Hindi Sentiment Analysis using Transformers

This notebook demonstrates fine-tuning a transformer-based model on the Hindi sentiment dataset from AI4Bharat/IndicSentiment.

1. Install Required Libraries

We begin by installing necessary libraries such as `datasets` from Hugging Face.

```
pip install datasets
```

```
Collecting datasets
  Downloading datasets-3.5.0-py3-none-any.whl.metadata (19 kB)
Requirement already satisfied: filelock in
/usr/local/lib/python3.11/dist-packages (from datasets) (3.18.0)
Requirement already satisfied: numpy>=1.17 in
/usr/local/lib/python3.11/dist-packages (from datasets) (2.0.2)
Requirement already satisfied: pyarrow>=15.0.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Collecting dill<0.3.9,>=0.3.0 (from datasets)
  Downloading dill-0.3.8-py3-none-any.whl.metadata (10 kB)
Requirement already satisfied: pandas in
/usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.32.2 in
/usr/local/lib/python3.11/dist-packages (from datasets) (2.32.3)
Requirement already satisfied: tqdm>=4.66.3 in
/usr/local/lib/python3.11/dist-packages (from datasets) (4.67.1)
Collecting xxhash (from datasets)
  Downloading xxhash-3.5.0-cp311-cp311-
manylinux_2_17_x86_64.manylinux2014_x86_64.whl.metadata (12 kB)
Collecting multiprocess<0.70.17 (from datasets)
  Downloading multiprocess-0.70.16-py311-none-any.whl.metadata (7.2
kB)
Collecting fsspec<=2024.12.0,>=2023.1.0 (from
fsspec[http]<=2024.12.0,>=2023.1.0->datasets)
  Downloading fsspec-2024.12.0-py3-none-any.whl.metadata (11 kB)
Requirement already satisfied: aiohttp in
/usr/local/lib/python3.11/dist-packages (from datasets) (3.11.15)
Requirement already satisfied: huggingface-hub>=0.24.0 in
/usr/local/lib/python3.11/dist-packages (from datasets) (0.30.2)
Requirement already satisfied: packaging in
/usr/local/lib/python3.11/dist-packages (from datasets) (24.2)
Requirement already satisfied: pyyaml>=5.1 in
/usr/local/lib/python3.11/dist-packages (from datasets) (6.0.2)
Requirement already satisfied: aiohappyeyeballs>=2.3.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(2.6.1)
```

```
Requirement already satisfied: aiosignal>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(1.3.2)
Requirement already satisfied: attrs>=17.3.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(25.3.0)
Requirement already satisfied: frozenlist>=1.1.1 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(1.5.0)
Requirement already satisfied: multidict<7.0,>=4.5 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(6.4.3)
Requirement already satisfied: propcache>=0.2.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(0.3.1)
Requirement already satisfied: yarl<2.0,>=1.17.0 in
/usr/local/lib/python3.11/dist-packages (from aiohttp->datasets)
(1.19.0)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-hub>=0.24.0-
>datasets) (4.13.2)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (3.4.1)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (2.3.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests>=2.32.2-
>datasets) (2025.1.31)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets)
(2.8.2)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets)
(2025.2)
Requirement already satisfied: tzdata>=2022.7 in
/usr/local/lib/python3.11/dist-packages (from pandas->datasets)
(2025.2)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2-
>pandas->datasets) (1.17.0)
Downloading datasets-3.5.0-py3-none-any.whl (491 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 491.2/491.2 kB 10.3 MB/s eta
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 116.3/116.3 kB 9.6 MB/s eta
```

```
0:00:00
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 183.9/183.9 kB 6.5 MB/s eta
0:00:00
ultiprocess-0.70.16-py311-none-any.whl (143 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 143.5/143.5 kB 11.4 MB/s eta
0:00:00
anylinux_2_17_x86_64.manylinux2014_x86_64.whl (194 kB)
━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 194.8/194.8 kB 11.5 MB/s eta
0:00:00
ultiprocess, datasets
  Attempting uninstall: fsspec
    Found existing installation: fsspec 2025.3.2
    Uninstalling fsspec-2025.3.2:
      Successfully uninstalled fsspec-2025.3.2
ERROR: pip's dependency resolver does not currently take into account
all the packages that are installed. This behaviour is the source of
the following dependency conflicts.
gcsfs 2025.3.2 requires fsspec==2025.3.2, but you have fsspec
2024.12.0 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cublas-cu12==12.4.5.8;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cublas-cu12 12.5.3.2 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-cupti-cu12==12.4.127;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cuda-cupti-cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-nvrtc-cu12==12.4.127;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cuda-nvrtc-cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cuda-runtime-cu12==12.4.127;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cuda-runtime-cu12 12.5.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cudnn-cu12==9.1.0.70;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cudnn-cu12 9.3.0.75 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cufft-cu12==11.2.1.3;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cufft-cu12 11.2.3.61 which is incompatible.
torch 2.6.0+cu124 requires nvidia-curand-cu12==10.3.5.147;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-curand-cu12 10.3.6.82 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusolver-cu12==11.6.1.9;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cusolver-cu12 11.6.3.83 which is incompatible.
torch 2.6.0+cu124 requires nvidia-cusparse-cu12==12.3.1.170;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-cusparse-cu12 12.5.1.3 which is incompatible.
torch 2.6.0+cu124 requires nvidia-nvjitlink-cu12==12.4.127;
platform_system == "Linux" and platform_machine == "x86_64", but you
have nvidia-nvjitlink-cu12 12.5.82 which is incompatible.
```

```
Successfully installed datasets-3.5.0 dill-0.3.8 fsspec-2024.12.0
multiprocess-0.70.16 xxhash-3.5.0
```

