mensajes según los contextos sin depender de palabras clave exactas; y finalmente el caso de Estudio y ejemplo de Uber, que realizó un análisis profundo de las redes sociales sobre los comentarios sobre Uber y sus determinadas categorías (Precios, seguridad, pagos, etc). Este análisis permitió a la empresa darse cuenta de problemas con cancelaciones y con el servicio que permitieron ponerles alerta y mejorar su servicio.

2. Actividad

2.2. Enunciado general

Verifique que la información mostrada en este artículo coincide con el artículo de la actividad R.1 de análisis de sentimientos:
https://www.analyticsvidhya.com/blog/2022/07/sentiment-analysis-using-python/

2.2.1.1 Enunciado actividad 1

Lleve a cabo un proceso de resumen, destacando las ideas principales y realice una comparación con el artículo del ejercicio anterior.
(Artículo 1)(Artículo 2)

2.2.1.2 Resolución

Este nuevo artículo explica cómo realizar análisis de sentimientos en Python, una técnica clave en el procesamiento de lenguaje natural para clasificar textos según sus sentimientos y se detallan diversas herramientas como; TextBlob, que es una librería que devuelve valores de polaridad y subjetividad; VADER, un analizador basado en el uso de las redes sociales; Modelos de aprendizaje automático; modelos basados en LSTM (Modelos de redes neuronales recurrentes para datos secuenciales); y modelos basados en transformaciones, como el uso de modelos pre entrenados

Ambos artículos abordan el análisis de sentimientos, pero con enfoques y profundidades diferentes:
El nuevo artículo presenta una explicación mucho más detallada sobre el uso de herramientas para el análisis de sentimientos, sin embargo el primer artículo muestra los principios básicos con menos profundidad técnica. Además, el segundo artículo muestra ejemplos detallados y un enfoque práctico basado principalmente en python, mientras que el artículo anterior menciona las aplicaciones generales de manera más abstracta resaltando su simplicidad y variedad de métodos.

2.2.2.1 Enunciado actividad 2

Adicionalmente, verifique el funcionamiento de los códigos de Python presentados en el artículo. Es fundamental que aumente el tamaño de la muestra en cada uno de los ejemplos para obtener resultados más consistentes.

2.2.2.2.2 Actividad del artículo 1

Instalación de textblob:

TextBlob

```
[ ] !pip install textblob
```

```
Requirement already satisfied: textblob in /usr/local/lib/python3.10/dist-packages (0.17.1)
Requirement already satisfied: nltk>=3.1 in /usr/local/lib/python3.10/dist-packages (from textblob) (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (2024.9.11)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk>=3.1->textblob) (4.66.5)
```

Código con más ejemplos:

```
from textblob import TextBlob # import TextBlob
# main.py
from textblob import TextBlob

# Definiendo una lista de ejemplos de texto
texts = [
```

```
    "The movie was so awesome.",         # positivo
    "The food here tastes terrible.",    # negativo
    "I love this place!",                # positivo
    "This is the worst service ever.",   # negativo
    "What a fantastic experience!",       # positivo
    "I'm so disappointed with my order.",# negativo
    "The view from the top is breathtaking.", # positivo
    "I hate waiting in long lines.",     # negativo
    "This book is a masterpiece.",        # positivo
    "I can't stand the noise in the city.", # negativo
    "The weather is beautiful today.",    # positivo
    "I feel so sad and alone.",           # negativo
    "This product exceeded my expectations!", # positivo
    "The customer service was unhelpful.", # negativo
    "The concert was absolutely thrilling!", # positivo
    "I'm frustrated with my computer.",   # negativo
    "I would recommend this restaurant to everyone!", # positivo
    "The film was a complete disaster.", # negativo
    "Her performance was mesmerizing.",    # positivo
    "I am not satisfied with the results." # negativo
]

# Inicializando listas para polaridad y subjetividad
polarities = []
subjectivities = []

# Procesando cada texto
for i, text in enumerate(texts):
    # Determinando la polaridad
    polarity = TextBlob(text).sentiment.polarity
    polarities.append(polarity)

    # Determinando la subjetividad
    subjectivity = TextBlob(text).sentiment.subjectivity
    subjectivities.append(subjectivity)

    print(f"Text {i+1}: '{text}'")
    print(f"Polarity: {polarity}, Subjectivity: {subjectivity}\n")

# Mostrando resultados finales
```

```
print("Polarities:", polarities)
print("Subjectivities:", subjectivities)
```

