```
!pip -q install pandas numpy scikit-learn tensorflow transformers torch datasets evaluate accelerate streamlit cloudflared joblib
!pip install -U transformers accelerate datasets evaluate
```

```
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from transformers) (3.19.1)
Requirement already satisfied: huggingface-hub<1.0,>=0.34.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.34.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in /usr/local/lib/python3.12/dist-packages (from transformers) (25.0)
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in /usr/local/lib/python3.12/dist-packages (from transformers) (2024.11.6)
Requirement already satisfied: requests in /usr/local/lib/python3.12/dist-packages (from transformers) (2.32.4)
Requirement already satisfied: tokenizers<0.22,>=0.21 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.21.4)
Requirement already satisfied: safetensors>=0.4.3 in /usr/local/lib/python3.12/dist-packages (from transformers) (0.6.2)
Requirement already satisfied: tqdm>=4.27 in /usr/local/lib/python3.12/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: psutil in /usr/local/lib/python3.12/dist-packages (from accelerate) (5.9.5)
Requirement already satisfied: torch>=2.0.0 in /usr/local/lib/python3.12/dist-packages (from accelerate) (2.8.0+cu126)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.12/dist-packages (from datasets) (18.1.0)
Requirement already satisfied: dill<0.3.9,>=0.3.0 in /usr/local/lib/python3.12/dist-packages (from datasets) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: xxhash in /usr/local/lib/python3.12/dist-packages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess<0.70.17 in /usr/local/lib/python3.12/dist-packages (from datasets) (0.70.16)
Requirement already satisfied: fsspec<=2025.3.0,>=2023.1.0 in /usr/local/lib/python3.12/dist-packages (from fsspec[http]<=2025.3.0,>=202
Requirement already satisfied: aiohttp!=4.0.0a0,!=4.0.0a1 in /usr/local/lib/python3.12/dist-packages (from fsspec[http]<=2025.3.0,>=2023
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.0-
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub<1.0,>=0.34.0->trans
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.4.3)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2.5.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests->transformers) (2025.8.3)
Requirement already satisfied: setuptools in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (75.2.0)
Requirement already satisfied: sympy>=1.13.3 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (1.13.3)
Requirement already satisfied: networkx in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (3.5)
Requirement already satisfied: jinja2 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (3.1.6)
Requirement already satisfied: nvidia-cuda-nvrtc-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate
Requirement already satisfied: nvidia-cuda-runtime-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelera
Requirement already satisfied: nvidia-cuda-cupti-cu12==12.6.80 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate
Requirement already satisfied: nvidia-cudnn-cu12==9.10.2.21 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-cublas-cu12==12.6.4.1 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-cufft-cu12==11.3.0.4 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (1
Requirement already satisfied: nvidia-curand-cu12==10.3.7.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-cusolver-cu12==11.7.1.2 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-cusparse-cu12==12.5.4.2 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-cusparselt-cu12==0.7.1 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-nccl-cu12==2.27.3 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (2.27
Requirement already satisfied: nvidia-nvtx-cu12==12.6.77 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (12
Requirement already satisfied: nvidia-nvjitlink-cu12==12.6.85 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: nvidia-cufile-cu12==1.11.1.6 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate)
Requirement already satisfied: triton==3.4.0 in /usr/local/lib/python3.12/dist-packages (from torch>=2.0.0->accelerate) (3.4.0)
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas->datasets) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas->datasets) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas->datasets) (2025.2)
Requirement already satisfied: aiohappyeyeballs>=2.5.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fssp
Requirement already satisfied: aiosignal>=1.4.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fsspec[http
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fsspec[http]<=
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fsspec[htt
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fsspec[h
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fsspec[http
Requirement already satisfied: yarl<2.0,>=1.17.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fsspec[htt
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas->datasets) (1.17
Requirement already satisfied: mpmath<1.4,>=1.1.0 in /usr/local/lib/python3.12/dist-packages (from sympy>=1.13.3->torch>=2.0.0->accelera
Requirement already satisfied: MarkupSafe>=2.0 in /usr/local/lib/python3.12/dist-packages (from jinja2->torch>=2.0.0->accelerate) (3.0.2
```