2. Load Dataset

We use the Hindi subset of the IndicSentiment dataset.

```
from datasets import load_dataset
import pandas as pd

# Load the Hindi split with the correct config name
dataset = load_dataset("AI4Bharat/IndicSentiment", name="translation-
hi")

# Check available splits
print(dataset)
df = pd.DataFrame(dataset['validation'])
```

```
/usr/local/lib/python3.11/dist-packages/huggingface_hub/utils/
_auth.py:94: UserWarning:
The secret `HF_TOKEN` does not exist in your Colab secrets.
To authenticate with the Hugging Face Hub, create a token in your
settings tab (https://huggingface.co/settings/tokens), set it as
secret in your Google Colab and restart your session.
You will be able to reuse this secret in all of your notebooks.
Please note that authentication is recommended but still optional to
access public models or datasets.
  warnings.warn(
```


```
{"model_id":"01b6a76d435243b6a8c88163091bdc6d","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"83e2fa2dd5a041d5b84979d9c12eaeb8","version_major":2,"vers
ion_minor":0}
```


```
The repository for AI4Bharat/IndicSentiment contains custom code which
must be executed to correctly load the dataset. You can inspect the
repository content at https://hf.co/datasets/AI4Bharat/IndicSentiment.
You can avoid this prompt in future by passing the argument
`trust_remote_code=True`.

Do you wish to run the custom code? [y/N] y
```


```
{"model_id":"161c10bb1d2646db884c2af06b4de1c7","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"9f07fc466d194bbba3a82e2f25e8269e","version_major":2,"vers
ion_minor":0}
```

{"model_id":"49fdc355ba72460e9fad26936c7cdcee","version_major":2,"version_minor":0}

{"model_id":"e8ce4f2410f74cf29edfa956d564ca4e","version_major":2,"version_minor":0}

```
DatasetDict({
    validation: Dataset({
        features: ['GENERIC CATEGORIES', 'CATEGORY', 'SUB-CATEGORY', 'PRODUCT', 'BRAND', 'ASPECTS', 'ASPECT COMBO', 'ENGLISH REVIEW', 'LABEL', 'INDIC REVIEW'],
        num_rows: 156
    })
    test: Dataset({
        features: ['GENERIC CATEGORIES', 'CATEGORY', 'SUB-CATEGORY', 'PRODUCT', 'BRAND', 'ASPECTS', 'ASPECT COMBO', 'ENGLISH REVIEW', 'LABEL', 'INDIC REVIEW'],
        num_rows: 1000
    })
})
```

```
df
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 156,\n \"fields\": [\n   {\n   \"column\": \"GENERIC CATEGORIES\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 14,\n   \"samples\": [\n   \"Food\",\n   \"Building Material\",\n   \"Home\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"CATEGORY\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 24,\n   \"samples\": [\n   \"Movies\",\n   \"Personal Care\",\n   \"Appliances\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"SUB-CATEGORY\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 37,\n   \"samples\": [\n   \"Strollers and Prams\",\n   \"Railways\",\n   \"Fan\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"PRODUCT\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 91,\n   \"samples\": [\n   \"Cafe\",\n   \"cream biscuit and wafers\",\n   \"Fully voiced reading\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"BRAND\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 138,\n   \"samples\": [\n   \"Digicare\",\n   \"Wynk music\",\n   \"The Pets Company\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\":

\"ASPECTS\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 92,\n   \"samples\": [\n   \"Continental food and beverages, snacks, bakery items, quick service, online payments, ambience, music, parking, wifi, price\",\n   \"flavours, quantity, taste, nutrition\",\n   \"dress/shirt, soft breathable material, half sleeves/sleeveless, pet comfortable raincoat with detachable hood\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"ASPECT COMBO\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 143,\n   \"samples\": [\n   \"zipper, velcro closure\",\n   \"Punctuality Food services\",\n   \"World-class engineering, high-speed potential, worldwide dealership support and bike performance, average, and mileage\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"ENGLISH REVIEW\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 156,\n   \"samples\": [\n   \"It is particularly effective in providing moisture to irritated, itchy and painful skin rashes.\",\n   \"It provides intensive moisturizing and gives a nice heavenly fruit fragrance\",\n   \"Well equipped with and well maintained garden. Adults have good options to exercise while children have good choices to play with sea saw, swings, slides etc.\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"LABEL\",\n   \"properties\": {\n   \"dtype\": \"category\",\n   \"num_unique_values\": 2,\n   \"samples\": [\n   \"Positive\",\n   \"Negative\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"INDIC REVIEW\",\n   \"properties\": {\n   \"dtype\": \"string\",\n   \"num_unique_values\": 156,\n   \"samples\": [\n   \"\\u092f\\u0939 \\u0935\\u093f\\u0936\\u0947\\u0937 \\u0930\\u0942\\u092a\\u0938\\u0947 \\u0907\\u0930\\u093f\\u091f\\u0947\\u091f\\u0947\\u0921, \\u0907\\u091a\\u0940 \\u0914\\u0930 \\u092a\\u0947\\u0928\\u092b\\u0941\\u0932 \\u0938\\u094d\\u0915\\u093f\\u0928 \\u0930\\u0948\\u0936\\u0948\\u095b \\u0915\\u094b \\u092e\\u0949\\u0907\\u0938\\u094d\\u091a\\u0930 \\u092a\\u094d\\u0930\\u0926\\u093e\\u0928 \\u0915\\u0930\\u0928\\u0947 \\u092e\\u0947\\u0902 \\u092a\\u094d\\u0930\\u092d\\u093e\\u0935\\u0940 \\u0939\\u0948\\u0964\",\n   \"\\u092f\\u0939 \\u092c\\u095d\\u093f\\u092f\\u093e \\u092e\\u0949\\u0907\\u0938\\u094d\\u091a\\u0930\\u093e\\u0907\\u091c\\u093f\\u0902\\u0917 \\u092a\\u094d\\u0930\\u0926\\u093e\\u0928 \\u0915\\u0930\\u0924\\u093e \\u0939\\u0948 \\u0914\\u0930 \\u090f\\u0915 \\u0905\\u091a\\u094d\\u091b\\u0940 \\u092b\\u094d\\u0930\\u0942\\u091f \\u0916\\u0941\\u0936\\u092c\\u0942 \\u0926\\u0947\\u0924\\u093e \\u0939\\u0948\"\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
df.shape
```

```
(156, 10)
```