Resultado de ejecución con más ejemplos:

```
Text 1: 'The movie was so awesome.'
Polarity: 1.0, Subjectivity: 1.0

Text 2: 'The food here tastes terrible.'
Polarity: -1.0, Subjectivity: 1.0

Text 3: 'I love this place!'
Polarity: 0.625, Subjectivity: 0.6

Text 4: 'This is the worst service ever.'
Polarity: -1.0, Subjectivity: 1.0

Text 5: 'What a fantastic experience!'
Polarity: 0.5, Subjectivity: 0.9

Text 6: 'I'm so disappointed with my order.'
Polarity: -0.75, Subjectivity: 0.75

Text 7: 'The view from the top is breathtaking.'
Polarity: 0.75, Subjectivity: 0.75

Text 8: 'I hate waiting in long lines.'
Polarity: -0.42500000000000004, Subjectivity: 0.65

Text 9: 'This book is a masterpiece.'
Polarity: 0.0, Subjectivity: 0.0

Text 10: 'I can't stand the noise in the city.'
Polarity: 0.0, Subjectivity: 0.0

Text 11: 'The weather is beautiful today.'
Polarity: 0.85, Subjectivity: 1.0

Text 12: 'I feel so sad and alone.'
Polarity: -0.5, Subjectivity: 1.0

Text 13: 'This product exceeded my expectations!'
Polarity: 0.0, Subjectivity: 0.0

Text 14: 'The customer service was unhelpful.'
Polarity: 0.0, Subjectivity: 0.0
```

```
Text 15: 'The concert was absolutely thrilling!'
Polarity: 0.3125, Subjectivity: 1.0

Text 16: 'I'm frustrated with my computer.'
Polarity: -0.7, Subjectivity: 0.2

Text 17: 'I would recommend this restaurant to everyone!'
Polarity: 0.0, Subjectivity: 0.0

Text 18: 'The film was a complete disaster.'
Polarity: 0.1, Subjectivity: 0.4

Text 19: 'Her performance was mesmerizing.'
Polarity: 0.3, Subjectivity: 0.7

Text 20: 'I am not satisfied with the results.'
Polarity: -0.25, Subjectivity: 1.0

Polarities: [1.0, -1.0, 0.625, -1.0, 0.5, -0.75, 0.75, -0.42500000000000004, 0.0, 0.0, 0.85, -0.5, 0.0, 0.0, 0.3125, -0.7, 0.0, 0.1, 0.3, -0.25]
Subjectivities: [1.0, 1.0, 0.6, 1.0, 0.9, 0.75, 0.75, 0.65, 0.0, 0.0, 1.0, 1.0, 0.0, 0.0, 1.0, 0.2, 0.0, 0.4, 0.7, 1.0]
```

2.2.2.2.3 Actividad del artículo 2

Instalación de Vader:

```
VADER

[3] !pip install vaderSentiment

Collecting vaderSentiment
  Downloading vaderSentiment-3.3.2-py2.py3-none-any.whl.metadata (572 bytes)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from vaderSentiment) (2.32.3)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->vaderSentiment) (2024.8.30)
Downloading vaderSentiment-3.3.2-py2.py3-none-any.whl (125 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 126.0/126.0 kB 2.4 MB/s eta 0:00:00
Installing collected packages: vaderSentiment
Successfully installed vaderSentiment-3.3.2
```