1. Load Data

```
import pandas as pd

cols = ["id","entity","sentiment","text"]
train = pd.read_csv("/content/twitter_training.csv", names=cols, header=None)
valid = pd.read_csv("/content/twitter_validation.csv", names=cols, header=None)
df = pd.concat([train, valid], ignore_index=True)

# Basic clean
df = df.dropna(subset=["sentiment","text"])
df["sentiment"] = df["sentiment"].str.strip().str.title()  # e.g., positive -> Positive
df = df[df["sentiment"].isin(["Positive","Negative","Neutral"])]

# Remove dupes & tiny texts
df["text"] = df["text"].astype(str).str.replace(r"\s+", " ", regex=True).str.strip()
```

```
df = df[df["text"].str.len() > 3].drop_duplicates(subset=["text","sentiment"])

print(df["sentiment"].value_counts())
df.head()
```

```
sentiment
Negative    21163
Positive    19069
Neutral     17094
Name: count, dtype: int64
```

	id	entity	sentiment	text
0	2401	Borderlands	Positive	im getting on borderlands and i will murder yo...
1	2401	Borderlands	Positive	I am coming to the borders and I will kill you...
2	2401	Borderlands	Positive	im getting on borderlands and i will kill you ...
3	2401	Borderlands	Positive	im coming on borderlands and i will murder you...
4	2401	Borderlands	Positive	im getting on borderlands 2 and i will murder ...

Next steps: Generate code with df | View recommended plots | New interactive sheet

2. Train/Val/Test split + label mapping

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
import numpy as np
import json


le = LabelEncoder()
df["label"] = le.fit_transform(df["sentiment"])   # maps to 0/1/2
id2label = {i: c for i, c in enumerate(le.classes_)}
label2id = {c: i for i, c in id2label.items()}
print("Label mapping:", label2id)
```

```
Label mapping: {'Negative': 0, 'Neutral': 1, 'Positive': 2}
```

```
train_df, temp_df = train_test_split(df, test_size=0.2, random_state=42, stratify=df["label"])
val_df, test_df   = train_test_split(temp_df, test_size=0.5, random_state=42, stratify=temp_df["label"])


# Save label maps for later (Streamlit)
with open("label_map.json","w") as f:
    json.dump({"id2label": id2label, "label2id": label2id}, f)
```

3. TF-IDF + Logistic Regression

```
from sklearn.pipeline import Pipeline
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, accuracy_score
import joblib


tfidf_lr = Pipeline([
    ("tfidf", TfidfVectorizer(ngram_range=(1,2), min_df=2, max_features=80000)),
    ("clf", LogisticRegression(max_iter=200))
])


tfidf_lr.fit(train_df["text"], train_df["label"])
pred_val = tfidf_lr.predict(val_df["text"])
print("TFIDF+LR Val Acc:", accuracy_score(val_df["label"], pred_val))
print(classification_report(val_df["label"], pred_val, target_names=le.classes_))
```

```
TFIDF+LR Val Acc: 0.8871446014303157
              precision    recall  f1-score   support

    Negative       0.87      0.93      0.90      2117
     Neutral       0.91      0.84      0.87      1709
```

```
         Positive       0.88      0.89      0.89      1907

         accuracy                           0.89      5733
        macro avg       0.89      0.88      0.89      5733
     weighted avg       0.89      0.89      0.89      5733
```

```
# Save
joblib.dump(tfidf_lr, "tfidf_lr.joblib")
```

```
['tfidf_lr.joblib']
```

4. Deep (Classic NN): BiLSTM with Keras

```
import tensorflow as tf
from tensorflow.keras.preprocessing.text import Tokenizer
from tensorflow.keras.preprocessing.sequence import pad_sequences
from tensorflow.keras import layers, models