3. Prepare the DataFrame

We only keep the columns relevant for sentiment classification: the Hindi review and its label.

```
df = df[['INDIC REVIEW', 'LABEL']]
```

```
df
```


```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 156,\n  \"fields\": [\
n    {\n      \"column\": \"INDIC REVIEW\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 156,\n
\"samples\": [\n          \"\\u092f\\u0939 \\u0935\\u093f\\u0936\\
u0947\\u0937 \\u0930\\u0942\\u092a \\u0938\\u0947 \\u0907\\u0930\\
u093f\\u091f\\u0947\\u091f\\u0947\\u0921, \\u0907\\u091a\\u0940 \\
u0914\\u0930 \\u092a\\u0947\\u0928\\u092b\\u0941\\u0932 \\u0938\\
u094d\\u0915\\u093f\\u0928 \\u0930\\u0948\\u0936\\u0948\\u095b \\
u0915\\u094b \\u092e\\u0949\\u0907\\u0938\\u094d\\u091a\\u0930 \\
u092a\\u094d\\u0930\\u0926\\u093e\\u0928 \\u0915\\u0930\\u0928\\u0947
\\u092e\\u0947\\u0902 \\u092a\\u094d\\u0930\\u092d\\u093e\\u0935\\
u0940 \\u0939\\u0948\\u0964\",\n          \"\\u092f\\u0939 \\u092c\\
u095d\\u093f\\u092f\\u093e \\u092e\\u0949\\u0907\\u0938\\u094d\\
u091a\\u0930\\u093e\\u0907\\u091c\\u093f\\u0902\\u0917 \\u092a\\
u094d\\u0930\\u0926\\u093e\\u0928 \\u0915\\u0930\\u0924\\u093e \\
u0939\\u0948 \\u0914\\u0930 \\u090f\\u0915 \\u0905\\u091a\\u094d\\
u091b\\u0940 \\u092b\\u094d\\u0930\\u0942\\u091f \\u0916\\u0941\\
u0936\\u092c\\u0942 \\u0926\\u0947\\u0924\\u093e \\u0939\\u0948\",\n
\"\\u0917\\u093e\\u0930\\u094d\\u0921\\u0928 \\u092e\\u0947\\u0902 \\
u0938\\u092d\\u0940 \\u0938\\u0941\\u0935\\u093f\\u0927\\u093e \\
u0914\\u0930 \\u0907\\u0938\\u0947 \\u0905\\u091a\\u0947 \\u0938\\
u0947 \\u092e\\u0947\\u0902\\u091f\\u0947\\u0928 \\u0915\\u0930\\u093e
\\u0917\\u092f\\u093e \\u0939\\u0941\\u0906 \\u0939\\u0948\\u0964 \\
u0935\\u092f\\u0938\\u094d\\u0915\\u094b\\u0902 \\u0915\\u0947 \\
u092a\\u093e\\u0938 \\u0935\\u094d\\u092f\\u093e\\u092f\\u093e\\u092e
\\u0915\\u0930\\u0928\\u0947 \\u0915\\u0947 \\u0932\\u093f\\u090f \\
u0905\\u091a\\u094d\\u091b\\u0947 \\u0935\\u093f\\u0915\\u0932\\
u094d\\u092a \\u0939\\u0948\\u0902 \\u091c\\u092c\\u0915\\u093f \\
u092c\\u091a\\u094d\\u091a\\u094b\\u0902 \\u0915\\u0947 \\u092a\\
u093e\\u0938 \\u0938\\u0940-\\u0938\\u0949, \\u0938\\u094d\\u0935\\
u093f\\u0902\\u0917\\u094d\\u0938, \\u0938\\u094d\\u0932\\u093e\\
u0907\\u0921\\u094d\\u0938 \\u0906\\u0926\\u093f \\u0915\\u0947 \\
u0938\\u093e\\u0925 \\u0916\\u0947\\u0932\\u0928\\u0947 \\u0915\\u0947
\\u0932\\u093f\\u090f \\u0905\\u091a\\u094d\\u091b\\u0947 \\u0935\\
u093f\\u0915\\u0932\\u094d\\u092a \\u0939\\u0948\\u0902\\u0964\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"LABEL\",\n      \"properties\":
```

```
{\n        \"dtype\": \"category\",\n        \"num_unique_values\": 2,\n        \"samples\": [\n          \"Positive\",\n          \"Negative\"\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```

```
# kaggel dataset
import kagglehub

# Download latest version
path = kagglehub.dataset_download("maheshmj007/hindi-language-sentiment-dataset")

print("Path to dataset files:", path)
```

```
Path to dataset files: /kaggle/input/hindi-language-sentiment-dataset
```

```
import os

os.listdir("/kaggle/input/hindi-language-sentiment-dataset")
```

```
['hindi sentiment analysis.csv']
```

```
kaggle_df = pd.read_csv("/kaggle/input/hindi-language-sentiment-dataset/hindi sentiment analysis.csv")

kaggle_df.columns
```

```
Index(['लोग वतन तक खा जाते हैं इसका इसे यकीन नहींमान जाएगा तू ले जाकर दिल्ली इसे दिखा ला दोस्त', 'negative'], dtype='object')
```

```
df = df.rename(columns={'INDIC REVIEW': 'review', 'LABEL':'sentiment'})

df.sentiment.value_counts()
```

```
sentiment
Negative    81
Positive    75
Name: count, dtype: int64
```

```
kaggle_df.rename(columns={'लोग वतन तक खा जाते हैं इसका इसे यकीन नहींमान जाएगा तू ले जाकर दिल्ली इसे दिखा ला दोस्त': 'review', 'negative': 'sentiment'}, inplace = True)

kaggle_df.sentiment.value_counts()
```

```
sentiment
positive    3254
negative    3173
neutral     2649
Name: count, dtype: int64
```

```
kaggle_df.shape
```

```
(9076, 2)
```

```
df.shape
```

```
(156, 2)
```

```
df1 = df.merge(kaggle_df, how = 'outer')
```

```
df1.shape
```

```
(9232, 2)
```

```
df1 = df1.merge(pd.read_excel('train.xlsx')[['content_hindi',
'labels']].rename(columns={'content_hindi': 'review',
'labels':'sentiment'}), how = 'outer')

df1 = df1.merge(pd.read_excel('test.xlsx')[['content_hindi',
'labels']].rename(columns={'content_hindi': 'review',
'labels':'sentiment'}), how = 'outer')
```

```
df1.sentiment.value_counts()
```

```
sentiment
positive    3254
negative    3173
neutral     2649
Negative    2001
Positive    1670
Neutral      830
Name: count, dtype: int64
```

```
df1.sentiment = df1.sentiment.apply(lambda x: x.lower())
```

```
df1.sentiment.value_counts()
```

```
sentiment
negative    5174
positive    4924
neutral     3479
Name: count, dtype: int64
```

```
# importing libs
from datasets import load_dataset, DatasetDict, ClassLabel, Dataset

import transformers
from transformers import AutoTokenizer,
AutoModelForSequenceClassification, TrainingArguments, Trainer
```