Código con más ejemplos:

```python
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer

# Inicializando el analizador de sentimientos
sentiment = SentimentIntensityAnalyzer()

# Definiendo una lista de ejemplos de texto
texts = [
    "The book was a perfect balance between writing style and plot.", #
positivo
    "The pizza tastes terrible.",
# negativo
    "I absolutely loved the new movie!",                               #
positivo
    "The service was awful and very slow.",                            #
negativo
    "This is one of the best vacations I've ever had.",                #
positivo
```

```
    "I'm so unhappy with my experience here.",                          #
negativo
    "What a delightful surprise to find this cafe!",                    #
positivo
    "The noise outside is really annoying.",                            #
negativo
    "Her artwork is incredibly beautiful.",
# positivo
    "I'm frustrated with the long wait times.",                         #
negativo
    "The atmosphere in this restaurant is amazing.",                    #
positivo
    "This product did not meet my expectations.",                       #
negativo
    "The concert was unforgettable and magical.",                       #
positivo
    "I'm not satisfied with my purchase at all.",                       #
negativo
    "He gave an inspiring speech that motivated everyone.",             #
positivo
    "I feel very disappointed by the lack of options.",                 #
negativo
    "The new restaurant in town has delicious food!",                  #
positivo
    "I cannot recommend this hotel due to its poor cleanliness.",      #
negativo
    "The weather was perfect for a day at the beach.",                  #
positivo
    "I am very upset about the recent changes.",                        #
negativo
    "This book is a real page-turner and keeps you engaged!",          #
positivo
    "The quality of the service was subpar and not worth the price."   #
negativo
]

# Procesando cada texto y mostrando el sentimiento
for i, text in enumerate(texts):
    # Obteniendo los puntajes de polaridad
    sentiment_scores = sentiment.polarity_scores(text)
```

```
    print(f"Sentiment of text {i + 1}: '{text}' ->
{sentiment_scores}\n")
```

Resultado de ejecución con más ejemplos:

```
Sentiment of text 1: 'The book was a perfect balance between writing style and plot.' -> {'neg': 0.0, 'neu': 0.73, 'pos': 0.27, 'compound': 0.5719}
Sentiment of text 2: 'The pizza tastes terrible.' -> {'neg': 0.508, 'neu': 0.492, 'pos': 0.0, 'compound': -0.4767}
Sentiment of text 3: 'I absolutely loved the new movie!' -> {'neg': 0.0, 'neu': 0.527, 'pos': 0.473, 'compound': 0.6689}
Sentiment of text 4: 'The service was awful and very slow.' -> {'neg': 0.333, 'neu': 0.667, 'pos': 0.0, 'compound': -0.4588}
Sentiment of text 5: 'This is one of the best vacations I've ever had.' -> {'neg': 0.0, 'neu': 0.682, 'pos': 0.318, 'compound': 0.6369}
Sentiment of text 6: 'I'm so unhappy with my experience here.' -> {'neg': 0.34, 'neu': 0.66, 'pos': 0.0, 'compound': -0.4754}
Sentiment of text 7: 'What a delightful surprise to find this cafe!' -> {'neg': 0.0, 'neu': 0.492, 'pos': 0.508, 'compound': 0.7345}
Sentiment of text 8: 'The noise outside is really annoying.' -> {'neg': 0.374, 'neu': 0.626, 'pos': 0.0, 'compound': -0.4576}
Sentiment of text 9: 'Her artwork is incredibly beautiful.' -> {'neg': 0.0, 'neu': 0.488, 'pos': 0.512, 'compound': 0.6361}
Sentiment of text 10: 'I'm frustrated with the long wait times.' -> {'neg': 0.362, 'neu': 0.638, 'pos': 0.0, 'compound': -0.5267}
Sentiment of text 11: 'The atmosphere in this restaurant is amazing.' -> {'neg': 0.0, 'neu': 0.612, 'pos': 0.388, 'compound': 0.5859}
Sentiment of text 12: 'This product did not meet my expectations.' -> {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
Sentiment of text 13: 'The concert was unforgettable and magical.' -> {'neg': 0.0, 'neu': 1.0, 'pos': 0.0, 'compound': 0.0}
Sentiment of text 14: 'I'm not satisfied with my purchase at all.' -> {'neg': 0.25, 'neu': 0.75, 'pos': 0.0, 'compound': -0.3252}
Sentiment of text 15: 'He gave an inspiring speech that motivated everyone.' -> {'neg': 0.0, 'neu': 0.508, 'pos': 0.492, 'compound': 0.7003}
Sentiment of text 16: 'I feel very disappointed by the lack of options.' -> {'neg': 0.449, 'neu': 0.551, 'pos': 0.0, 'compound': -0.6901}
Sentiment of text 17: 'The new restaurant in town has delicious food!' -> {'neg': 0.0, 'neu': 0.637, 'pos': 0.363, 'compound': 0.6114}
Sentiment of text 18: 'I cannot recommend this hotel due to its poor cleanliness.' -> {'neg': 0.394, 'neu': 0.606, 'pos': 0.0, 'compound': -0.6381}
Sentiment of text 19: 'The weather was perfect for a day at the beach.' -> {'neg': 0.0, 'neu': 0.709, 'pos': 0.291, 'compound': 0.5719}
Sentiment of text 20: 'I am very upset about the recent changes.' -> {'neg': 0.292, 'neu': 0.708, 'pos': 0.0, 'compound': -0.4391}
Sentiment of text 21: 'This book is a real page-turner and keeps you engaged!' -> {'neg': 0.0, 'neu': 0.751, 'pos': 0.249, 'compound': 0.4574}
Sentiment of text 22: 'The quality of the service was subpar and not worth the price.' -> {'neg': 0.132, 'neu': 0.868, 'pos': 0.0, 'compound': -0.1695}
```