MAX_WORDS = 30000
MAX_LEN   = 64

tokenizer = Tokenizer(num_words=MAX_WORDS, oov_token="<UNK>")
tokenizer.fit_on_texts(train_df["text"])
X_train = pad_sequences(tokenizer.texts_to_sequences(train_df["text"]), maxlen=MAX_LEN)
X_val   = pad_sequences(tokenizer.texts_to_sequences(val_df["text"]),   maxlen=MAX_LEN)
y_train = train_df["label"].values
y_val   = val_df["label"].values


model = models.Sequential([
    layers.Embedding(input_dim=MAX_WORDS, output_dim=128, input_length=MAX_LEN),
    layers.Bidirectional(layers.LSTM(64, return_sequences=True)),
    layers.GlobalMaxPool1D(),
    layers.Dense(64, activation="relu"),
    layers.Dropout(0.2),
    layers.Dense(len(le.classes_), activation="softmax")
])
```

```
/usr/local/lib/python3.12/dist-packages/keras/src/layers/core/embedding.py:97: UserWarning: Argument `input_length` is deprecated. Just
  warnings.warn(
```

```
model.compile(optimizer="adam", loss="sparse_categorical_crossentropy", metrics=["accuracy"])
history = model.fit(X_train, y_train, epochs=3, batch_size=128, validation_data=(X_val, y_val), verbose=1)
```

```
Epoch 1/3
359/359 ━━━━━━━━━━━━━━━━━━━━ 13s 16ms/step - accuracy: 0.5771 - loss: 0.8750 - val_accuracy: 0.8278 - val_loss: 0.4531
Epoch 2/3
359/359 ━━━━━━━━━━━━━━━━━━━━ 5s 13ms/step - accuracy: 0.8991 - loss: 0.2885 - val_accuracy: 0.9037 - val_loss: 0.2732
Epoch 3/3
359/359 ━━━━━━━━━━━━━━━━━━━━ 5s 15ms/step - accuracy: 0.9502 - loss: 0.1510 - val_accuracy: 0.9144 - val_loss: 0.2503
```

```
# Quick evaluation
val_probs = model.predict(X_val, verbose=0)
val_pred  = val_probs.argmax(axis=1)
print("BiLSTM Val Acc:", (val_pred == y_val).mean())
```

```
BiLSTM Val Acc: 0.9143554857840572
```

```
# Save model + tokenizer
model.save("bilstm_keras.keras")

import pickle, json
with open("keras_tokenizer.pkl","wb") as f:
    pickle.dump(tokenizer, f)
```

5. Transformer: DistilBERT fine-tuning

```
from datasets import Dataset
from transformers import AutoTokenizer, AutoModelForSequenceClassification, DataCollatorWithPadding, TrainingArguments, Trainer
import evaluate, numpy as np
import torch


# Build HuggingFace Datasets
train_hf = Dataset.from_pandas(train_df[["text","label"]])
val_hf   = Dataset.from_pandas(val_df[["text","label"]])
test_hf  = Dataset.from_pandas(test_df[["text","label"]])

model_name = "distilbert-base-uncased"
tok = AutoTokenizer.from_pretrained(model_name)


def tokenize_fn(batch):
    return tok(batch["text"], truncation=True, max_length=128)

train_tok = train_hf.map(tokenize_fn, batched=True)
val_tok   = val_hf.map(tokenize_fn, batched=True)
test_tok  = test_hf.map(tokenize_fn, batched=True)

data_collator = DataCollatorWithPadding(tokenizer=tok)
num_labels = len(le.classes_)

model = AutoModelForSequenceClassification.from_pretrained(
    model_name, num_labels=num_labels, id2label=id2label, label2id=label2id
)

metric_acc = evaluate.load("accuracy")
metric_f1  = evaluate.load("f1")

def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=1)
    return {
        "accuracy": metric_acc.compute(predictions=preds, references=labels)["accuracy"],
        "f1_weighted": metric_f1.compute(predictions=preds, references=labels, average="weighted")["f1"]
    }
```

```
Map: 100%                                                       45860/45860 [00:04<00:00, 10075.29 examples/s]

Map: 100%                                                       5733/5733 [00:00<00:00, 9854.38 examples/s]

Map: 100%                                                       5733/5733 [00:00<00:00, 9932.19 examples/s]

Some weights of DistilBertForSequenceClassification were not initialized from the model checkpoint at distilbert-base-uncased and are n
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.