```
df1
```

{"summary":"{\n \"name\": \"df1\",\n \"rows\": 13577,\n \"fields\":
[\n   {\n   \"column\": \"review\",\n   \"properties\": {\n
\"dtype\": \"string\",\n   \"num_unique_values\": 11957,\n
\"samples\": [\n   \"\\u0906\\u0924\\u0902\\u0915 '\\u0930\\
u094b\\u091c\\u093c\\u0917\\u093e\\u0930' \\u0939\\u0948 !\\n\\u0930\\
u093e\\u0939\\u0941\\u0932 \\u0917\\u093e\\u0902\\u0927\\u0940 \\
u0938\\u0947 \\u0938\\u092e\\u091d\\u093f\\u090f \\u0906\\u0924\\
u0902\\u0915 \\u0915\\u093e '\\u0905\\u0930\\u094d\\u0925\\u0936\\
u093e\\u0938\\u094d\\u0924\\u094d\\u0930'\\u0964 6 \\u092c\\u091c\\
u0947 \\u091f\\u0915\\u094d\\u0915\\u0930, \\u0939\\u0932\\u094d\\
u0932\\u093e \\u092c\\u094b\\u0932 \\u092e\\u0947\\u0902 @sambitswaraj
Vs Rajiv T\\u2026 https://t.co/hCgXfbKrth\",\n   \"\\u092f\\
u0939 \\u092c\\u0948\\u0932 \\u0905\\u0921\\u093c\\u093f\\u092f\\u0932
\\u0939\\u0948, \\u0916\\u0947\\u0924 \\u0915\\u0940 \\u091c\\u0941\\
u0924\\u093e\\u0908 \\u0915\\u0930\\u0924\\u0947 \\u0938\\u092e\\u092f
\\u092c\\u093e\\u0930-\\u092c\\u093e\\u0930 \\u0905\\u0921\\u093c \\
u091c\\u093e\\u0924\\u093e \\u0939\\u0948\",\n   \"\\u092d\\
u093e\\u0930\\u0924\\u0940\\u092f \\u091f\\u0940\\u092e \\u090f\\u0915
\\u092c\\u0947\\u0939\\u0926 \\u0939\\u0940 \\u0915\\u0920\\u093f\\
u0928 \\u0926\\u094c\\u0930\\u0947 \\u0915\\u0947 \\u0932\\u093f\\
u090f \\u0907\\u0902\\u0917\\u094d\\u0932\\u0948\\u0902\\u0921 \\
u0917\\u0908 \\u0925\\u0940\\u0964\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   },\n   {\n   \"column\": \"sentiment\",\n
\"properties\": {\n   \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n   \"samples\": [\n
\"negative\",\n   \"positive\",\n   \"neutral\"\n
],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n
}\n   }\n]\n}","type":"dataframe","variable_name":"df1"}

```
dataset = Dataset.from_pandas(df1)
df_dict = DatasetDict({'train': dataset})

df_dict
```

```
DatasetDict({
    train: Dataset({
        features: ['review', 'sentiment'],
        num_rows: 13577
    })
})
```

```
cl = ClassLabel(num_classes=3, names=["neutral", "positive",
"negative"])

df1
```

{"summary":"{\n \"name\": \"df1\",\n \"rows\": 13577,\n \"fields\":
[\n   {\n   \"column\": \"review\",\n   \"properties\": {\n
\"dtype\": \"string\",\n   \"num_unique_values\": 11957,\n
\"samples\": [\n   \"\\u0906\\u0924\\u0902\\u0915 '\\u0930\\

```
u094b\\u091c\\u093c\\u0917\\u093e\\u0930' \\u0939\\u0948 !\\n\\u0930\\
u093e\\u0939\\u0941\\u0932 \\u0917\\u093e\\u0902\\u0927\\u0940 \\
u0938\\u0947 \\u0938\\u092e\\u091d\\u093f\\u090f \\u0906\\u0924\\
u0902\\u0915 \\u0915\\u093e '\\u0905\\u0930\\u094d\\u0925\\u0936\\
u093e\\u0938\\u094d\\u0924\\u094d\\u0930'\\u0964 6 \\u092c\\u091c\\
u0947 \\u091f\\u0915\\u094d\\u0915\\u0930, \\u0939\\u0932\\u094d\\
u0932\\u093e \\u092c\\u094b\\u0932 \\u092e\\u0947\\u0902 @sambitswaraj
Vs Rajiv T\\u2026 https://t.co/hCgXfbKrth\",\n          \"\\u092f\\
u0939 \\u092c\\u0948\\u0932 \\u0905\\u0921\\u093c\\u093f\\u092f\\u0932
\\u0939\\u0948, \\u0916\\u0947\\u0924 \\u0915\\u0940 \\u091c\\u0941\\
u0924\\u093e\\u0908 \\u0915\\u0930\\u0924\\u0947 \\u0938\\u092e\\u092f
\\u092c\\u093e\\u0930-\\u092c\\u093e\\u0930 \\u0905\\u0921\\u093c \\
u091c\\u093e\\u0924\\u093e \\u0939\\u0948\",\n          \"\\u092d\\
u093e\\u0930\\u0924\\u0940\\u092f \\u091f\\u0940\\u092e \\u090f\\u0915
\\u092c\\u0947\\u0939\\u0926 \\u0939\\u0940 \\u0915\\u0920\\u093f\\
u0928 \\u0926\\u094c\\u0930\\u0947 \\u0915\\u0947 \\u0932\\u093f\\
u090f \\u0907\\u0902\\u0917\\u094d\\u0932\\u0948\\u0902\\u0921 \\
u0917\\u0908 \\u0925\\u0940\\u0964\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"sentiment\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"negative\",\n          \"positive\",\n          \"neutral\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    }\n  ]\n}","type":"dataframe","variable_name":"df1"}

import re
import pandas as pd

def clean_text_preserve_hindi_only(text):
    # Remove hyperlinks
    text = re.sub(r'http\S+|www.\S+', '', text)

    # Remove all English letters/words
    text = re.sub(r'[a-zA-Z]', '', text)

    # Remove unwanted special characters
    # Keep: Hindi (\u0900-\u097F), numbers, spaces, punctuation,
emojis
    text = re.sub(r'[^\s\u0900-\u097F0-9.,!?😀-]', '', text)

    return text
df1['review'] =
df1['review'].astype(str).apply(clean_text_preserve_hindi_only)

# import torch

# torch.device("cuda" if torch.cuda.is_available() else "cpu")
```

```
# from transformers import pipeline
# translator = pipeline("translation_en_to_hi",
model="Helsinki-NLP/opus-mt-en-hi")