2.2.2.2.4 Actividad del artículo 3

Código para el análisis de sentimientos utilizando el enfoque de vectorización de bolsa de palabras

código:

```
#Loading the Dataset
import pandas as pd
data = pd.read_csv('data.csv')
#Pre-Prcoessing and Bag of Word Vectorization using Count Vectorizer
from sklearn.feature_extraction.text import CountVectorizer
from nltk.tokenize import RegexpTokenizer
token = RegexpTokenizer(r'[a-zA-Z0-9]+')
cv = CountVectorizer(stop_words='english',ngram_range = (1,1),tokenizer
= token.tokenize)
```

```
text_counts = cv.fit_transform(data['Sentence'])
#Splitting the data into trainig and testing
from sklearn.model_selection import train_test_split
X_train, X_test, Y_train, Y_test = train_test_split(text_counts,
data['Sentiment'], test_size=0.25, random_state=5)
#Training the model
from sklearn.naive_bayes import MultinomialNB
MNB = MultinomialNB()
MNB.fit(X_train, Y_train)
#Caluclating the accuracy score of the model
from sklearn import metrics
predicted = MNB.predict(X_test)
accuracy_score = metrics.accuracy_score(predicted, Y_test)
print("Accuracuy Score: ",accuracy_score)
```

Resultado:

```
Accuracuy Score:  0.6851471594798083
```

2.2.2.2.5 Actividad del artículo 4

Instalación Keras:

```
!pip install keras

Requirement already satisfied: keras in /usr/local/lib/python3.10/dist-packages (3.4.1)
Requirement already satisfied: absl-py in /usr/local/lib/python3.10/dist-packages (from keras) (1.4.0)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from keras) (1.26.4)
Requirement already satisfied: rich in /usr/local/lib/python3.10/dist-packages (from keras) (13.9.3)
Requirement already satisfied: namex in /usr/local/lib/python3.10/dist-packages (from keras) (0.0.8)
Requirement already satisfied: h5py in /usr/local/lib/python3.10/dist-packages (from keras) (3.11.0)
Requirement already satisfied: optree in /usr/local/lib/python3.10/dist-packages (from keras) (0.13.0)
Requirement already satisfied: ml-dtypes in /usr/local/lib/python3.10/dist-packages (from keras) (0.4.1)
Requirement already satisfied: packaging in /usr/local/lib/python3.10/dist-packages (from keras) (24.1)
Requirement already satisfied: typing-extensions>=4.5.0 in /usr/local/lib/python3.10/dist-packages (from optree->keras) (4.12.2)
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.10/dist-packages (from rich->keras) (3.0.0)
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.10/dist-packages (from rich->keras) (2.18.0)
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.10/dist-packages (from markdown-it-py>=2.2.0->rich->keras) (0.1.2)
```