Downloading builder script:       4.20k/? [00:00<00:00, 337kB/s]

Downloading builder script:       6.79k/? [00:00<00:00, 341kB/s]
```

```
import transformers
print(transformers.__version__)
```

```
4.55.3
```

```
from transformers import TrainingArguments
args = TrainingArguments(
    output_dir="sa_distilbert_out",
    learning_rate=2e-5,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=32,
    num_train_epochs=2,
    weight_decay=0.01,
    eval_strategy="epoch",
    save_strategy="epoch",
    report_to="none",
    logging_steps=50,
    push_to_hub=False,
    load_best_model_at_end=True,
    metric_for_best_model="f1_weighted",
)
```

```
from huggingface_hub import login
login()
```

```
trainer = Trainer(
    model=model,
    args=args,
    train_dataset=train_tok,
    eval_dataset=val_tok,
    tokenizer=tok,
    data_collator=data_collator,
    compute_metrics=compute_metrics
)

trainer.train()
metrics = trainer.evaluate(val_tok)
metrics
```

```
/tmp/ipython-input-2048108774.py:1: FutureWarning: `tokenizer` is deprecated and will be removed in version 5.0.0 for `Trainer.__init__
  trainer = Trainer(
```

[5734/5734 11:13, Epoch 2/2]

Epoch	Training Loss	Validation Loss	Accuracy	F1 Weighted
1	0.475900	0.427929	0.841444	0.840773
2	0.266700	0.330047	0.884703	0.884673

[5734/5734 11:45, Epoch 2/2]

Epoch	Training Loss	Validation Loss	Accuracy	F1 Weighted
1	0.475900	0.427929	0.841444	0.840773
2	0.266700	0.330047	0.884703	0.884673

[180/180 00:12]

```
{'eval_loss': 0.330047070980072,
 'eval_accuracy': 0.8847025989883133,
 'eval_f1_weighted': 0.8846728418463662,
 'eval_runtime': 13.151,
 'eval_samples_per_second': 435.938,
 'eval_steps_per_second': 13.687,
 'epoch': 2.0}
```

6. Evaluate on test set & save the Transformer

```
test_metrics = trainer.evaluate(test_tok)
print(test_metrics)
```

[180/180 00:32]

```
{'eval_loss': 0.32654109597206116, 'eval_accuracy': 0.8881911739054597, 'eval_f1_weighted': 0.8880672076403741, 'eval_runtime': 15.3935
```

```
# Save model and tokenizer for Streamlit
save_dir = "sa_distilbert"
trainer.model.save_pretrained(save_dir)
tok.save_pretrained(save_dir)
```

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

```
# Also keep label maps with it
import json, os
with open(os.path.join(save_dir,"label_map.json"),"w") as f:
    json.dump({"id2label": id2label, "label2id": label2id}, f)
```

7. Streamlit app that lets you pick a model and predict

```
%%writefile app.py
import streamlit as st
import json, joblib, pickle, numpy as np, os
import tensorflow as tf
from tensorflow.keras.preprocessing.sequence import pad_sequences
from transformers import AutoTokenizer, AutoModelForSequenceClassification, TextClassificationPipeline

st.set_page_config(page_title="Sentiment Analysis", page_icon="🧠")
st.title("🧠 Sentiment Analysis (Positive / Negative / Neutral)")

st.markdown("Choose a model and enter text:")

MODEL_CHOICES = ["Transformer (DistilBERT)", "TF-IDF + Logistic Regression", "BiLSTM (Keras)"]
choice = st.selectbox("Model", MODEL_CHOICES)

text = st.text_area("Your text", height=120, placeholder="Type or paste a customer review here...")
btn = st.button("Analyze")

@st.cache_resource(show_spinner=False)
def load_transformer():
    dir_ = "sa_distilbert"
    tok = AutoTokenizer.from_pretrained(dir_)
    mdl = AutoModelForSequenceClassification.from_pretrained(dir_)
    with open(os.path.join(dir_,"label_map.json")) as f:
        lm = json.load(f)
    pipe = TextClassificationPipeline(model=mdl, tokenizer=tok, return_all_scores=True, truncation=True)
    return pipe, lm["id2label"]