# from transformers import pipeline

# translator = pipeline("translation_en_to_hi",
model="Helsinki-NLP/opus-mt-en-hi", device=-1)  # CPU-safe

# # Use the same cleaning & selective translation log

# import re
# import pandas as pd
# from transformers import pipeline

# # Load translator (English to Hindi)
# translator = pipeline("translation_en_to_hi",
model="Helsinki-NLP/opus-mt-en-hi",device=-1)

# # Function to clean the text (preserve Hindi, emojis, etc.)
# def clean_text_preserve_hindi(text):
#     # Remove hyperlinks
#     text = re.sub(r'http\S+|www.\S+', '', text)
#     # Remove unwanted special characters (keep Hindi, English,
numbers, punctuation, emojis)
#     text = re.sub(r'[^\w\s\u0900-\u097F.,!?😀-]', '', text)
#     return text

# # Function to detect English and translate if needed
# def translate_if_english(text):
#     text = str(text)  # Ensure it's a string
#     cleaned_text = clean_text_preserve_hindi(text)
#     if re.search(r'[a-zA-Z]', cleaned_text):  # Check if English is
present
#         try:
#             result = translator(cleaned_text, max_length=512)
#             return result[0]['translation_text']
#         except:
#             return cleaned_text  # fallback
#     return cleaned_text  # No English → just cleaned

# # Apply it to the DataFrame

# df1['review'].apply(translate_if_english)

# df1['review'] =
df1['review'].astype(str).apply(clean_text_preserve_hindi_only)
df1
```


```
{"summary":"{\n  \"name\": \"df1\",\n  \"rows\": 13577,\n  \"fields\":
[\n    {\n      \"column\": \"review\",\n      \"properties\": {\n
```

\"dtype\": \"string\",\n   \"num_unique_values\": 11900,\n
\"samples\": [\n   \"\\u092e\\u0948\\u0902\\u0928\\u0947 \\
u092f\\u0939 \\u092c\\u094d\\u0930\\u0936 02\\u092e\\u0908 2022 \\
u092e\\u0947\\u0902 \\u0916\\u0930\\u0940\\u0926\\u093e \\u0925\\
u093e, \\u0914\\u0930 \\u0905\\u092c, \\u0932\\u0917\\u092d\\u0917 \\
u090f\\u0915 \\u0935\\u0930\\u094d\\u0937 \\u0939\\u094b \\u0917\\
u092f\\u093e \\u0939\\u0948\\u0964 \\u0914\\u0930 \\u0924\\u092c \\
u0938\\u0947, \\u092e\\u0948\\u0902\\u0928\\u0947 \\u0907\\u0938 \\
u091a\\u0940\\u091c\\u093c \\u0915\\u094b \\u0926\\u094b \\u092c\\
u093e\\u0930 \\u092c\\u0926\\u0932\\u093e \\u0939\\u0948, \\u0914\\
u0930 \\u092f\\u0939 \\u092e\\u0947\\u0930\\u093e \\u0924\\u0940\\
u0938\\u0930\\u093e \\u0939\\u0948 \\u091c\\u093f\\u0938\\u0947 \\
u092e\\u0948\\u0902 \\u0938\\u093e\\u0935\\u0927\\u093e\\u0928\\u0940
\\u0938\\u0947 \\u0909\\u092a\\u092f\\u094b\\u0917 \\u0915\\u0930 \\
u0930\\u0939\\u093e \\u0939\\u0942\\u0902\\u0964 \\u092c\\u094d\\
u0930\\u0936 \\u0905\\u091a\\u094d\\u091b\\u093e \\u0939\\u0948 \\
u092e\\u0948\\u0902\\u0928\\u0947 \\u0907\\u0938\\u0915\\u0947 \\
u092c\\u093e\\u0930\\u0947 \\u092e\\u0947\\u0902 \\u0935\\u093f\\
u0938\\u094d\\u0924\\u093e\\u0930 \\u0938\\u0947 \\u092c\\u0939\\
u0941\\u0924 \\u0915\\u0941\\u091b \\u0932\\u093f\\u0916\\u093e \\
u0939\\u0948 \\u0932\\u0947\\u0915\\u093f\\u0928 \\u092f\\u0939\\
u093e\\u0902 \\u092a\\u094b\\u0938\\u094d\\u091f \\u0928\\u0939\\
u0940\\u0902 \\u0915\\u0930 \\u0930\\u0939\\u093e \\u0939\\u0942\\
u0902 \\u0915\\u094d\\u092f\\u094b\\u0902\\u0915\\u093f \\u091c\\u092c
\\u0909\\u0924\\u094d\\u092a\\u093e\\u0926 \\u0935\\u093f\\u0936\\
u094d\\u0935\\u0938\\u0928\\u0940\\u092f \\u0928\\u0939\\u0940\\u0902
\\u0939\\u0948 \\u0924\\u094b \\u0907\\u0938\\u0938\\u0947 \\u0915\\
u094b\\u0908 \\u092b\\u0930\\u094d\\u0915 \\u0928\\u0939\\u0940\\u0902
\\u092a\\u0921\\u093c\\u0924\\u093e\\u0964 \\u090f\\u0915 \\u0907\\
u0932\\u0947\\u0915\\u094d\\u091f\\u094d\\u0930\\u093f\\u0915 \\
u091f\\u0942\\u0925\\u092c\\u094d\\u0930\\u0936 \\u091c\\u0932\\
u0930\\u094b\\u0927\\u0915 \\u0928\\u0939\\u0940\\u0902 \\u0939\\
u0948, \\u091c\\u094b \\u092e\\u0947\\u0930\\u0947 \\u0932\\u093f\\
u090f \\u0939\\u093e\\u0938\\u094d\\u092f\\u093e\\u0938\",\n
\"\\u092f\\u0939 \\u092c\\u0939\\u0941\\u0924 \\u0905\\u091a\\u094d\\
u091b\\u093e \\u0939\\u0948 \\u092e\\u0948\\u0902 \\u0905\\u092d\\
u0940 \\u092d\\u0940 \\u0907\\u0938\\u0947 2024 \\u0924\\u0915 \\
u092e\\u091c\\u092c\\u0942\\u0924\\u0940 \\u0938\\u0947 \\u0909\\
u092a\\u092f\\u094b\\u0917 \\u0915\\u0930 \\u0930\\u0939\\u093e \\
u0939\\u0942\\u0902\\u0964 2021 \\u092e\\u0947\\u0902 \\u0916\\u0930\\
u0940\\u0926\\u093e \\u0917\\u092f\\u093e\",\n   \"\\u092e\\
u0948\\u0902 \\u0916\\u0941\\u0936 \\u0928\\u0939\\u0940\\u0902 \\
u0939\\u0942\\u0902... \\u0915\\u094d\\u092f\\u094b\\u0902\\u0915\\
u093f \\u0907\\u0938\\u0915\\u0940 \\u0917\\u0941\\u0923\\u0935\\
u0924\\u094d\\u0924\\u093e \\u092c\\u0939\\u0941\\u0924 \\u0916\\
u0930\\u093e\\u092c \\u0939\\u0948, \\u0915\\u0941\\u091b \\u091a\\
u093e\\u092c\\u093f\\u092f\\u093e\\u0902 \\u0915\\u093e\\u092e \\
u0928\\u0939\\u0940\\u0902 \\u0915\\u0930\\u0924\\u0940 \\
u0939\\u0948\\u0902\\u0964\"\n   ],\n   \"semantic_type\":