código:

```
# Importing necessary libraries
import nltk
import pandas as pd
from textblob import Word
from nltk.corpus import stopwords
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import classification_report, confusion_matrix,
accuracy_score
from keras.models import Sequential
```

```
from tensorflow.keras.layers import LeakyReLU
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from keras.layers import Dense, Embedding, LSTM, SpatialDropout1D
from sklearn.model_selection import train_test_split

# Download the necessary stopwords dataset
nltk.download('stopwords')  # Downloads English stopwords
nltk.download('wordnet')  # Downloads WordNet lexicon for lemmatization
nltk.download('omw-1.4')  # Downloads additional WordNet lexicon

# Loading the dataset
data = pd.read_csv('data.csv')  # Loads the dataset from a CSV file

# Pre-Processing the text
def cleaning(df, stop_words):
    # Lowercasing the sentence
    df['Sentence'] = df['Sentence'].apply(lambda x: ' '.join(x.lower()
for x in x.split()))  # Converts to lowercase
    # Replacing digits/numbers
    df['Sentence'] = df['Sentence'].str.replace('\d+', '', regex=True)
# Removes digits from the text
    # Removing stop words
    df['Sentence'] = df['Sentence'].apply(lambda x: ' '.join(x for x in
x.split() if x not in stop_words))  # Removes stop words
    # Lemmatization
    df['Sentence'] = df['Sentence'].apply(lambda x: '
'.join([Word(word).lemmatize() for word in x.split()]))  # Applies
lemmatization to words
    return df  # Returns the processed DataFrame

# Get the list of stop words in English
stop_words = stopwords.words('english')
# Clean the dataset
data_cleaned = cleaning(data, stop_words)

# Generating Embeddings using tokenizer
tokenizer = Tokenizer(num_words=500, split=' ')
tokenizer.fit_on_texts(data_cleaned['Sentence'].values)
```

```
# Defining X and y
X = tokenizer.texts_to_sequences(data_cleaned['Sentence'].values)
X = pad_sequences(X)

# Convert labels to one-hot encoding
y = pd.get_dummies(data_cleaned['Sentiment']).values

# Model Building
model = Sequential()
model.add(Embedding(500, 120, input_length=X.shape[1]))
model.add(SpatialDropout1D(0.4))
model.add(LSTM(704, dropout=0.2, recurrent_dropout=0.2))
model.add(Dense(352))
model.add(LeakyReLU(negative_slope=0.1))
model.add(Dense(3, activation='softmax'))
model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])  # Compile
print(model.summary())

# Splitting the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

# Model Training
model.fit(X_train, y_train, epochs=20, batch_size=32, verbose=1)

# Model Testing
model.evaluate(X_test, y_test)
```

Resultado:

```
Non-trainable params: 0 (0.00 B)
None
Epoch 1/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 85s 551ms/step - accuracy: 0.5441 - loss: 1.0236
Epoch 2/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 84s 563ms/step - accuracy: 0.6350 - loss: 0.8185
Epoch 3/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 136s 520ms/step - accuracy: 0.6722 - loss: 0.7323
Epoch 4/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 81s 515ms/step - accuracy: 0.6996 - loss: 0.6697
Epoch 5/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 82s 515ms/step - accuracy: 0.7144 - loss: 0.6425
Epoch 6/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 76s 518ms/step - accuracy: 0.7234 - loss: 0.6228
Epoch 7/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 82s 516ms/step - accuracy: 0.7011 - loss: 0.6578
Epoch 8/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 83s 524ms/step - accuracy: 0.7335 - loss: 0.5791
Epoch 9/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 81s 518ms/step - accuracy: 0.7480 - loss: 0.5469
Epoch 10/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 83s 528ms/step - accuracy: 0.7589 - loss: 0.5203
Epoch 11/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 79s 508ms/step - accuracy: 0.7686 - loss: 0.5117
Epoch 12/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 84s 518ms/step - accuracy: 0.7605 - loss: 0.5055
Epoch 13/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 81s 514ms/step - accuracy: 0.7816 - loss: 0.4545
Epoch 14/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 83s 524ms/step - accuracy: 0.7430 - loss: 0.5478
Epoch 15/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 80s 513ms/step - accuracy: 0.7794 - loss: 0.4497
Epoch 16/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 88s 552ms/step - accuracy: 0.7885 - loss: 0.4426
Epoch 17/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 75s 509ms/step - accuracy: 0.7859 - loss: 0.4336
Epoch 18/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 85s 529ms/step - accuracy: 0.7899 - loss: 0.4443
Epoch 19/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 80s 515ms/step - accuracy: 0.7770 - loss: 0.4597
Epoch 20/20
147/147 ━━━━━━━━━━━━━━━━━━━━ 82s 512ms/step - accuracy: 0.8012 - loss: 0.4174
37/37 ━━━━━━━━━━━━━━━━━━━━ 7s 175ms/step - accuracy: 0.6724 - loss: 0.9244
[0.978830873966217, 0.6621043682098389]
```