@st.cache_resource(show_spinner=False)
def load_tfidf():
    pipe = joblib.load("tfidf_lr.joblib")
    with open("label_map.json") as f:
        lm = json.load(f)
    return pipe, lm["id2label"]

@st.cache_resource(show_spinner=False)
def load_bilstm():
    mdl = tf.keras.models.load_model("bilstm_keras")
    with open("keras_tokenizer.pkl","rb") as f:
        tok = pickle.load(f)
    with open("label_map.json") as f:
        lm = json.load(f)
    return mdl, tok, lm["id2label"]

def show_probs(probs_dict):
    st.subheader("Class probabilities")
    for item in probs_dict:
        st.write(f"**{item['label']}**: {item['score']:.3f}")

if btn and text.strip():
    if choice == "Transformer (DistilBERT)":
        pipe, id2label = load_transformer()
        out = pipe(text)[0]  # list of dicts
        # Sort for display
        out_sorted = sorted(out, key=lambda x: x["score"], reverse=True)
        st.success(f"**Prediction:** {out_sorted[0]['label']}")
        show_probs(out_sorted)

    elif choice == "TF-IDF + Logistic Regression":
        pipe, id2label = load_tfidf()
        proba = pipe.predict_proba([text])[0]
        pred_id = int(np.argmax(proba))
        pred_label = id2label[str(pred_id)]
        st.success(f"**Prediction:** {pred_label}")
        sorted_probs = [{"label": id2label[str(i)], "score": float(p)} for i,p in enumerate(proba)]
        sorted_probs.sort(key=lambda x: x["score"], reverse=True)
        show_probs(sorted_probs)

    else:  # BiLSTM
        mdl, tok, id2label = load_bilstm()
        MAX_LEN = 64
        seq = tok.texts_to_sequences([text])
        pad = pad_sequences(seq, maxlen=MAX_LEN)
        proba = mdl.predict(pad, verbose=0)[0]
        pred_id = int(np.argmax(proba))
        pred_label = id2label[str(pred_id)]
```

```
        st.success(f"**Prediction:** {pred_label}")
        sorted_probs = [{"label": id2label[str(i)], "score": float(p)} for i,p in enumerate(proba)]
        sorted_probs.sort(key=lambda x: x["score"], reverse=True)
        show_probs(sorted_probs)
```

```
Writing app.py
```

8. Run Streamlit

```
!pip install -q streamlit
!wget -q https://github.com/cloudflare/cloudflared/releases/latest/download/cloudflared-linux-amd64.deb
!dpkg -i cloudflared-linux-amd64.deb
```

```
(Reading database ... 126375 files and directories currently installed.)
Preparing to unpack cloudflared-linux-amd64.deb ...
Unpacking cloudflared (2025.8.1) over (2025.8.1) ...
Setting up cloudflared (2025.8.1) ...
Processing triggers for man-db (2.10.2-1) ...
```

```
!streamlit run app.py --server.headless true &>/content/logs.txt &
import time
time.sleep(5)  # wait for server to boot
```