```
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"sentiment\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 3,\n
\"samples\": [\n          \"negative\",\n          \"positive\",\n
\"neutral\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df1"}
```

```
dataset = Dataset.from_pandas(df1)
```

```
dataset
```

```
Dataset({
    features: ['review', 'sentiment'],
    num_rows: 13577
})
```

```
dd = DatasetDict({'train': dataset})
```

```
# from transformers import train_test_split
```

```
d1 = dd['train'].train_test_split(test_size = 0.2)
```

```
d1
```

```
DatasetDict({
    train: Dataset({
        features: ['review', 'sentiment'],
        num_rows: 10861
    })
    test: Dataset({
        features: ['review', 'sentiment'],
        num_rows: 2716
    })
})
```

```
d2 = d1['train'].train_test_split(test_size = 0.2)
```

```
main_data = DatasetDict({
    'train': d2['train'],
    'validation': d2['test'],
    'test': d1['test']
})
```

```
main_data
```

```
DatasetDict({
    train: Dataset({
        features: ['review', 'sentiment'],
        num_rows: 8688
    })
    validation: Dataset({
```

```
        features: ['review', 'sentiment'],
        num_rows: 2173
    })
    test: Dataset({
        features: ['review', 'sentiment'],
        num_rows: 2716
    })
})

# Load model directly
from transformers import AutoTokenizer,
AutoModelForSequenceClassification

tokenizer = AutoTokenizer.from_pretrained("google-bert/bert-base-
multilingual-cased")
model = AutoModelForSequenceClassification.from_pretrained("google-
bert/bert-base-multilingual-cased", num_labels=3)
```

{"model_id":"18f3ce61fb81415394b3728bae4c09f8","version_major":2,"vers
ion_minor":0}

{"model_id":"8b638707fd6745f084c8d586ab20d942","version_major":2,"vers
ion_minor":0}

{"model_id":"5cf1bf0208fb4394abd23cddc39d222c","version_major":2,"vers
ion_minor":0}

{"model_id":"f04cd972729549c8bb859f83ad376293","version_major":2,"vers
ion_minor":0}

```
Xet Storage is enabled for this repo, but the 'hf_xet' package is not
installed. Falling back to regular HTTP download. For better
performance, install the package with: `pip install
huggingface_hub[hf_xet]` or `pip install hf_xet`
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this
repo, but the 'hf_xet' package is not installed. Falling back to
regular HTTP download. For better performance, install the package
with: `pip install huggingface_hub[hf_xet]` or `pip install hf_xet`
```

{"model_id":"718b2e7f4b5448faa1e7717a675a97c9","version_major":2,"vers
ion_minor":0}

```
Some weights of BertForSequenceClassification were not initialized
from the model checkpoint at google-bert/bert-base-multilingual-cased
and are newly initialized: ['classifier.bias', 'classifier.weight']
You should probably TRAIN this model on a down-stream task to be able
to use it for predictions and inference.

def preproc(batch):
  l = []
  # FV
```

```
  for i in batch['review']:
    if i is not None:
      l.append(str(i))
    else:
      l.append("")

  token_batch = tokenizer(l, max_length=512, truncation=True,
padding='max_length')

  l1 = []
  #cL
  for i in batch['sentiment']:
    if i is not None:
      l1.append(cl.str2int(i))
    else:
      l1.append(0)
  token_batch['label'] = l1

  return token_batch

main_data = main_data.map(preproc, batched=True,
remove_columns=['review', 'sentiment'])
```


```
{"model_id":"d6c68c23aacc43318c3451a3f8ad2ce2","version_major":2,"vers
ion_minor":0}

{"model_id":"47059243c0864021bd096928cc6f2f63","version_major":2,"vers
ion_minor":0}

{"model_id":"bd6d0ae1a48048478220b6f3ff247f2c","version_major":2,"vers
ion_minor":0}
```


```
model

BertForSequenceClassification(
  (bert): BertModel(
    (embeddings): BertEmbeddings(
      (word_embeddings): Embedding(119547, 768, padding_idx=0)
      (position_embeddings): Embedding(512, 768)
      (token_type_embeddings): Embedding(2, 768)
      (LayerNorm): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
      (dropout): Dropout(p=0.1, inplace=False)
    )
    (encoder): BertEncoder(
      (layer): ModuleList(
        (0-11): 12 x BertLayer(
          (attention): BertAttention(
            (self): BertSdpaSelfAttention(
              (query): Linear(in_features=768, out_features=768,
bias=True)
```

```
              (key): Linear(in_features=768, out_features=768,
bias=True)
              (value): Linear(in_features=768, out_features=768,
bias=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
            (output): BertSelfOutput(
              (dense): Linear(in_features=768, out_features=768,
bias=True)
              (LayerNorm): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
              (dropout): Dropout(p=0.1, inplace=False)
            )
          )
          (intermediate): BertIntermediate(
            (dense): Linear(in_features=768, out_features=3072,
bias=True)
            (intermediate_act_fn): GELUActivation()
          )
          (output): BertOutput(
            (dense): Linear(in_features=3072, out_features=768,
bias=True)
            (LayerNorm): LayerNorm((768,), eps=1e-12,
elementwise_affine=True)
            (dropout): Dropout(p=0.1, inplace=False)
          )
        )
      )
    )
    (pooler): BertPooler(
      (dense): Linear(in_features=768, out_features=768, bias=True)
      (activation): Tanh()
    )
  )
  (dropout): Dropout(p=0.1, inplace=False)
  (classifier): Linear(in_features=768, out_features=3, bias=True)
)

from transformers import TrainingArguments, Trainer

ta = TrainingArguments(
    output_dir="/content/saved_model",
    eval_strategy="epoch",
    save_strategy="epoch",
    save_total_limit=1,
    load_best_model_at_end=True,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=3,
    learning_rate=2e-5,
```