2.2.2.2.6 Actividad del artículo 5

Instalación de transformers:

```
[6]  !pip install transformers

Requirement already satisfied: transformers in /usr/local/lib/python3.10/dist-packages (4.44.2)
Requirement already satisfied: filelock in /usr/local/lib/python3.10/dist-packages (from transformers) (3.16.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.23.2 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.24.7)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (1.26.4)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.10/dist-packages (from transformers) (24.1)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.10/dist-packages (from transformers) (2024.9.11)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: safetensors>=0.4.1 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.4.5)
Requirement already satisfied: tokenizers<0.20,>=0.19 in /usr/local/lib/python3.10/dist-packages (from transformers) (0.19.1)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.10/dist-packages (from transformers) (4.66.5)
Requirement already satisfied: fsspec>=2023.5.0 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.23.2->transformers) (2024.6.1)
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.10/dist-packages (from huggingface-hub<1.0,>=0.23.2->transformers) (4.12.2)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.4.0)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2.2.3)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->transformers) (2024.8.30)
```

Código con más ejemplos:

```
import transformers
```

```
from transformers import pipeline

# Inicializando el pipeline de análisis de sentimientos
sentiment_pipeline = pipeline("sentiment-analysis")

# Definiendo una lista de ejemplos de texto
data = [
    "It was the best of times.",                    # positivo
    "It was the worst of times.",                    # negativo
    "I absolutely love this place!",                # positivo
    "The service here is terrible.",                 # negativo
    "What an amazing experience!",                   # positivo
    "I'm so disappointed with the movie.",          # negativo
    "The view from the top is breathtaking.",       # positivo
    "I can't stand the noise from the street.",     # negativo
    "This book is a fantastic read!",                # positivo
    "The food was bland and uninspiring.",           # negativo
    "Her performance was outstanding!",              # positivo
    "I feel very let down by this product.",         # negativo
    "The weather is perfect for a picnic.",          # positivo
    "I'm really frustrated with the delay.",         # negativo
    "This restaurant serves the best pasta I've ever had!", # positivo
    "I'm unhappy with how my complaint was handled.", # negativo
```

```
    "The concert was exhilarating!",                  # positivo
    "I don't think I will return to this place.",    # negativo
    "The movie had a beautiful soundtrack.",         # positivo
    "I felt bored and restless during the presentation.", # negativo
    "This app has changed my life for the better!" # positivo
]

# Realizando el análisis de sentimientos
results = sentiment_pipeline(data)

# Imprimiendo los resultados
for i, text in enumerate(data):
    print(f"Text: '{text}' -> Sentiment: {results[i]}")
```

Resultado:

```
Text: 'It was the best of times.' -> Sentiment: {'label': 'POSITIVE', 'score': 0.999457061290741}
Text: 'It was the worst of times.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9997925162315369}
Text: 'I absolutely love this place!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.999887228012085}
Text: 'The service here is terrible.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9997105002403259}
Text: 'What an amazing experience!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9998835325241089}
Text: 'I'm so disappointed with the movie.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9998205304145813}
Text: 'The view from the top is breathtaking.' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9998657703399658}
Text: 'I can't stand the noise from the street.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.997144877910614}
Text: 'This book is a fantastic read!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.999881386756897}
Text: 'The food was bland and uninspiring.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9997547268867493}
Text: 'Her performance was outstanding!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9998682737350464}
Text: 'I feel very let down by this product.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9998169541358948}
Text: 'The weather is perfect for a picnic.' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9997969269752502}
Text: 'I'm really frustrated with the delay.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9996020197868347}
Text: 'This restaurant serves the best pasta I've ever had!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9997825026512146}
Text: 'I'm unhappy with how my complaint was handled.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9996727705001831}
Text: 'The concert was exhilarating!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9998378753662109}
Text: 'I don't think I will return to this place.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.9951905012130737}
Text: 'The movie had a beautiful soundtrack.' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9998812675476074}
Text: 'I felt bored and restless during the presentation.' -> Sentiment: {'label': 'NEGATIVE', 'score': 0.99974125623703}
Text: 'This app has changed my life for the better!' -> Sentiment: {'label': 'POSITIVE', 'score': 0.9719956517219543}
```

3. Enlace del colab