```
!cloudflared tunnel --url http://localhost:8501 --no-autoupdate
```

```
2025-08-21T21:12:26Z INF Thank you for trying Cloudflare Tunnel. Doing so, without a Cloudflare account, is a quick way to experiment a
2025-08-21T21:12:26Z INF Requesting new quick Tunnel on trycloudflare.com...
2025-08-21T21:12:31Z INF +--------------------------------------------------------------------------------------------+
2025-08-21T21:12:31Z INF |  Your quick Tunnel has been created! Visit it at (it may take some time to be reachable):  |
2025-08-21T21:12:31Z INF |  https://britain-alpha-suites-belts.trycloudflare.com                                      |
2025-08-21T21:12:31Z INF +--------------------------------------------------------------------------------------------+
2025-08-21T21:12:31Z INF Cannot determine default configuration path. No file [config.yml config.yaml] in [~/.cloudflared ~/.cloudflare
2025-08-21T21:12:31Z INF Version 2025.8.1 (Checksum a66353004197ee4c1fcb68549203824882bba62378ad4d00d234bdb8251f1114)
2025-08-21T21:12:31Z INF GOOS: linux, GOVersion: go1.24.4, GoArch: amd64
2025-08-21T21:12:31Z INF Settings: map[ha-connections:1 no-autoupdate:true protocol:quic url:http://localhost:8501]
2025-08-21T21:12:31Z INF cloudflared will not automatically update if installed by a package manager.
2025-08-21T21:12:31Z INF Generated Connector ID: c5f99dd0-7aad-42b4-aa7e-08fee96f4af6
2025-08-21T21:12:31Z INF Initial protocol quic
2025-08-21T21:12:31Z INF ICMP proxy will use 172.28.0.12 as source for IPv4
2025-08-21T21:12:31Z INF ICMP proxy will use :: as source for IPv6
2025-08-21T21:12:31Z ERR Cannot determine default origin certificate path. No file cert.pem in [~/.cloudflared ~/.cloudflare-warp ~/clo
2025-08-21T21:12:31Z INF ICMP proxy will use 172.28.0.12 as source for IPv4
2025-08-21T21:12:31Z INF ICMP proxy will use :: as source for IPv6
2025-08-21T21:12:31Z INF Starting metrics server on 127.0.0.1:20241/metrics
2025-08-21T21:12:31Z INF Tunnel connection curve preferences: [X25519MLKEM768 CurveP256] connIndex=0 event=0 ip=198.41.192.27
2025/08/21 21:12:31 failed to sufficiently increase receive buffer size (was: 208 kiB, wanted: 7168 kiB, got: 416 kiB). See https://git
2025-08-21T21:12:32Z INF Registered tunnel connection connIndex=0 connection=5df68663-b2b6-4d0f-ae48-3a405cf0b9f9 event=0 ip=198.41.192
2025-08-21T21:14:28Z INF Initiating graceful shutdown due to signal interrupt ...
2025-08-21T21:14:29Z ERR failed to run the datagram handler error="context canceled" connIndex=0 event=0 ip=198.41.192.27
2025-08-21T21:14:29Z ERR failed to serve tunnel connection error="accept stream listener encountered a failure while serving" connIndex
2025-08-21T21:14:29Z ERR Serve tunnel error error="accept stream listener encountered a failure while serving" connIndex=0 event=0 ip=1
2025-08-21T21:14:29Z INF Retrying connection in up to 1s connIndex=0 event=0 ip=198.41.192.27
2025-08-21T21:14:29Z ERR Connection terminated connIndex=0
2025-08-21T21:14:29Z ERR no more connections active and exiting
2025-08-21T21:14:29Z INF Tunnel server stopped
2025-08-21T21:14:29Z INF Metrics server stopped
```

Final Push Git

```
from google.colab import auth
auth.authenticate_user()
```

```
!apt-get install git
```

```
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
```

```
git is already the newest version (1:2.34.1-1ubuntu1.15).
0 upgraded, 0 newly installed, 0 to remove and 35 not upgraded.
```

```
!git config --global user.email "duskee1234@gmail.com"
!git config --global user.name "malakasaber"
```

```
!git clone https://github.com/malakasaber/Sentiment-Analysis.git
```

```
Cloning into 'Sentiment-Analysis'...
warning: You appear to have cloned an empty repository.
```

```
!cp /content/NLP_Task_InstantTraining_Week3.ipynb /content/Sentiment-Analysis/
```

```
cp: cannot stat '/content/NLP_Task_InstantTraining_Week3.ipynb': No such file or directory
```

```
%cd /content/Sentiment-Analysis/
!git add .
!git commit -m "Added Colab notebook"
!git push origin main
```

```
/content/Sentiment-Analysis
On branch main

Initial commit

nothing to commit (create/copy files and use "git add" to track)
error: src refspec main does not match any
error: failed to push some refs to 'https://github.com/malakasaber/Sentiment-Analysis.git'
```

Start coding or generate with AI.