```
    logging_dir="/content/saved_model",
    metric_for_best_model="eval_loss"
)

from transformers import EarlyStoppingCallback

early_stopper = EarlyStoppingCallback(early_stopping_patience=2)

tr = Trainer(
    model=model,
    args=ta,
    train_dataset=main_data['train'],
    eval_dataset=main_data['validation'],
    tokenizer=tokenizer,
    callbacks=[early_stopper]
)
```

```
<ipython-input-54-c5ae2b6fa414>:1: FutureWarning: `tokenizer` is
deprecated and will be removed in version 5.0.0 for
`Trainer.__init__`. Use `processing_class` instead.
  tr = Trainer(
```

```
model = tr.train()
```

```
wandb: WARNING The `run_name` is currently set to the same value as
`TrainingArguments.output_dir`. If this was not intended, please
specify a different run name by setting the
`TrainingArguments.run_name` parameter.
wandb: Using wandb-core as the SDK backend.  Please refer to
https://wandb.me/wandb-core for more information.
```

```
<IPython.core.display.Javascript object>
```

```
wandb: Logging into wandb.ai. (Learn how to deploy a W&B server
locally: https://wandb.me/wandb-server)
wandb: You can find your API key in your browser here:
https://wandb.ai/authorize
wandb: Paste an API key from your profile and hit enter:
```

```
 ..........
```

```
wandb: WARNING If you're specifying your api key in code, ensure this
code is not shared publicly.
wandb: WARNING Consider setting the WANDB_API_KEY environment
variable, or running `wandb login` from the command line.
wandb: No netrc file found, creating one.
wandb: Appending key for api.wandb.ai to your netrc file: /root/.netrc
wandb: Currently logged in as: trohith89 (trohith89-innomatics-
research-labs) to https://api.wandb.ai. Use `wandb login --relogin` to
force relogin
```

```
<IPython.core.display.HTML object>
```

```
<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>

<IPython.core.display.HTML object>
```

```
# Save the trained model, not the TrainOutput object
tr.model.save_pretrained("/content/hin_bert_model")
tokenizer.save_pretrained("/content/hin_bert_model")
```

```
('/content/hin_bert_model/tokenizer_config.json',
 '/content/hin_bert_model/special_tokens_map.json',
 '/content/hin_bert_model/vocab.txt',
 '/content/hin_bert_model/added_tokens.json',
 '/content/hin_bert_model/tokenizer.json')
```

```
from transformers import AutoModelForSequenceClassification,
AutoTokenizer

model =
AutoModelForSequenceClassification.from_pretrained("/content/hin_bert_
model")
tokenizer = AutoTokenizer.from_pretrained("/content/hin_bert_model")
```

```
!pip install -q huggingface_hub
from huggingface_hub import notebook_login

notebook_login()
```

```
{"model_id":"8864608c57ab4c859cd0f200c3aa1144","version_major":2,"vers
ion_minor":0}
```

```
model.push_to_hub("trohith89/Hindi_Sentiment_3_class")
tokenizer.push_to_hub("trohith89/Hindi_Sentiment_3_class")
```

```
{"model_id":"dd73b813c00e48f48fd2025c38c6dbbc","version_major":2,"vers
ion_minor":0}
```

```
{"model_id":"f9639b8c69a84f1aac7861ee0ce88743","version_major":2,"vers
ion_minor":0}
```

```
{"type":"string"}
```

```
# Use a pipeline as a high-level helper
from transformers import pipeline

pipe = pipeline("text-classification",
model="trohith89/Hindi_Sentiment_3_class")
```

{"model_id":"f00e3d4c81c549df8ba7bdd309d35e76","version_major":2,"version_minor":0}

{"model_id":"9a148f962e9b47d6bb1a3c84684fde4a","version_major":2,"version_minor":0}

{"model_id":"cc65c63612664e9c95d6636a10ab6349","version_major":2,"version_minor":0}

{"model_id":"3a4c602ba4554371b815206c6c0f0271","version_major":2,"version_minor":0}

{"model_id":"7aa17434d5c84cd7885f9641187c70fa","version_major":2,"version_minor":0}

{"model_id":"8857661df0f343dc8baac212bd88228f","version_major":2,"version_minor":0}

```
Device set to use cuda:0
```

```
names=["neutral", "positive", "negative"]

str = "यह फिल्म बहुत अच्छी थी और अभिनय शानदार था।"
# testing on hindi text for sentiment analysis
names[int(pipe(str)[0]['label'].split("_")[1])]
```

{"type":"string"}

```
# Define the label mapping (you may need to adjust based on the model you're using)
names = ["neutral", "positive", "negative"]

# Hindi test sentences
hindi_sentences = [
    "यह फिल्म बहुत अच्छी थी और अभिनय शानदार था।",
    "मुझे यह सेवा बहुत खराब लगी, बिल्कुल भी संतुष्ट नहीं हूँ।",
    "खाना ठीक-ठाक था, कुछ खास नहीं।",
    "इस मोबाइल की बैटरी लाइफ शानदार है।",
    "डिलीवरी समय पर नहीं हुई, बहुत निराशाजनक अनुभव रहा।",
    "यह जगह बहुत सुंदर है और यहां का वातावरण शांतिपूर्ण है।",
    "उत्पाद की गुणवत्ता उम्मीद से बहुत कम थी।",
    "मुझे यह किताब पढ़कर बहुत प्रेरणा मिली।",
    "कस्टमर सपोर्ट से बात करना आसान नहीं था।",
    "सब कुछ सामान्य था, न बहुत अच्छा न बहुत बुरा।"
]

# Loop through and print predicted sentiment
for sentence in hindi_sentences:
    label = pipe(sentence)[0]['label']  # e.g., 'LABEL_1'
```

```
    label_index = int(label.split("_")[-1])  # extract numeric part
    print(f"Text: {sentence}\nPredicted Sentiment:
{names[label_index]}\n")
```

```
You seem to be using the pipelines sequentially on GPU. In order to
maximize efficiency please use a dataset

Text: यह फिल्म बहुत अच्छी थी और अभिनय शानदार था।
Predicted Sentiment: positive

Text: मुझे यह सेवा बहुत खराब लगी, बिल्कुल भी संतुष्ट नहीं हूँ।
Predicted Sentiment: negative

