```
from google.colab import files

uploaded = files.upload()
```

```
Sélect. fichiers Aucun fichier choisi    Upload widget is only available when the cell has been executed in the current browser session. Please rerun
this cell to enable.
Saving test.txt to test.txt
Saving train.txt to train.txt
```

```
import pandas as pd

df_train = pd.read_csv("train.txt", sep=";", header=None)
df_validation = pd.read_csv("validation.txt", sep=";", header=None)
df_test = pd.read_csv("test.txt", sep=";", header=None)

df_train.columns = ['text', 'label']
df_validation.columns = ['text', 'label']
df_test.columns = ['text', 'label']

print(df_train.head())
print("Train shape:", df_train.shape)
```

```
                                              text    label
0                          i didnt feel humiliated  sadness
1  i can go from feeling so hopeless to so damned...  sadness
2   im grabbing a minute to post i feel greedy wrong    anger
3  i am ever feeling nostalgic about the fireplac...     love
4                             i am feeling grouchy    anger
Train shape: (16000, 2)
```

```
pip install evaluate
```

```
Collecting evaluate
  Downloading evaluate-0.4.6-py3-none-any.whl.metadata (9.5 kB)
Requirement already satisfied: datasets>=2.0.0 in /usr/local/lib/python3.12/dist-packages (from evaluate) (4.0.0)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.12/dist-packages (from evaluate) (2.0.2)
Requirement already satisfied: dill in /usr/local/lib/python3.12/dist-packages (from evaluate) (0.3.8)
Requirement already satisfied: pandas in /usr/local/lib/python3.12/dist-packages (from evaluate) (2.2.2)
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.12/dist-packages (from evaluate) (2.32.4)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.12/dist-packages (from evaluate) (4.67.1)
Requirement already satisfied: xxhash in /usr/local/lib/python3.12/dist-packages (from evaluate) (3.6.0)
Requirement already satisfied: multiprocess in /usr/local/lib/python3.12/dist-packages (from evaluate) (0.70.16)
Requirement already satisfied: fsspec>=2021.05.0 in /usr/local/lib/python3.12/dist-packages (from fsspec[http]>=2021.05.0->e
Requirement already satisfied: huggingface-hub>=0.7.0 in /usr/local/lib/python3.12/dist-packages (from evaluate) (0.36.0)
Requirement already satisfied: packaging in /usr/local/lib/python3.12/dist-packages (from evaluate) (25.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.12/dist-packages (from datasets>=2.0.0->evaluate) (3.20.0)
Requirement already satisfied: pyarrow>=15.0.0 in /usr/local/lib/python3.12/dist-packages (from datasets>=2.0.0->evaluate) (
Requirement already satisfied: pyyaml>=5.1 in /usr/local/lib/python3.12/dist-packages (from datasets>=2.0.0->evaluate) (6.0.
Requirement already satisfied: aiohttp!=4.0.0a0,!=4.0.0a1 in /usr/local/lib/python3.12/dist-packages (from fsspec[http]>=202
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub>=
Requirement already satisfied: hf-xet<2.0.0,>=1.1.3 in /usr/local/lib/python3.12/dist-packages (from huggingface-hub>=0.7.0-
Requirement already satisfied: charset_normalizer<4,>=2 in /usr/local/lib/python3.12/dist-packages (from requests>=2.19.0->e
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.12/dist-packages (from requests>=2.19.0->evaluate) (3.
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.12/dist-packages (from requests>=2.19.0->evaluat
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.12/dist-packages (from requests>=2.19.0->evaluat
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.12/dist-packages (from pandas->evaluate) (2.
Requirement already satisfied: pytz>=2020.1 in /usr/local/lib/python3.12/dist-packages (from pandas->evaluate) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.12/dist-packages (from pandas->evaluate) (2025.3)
Requirement already satisfied: aiohappyeyeballs>=2.5.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4
Requirement already satisfied: aiosignal>=1.4.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
Requirement already satisfied: attrs>=17.3.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1->fs
Requirement already satisfied: frozenlist>=1.1.1 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1
Requirement already satisfied: multidict<7.0,>=4.5 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0
Requirement already satisfied: propcache>=0.2.0 in /usr/local/lib/python3.12/dist-packages (from aiohttp!=4.0.0a0,!=4.0.0a1-
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
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.12/dist-packages (from python-dateutil>=2.8.2->pandas->eva
Downloading evaluate-0.4.6-py3-none-any.whl (84 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 84.1/84.1 kB 5.0 MB/s eta 0:00:00
Installing collected packages: evaluate
Successfully installed evaluate-0.4.6
```

```
!pip install transformers datasets accelerate -q
```

```
from datasets import Dataset, DatasetDict
from transformers import AutoTokenizer, AutoModelForSequenceClassification, TrainingArguments, Trainer, DataCollatorWithPac
import numpy as np
import evaluate

# Only train and test (as per project rules)
train_dataset = Dataset.from_pandas(df_train[['text', 'label']])
test_dataset = Dataset.from_pandas(df_test[['text', 'label']])
```

```
dataset = DatasetDict({'train': train_dataset, 'test': test_dataset})

# Label mapping
label_names = sorted(df_train['label'].unique())
label2id = {l: i for i, l in enumerate(label_names)}
id2label = {i: l for l, i in label2id.items()}

dataset = dataset.map(lambda x: {"label": label2id[x["label"]]})
```

```
Map: 100%                                      16000/16000 [00:00<00:00, 27833.00 examples/s]

Map: 100%                                      2000/2000 [00:00<00:00, 20974.04 examples/s]
```

```
# Tokenize
model_name = "distilbert-base-uncased"
tokenizer = AutoTokenizer.from_pretrained(model_name)

def tokenize(examples):
    return tokenizer(examples["text"], truncation=True, max_length=128)

tokenized_dataset = dataset.map(tokenize, batched=True)

# Model
model = AutoModelForSequenceClassification.from_pretrained(
    model_name, num_labels=6, id2label=id2label, label2id=label2id
)
```

```
/usr/local/lib/python3.12/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
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(
tokenizer_config.json: 100%                              48.0/48.0 [00:00<00:00, 2.16kB/s]

config.json: 100%                                        483/483 [00:00<00:00, 23.9kB/s]

vocab.txt: 100%                                          232k/232k [00:00<00:00, 6.73MB/s]

tokenizer.json: 100%                                     466k/466k [00:00<00:00, 13.4MB/s]

Map: 100%                                                16000/16000 [00:02<00:00, 6699.61 examples/s]

Map: 100%                                                2000/2000 [00:00<00:00, 6348.32 examples/s]

model.safetensors: 100%                                  268M/268M [00:04<00:00, 127MB/s]

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