Text: खाना ठीक-ठाक था, कुछ खास नहीं।
Predicted Sentiment: neutral

Text: इस मोबाइल की बैटरी लाइफ शानदार है।
Predicted Sentiment: positive

Text: डिलीवरी समय पर नहीं हुई, बहुत निराशाजनक अनुभव रहा।
Predicted Sentiment: negative

Text: यह जगह बहुत सुंदर है और यहां का वातावरण शांतिपूर्ण है।
Predicted Sentiment: positive

Text: उत्पाद की गुणवत्ता उम्मीद से बहुत कम थी।
Predicted Sentiment: negative

Text: मुझे यह किताब पढ़कर बहुत प्रेरणा मिली।
Predicted Sentiment: positive

Text: कस्टमर सपोर्ट से बात करना आसान नहीं था।
Predicted Sentiment: negative

Text: सब कुछ सामान्य था, न बहुत अच्छा न बहुत बुरा।
Predicted Sentiment: neutral
```

```
# Define the label mapping (you may need to adjust based on the model
you're using)
names = ["neutral", "positive", "negative"]

# Hindi test sentences
more_hindi_sentences = [
    "यह मोबाइल बहुत धीमा चलता है, उम्मीद के मुताबिक नहीं है।",
    "आज का मौसम बहुत सुहावना है, मन खुश हो गया।",
    "मुझे यह गाना पसंद नहीं आया, सुर और शब्द अच्छे नहीं थे।",
    "सेमिनार बहुत जानकारीपूर्ण था, मैंने बहुत कुछ सीखा।",
    "यह रेस्तरां बहुत महंगा है लेकिन खाना औसत था।",
```

```
    "मेरा अनुभव यहाँ बहुत ही निराशाजनक रहा।",
    "यह पुस्तक रोचक थी लेकिन अंत थोड़ा कमजोर था।",
    "क्लास का माहौल पढ़ाई के लिए बहुत अनुकूल है।",
    "कंप्यूटर बार-बार हैंग हो रहा है, बहुत परेशान हूँ।",
    "फिल्म का निर्देशन अच्छा था लेकिन कहानी साधारण थी।",
    "डॉक्टर ने सही समय पर इलाज शुरू किया, अब मैं बेहतर महसूस कर रहा हूँ।",
    "प्रोजेक्ट पूरा करने में काफी समय लगा लेकिन परिणाम अच्छा आया।",
    "मुझे इस यात्रा में बहुत आनंद आया, सब कुछ बेहतरीन था।",
    "यह घड़ी दिखने में तो सुंदर है पर समय सही नहीं बताती।",
    "आज ऑफिस में कुछ खास नहीं हुआ, बस रोज़ जैसा दिन था।",
    "ग्राहक सेवा से बात करना बहुत ही आसान और मददगार रहा।",
    "मुझे यह ऐप उपयोग करने में कठिनाई हो रही है।",
    "ऑनलाइन क्लास की गुणवत्ता में सुधार की जरूरत है।",
    "पार्क साफ-सुथरा और शांतिपूर्ण था, टहलना अच्छा लगा।",
    "यह उत्पाद विज्ञापन जितना अच्छा नहीं निकला।"
]


# Loop through and print predicted sentiment
for sentence in more_hindi_sentences:
    label = pipe(sentence)[0]['label']  # e.g., 'LABEL_1'
    label_index = int(label.split("_")[-1])  # extract numeric part
    print(f"Text: {sentence}\nPredicted Sentiment:
{names[label_index]}\n")
```

```
Text: यह मोबाइल बहुत धीमा चलता है, उम्मीद के मुताबिक नहीं है।
Predicted Sentiment: negative

Text: आज का मौसम बहुत सुहावना है, मन खुश हो गया।
Predicted Sentiment: negative

Text: मुझे यह गाना पसंद नहीं आया, सुर और शब्द अच्छे नहीं थे।
Predicted Sentiment: negative

Text: सेमिनार बहुत जानकारीपूर्ण था, मैंने बहुत कुछ सीखा।
Predicted Sentiment: positive

Text: यह रेस्तरां बहुत महंगा है लेकिन खाना औसत था।
Predicted Sentiment: negative

Text: मेरा अनुभव यहाँ बहुत ही निराशाजनक रहा।
Predicted Sentiment: negative

Text: यह पुस्तक रोचक थी लेकिन अंत थोड़ा कमजोर था।
Predicted Sentiment: positive

Text: क्लास का माहौल पढ़ाई के लिए बहुत अनुकूल है।
Predicted Sentiment: positive
```

```
Text: कंप्यूटर बार-बार हैंग हो रहा है, बहुत परेशान हूँ।
Predicted Sentiment: negative

Text: फिल्म का निर्देशन अच्छा था लेकिन कहानी साधारण थी।
Predicted Sentiment: positive

Text: डॉक्टर ने सही समय पर इलाज शुरू किया, अब मैं बेहतर महसूस कर रहा हूँ।
Predicted Sentiment: positive

Text: प्रोजेक्ट पूरा करने में काफी समय लगा लेकिन परिणाम अच्छा आया।
Predicted Sentiment: positive

Text: मुझे इस यात्रा में बहुत आनंद आया, सब कुछ बेहतरीन था।
Predicted Sentiment: positive

Text: यह घड़ी दिखने में तो सुंदर है पर समय सही नहीं बताती।
Predicted Sentiment: positive

Text: आज ऑफिस में कुछ खास नहीं हुआ, बस रोज़ जैसा दिन था।
Predicted Sentiment: neutral

Text: ग्राहक सेवा से बात करना बहुत ही आसान और मददगार रहा।
Predicted Sentiment: positive

Text: मुझे यह ऐप उपयोग करने में कठिनाई हो रही है।
Predicted Sentiment: negative

Text: ऑनलाइन क्लास की गुणवत्ता में सुधार की जरूरत है।
Predicted Sentiment: positive

Text: पार्क साफ-सुथरा और शांतिपूर्ण था, टहलना अच्छा लगा।
Predicted Sentiment: positive

Text: यह उत्पाद विज्ञापन जितना अच्छा नहीं निकला।
Predicted Sentiment: negative
```

4. Tokenization and Preprocessing

Add your tokenizer and data preprocessing steps here.

5. Model Fine-Tuning

Include your model loading, training arguments, and trainer here.

6. Evaluation and Insights

Display evaluation metrics, plots, and interpretation of results.