```
# Metrics
accuracy = evaluate.load("accuracy")
def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=1)
    return accuracy.compute(predictions=preds, references=labels)
```

```
Downloading builder script:    4.20k/? [00:00<00:00, 161kB/s]
```

```
# Training
training_args = TrainingArguments(
    output_dir="./results",
    num_train_epochs=3,
    per_device_train_batch_size=32,
    per_device_eval_batch_size=32,
    eval_strategy="no",
    save_strategy="no",
    learning_rate=2e-5,
    load_best_model_at_end=False,
    report_to="none"
)

data_collator = DataCollatorWithPadding(tokenizer=tokenizer)

trainer = Trainer(
    model=model,
    args=training_args
    args=training_args,
    train_dataset=tokenized_dataset["train"],
    compute_metrics=compute_metrics,
    data_collator=data_collator
)

trainer.train()
```

```
[1500/1500 03:28, Epoch 3/3]
Step  Training Loss
 500       0.622500
1000       0.172400
1500       0.116600
TrainOutput(global_step=1500, training_loss=0.3037999827067057, metrics={'train_runtime': 210.0306,
'train_samples_per_second': 228.538, 'train_steps_per_second': 7.142, 'total_flos': 647803507542912.0, 'train_loss':
0.3037999827067057, 'epoch': 3.0})
```

```
# Final test evaluation
test_results = trainer.evaluate(tokenized_dataset["test"])
print(test_results)
```

```
[63/63 00:02]
{'eval_loss': 0.1772298514842987, 'eval_accuracy': 0.924, 'eval_runtime': 2.9683, 'eval_samples_per_second': 673.777, 'eval
```

```
Commencez à coder ou à générer avec l'IA.
```

```
trainer.save_model("./my_emotion_model")
```

```
from transformers import pipeline
classifier = pipeline("text-classification", model="./my_emotion_model", tokenizer=tokenizer)
print(classifier("i feel so happy today"))
print(classifier("this makes me really angry"))
```

```
Device set to use cuda:0
[{'label': 'joy', 'score': 0.9970676302909851}]
[{'label': 'anger', 'score': 0.9944250583648682}]
```

```
df_validation = pd.read_csv("validation.txt", sep=";", header=None)
df_validation.columns = ['text', 'label']
```

```
valid_dataset = Dataset.from_pandas(df_validation[['text', 'label']])
valid_dataset = valid_dataset.map(lambda x: {"label": label2id[x["label"]]})
valid_tokenized = valid_dataset.map(tokenize, batched=True)

valid_results = trainer.evaluate(valid_tokenized)
print(valid_results)
```

```
Map: 100%                                   2000/2000 [00:00<00:00, 9907.98 examples/s]
Map: 100%                                   2000/2000 [00:00<00:00, 4398.25 examples/s]
[63/63 05:34]
{'eval_loss': 0.14885811507701874, 'eval_accuracy': 0.9385, 'eval_runtime': 3.556, 'eval_samples_per_second': 562.434, 'eval
```