```
!pip install transformers
!pip install pandas
```

```
Requirement already satisfied: transformers in
/usr/local/lib/python3.11/dist-packages (4.52.4)
Requirement already satisfied: filelock in
/usr/local/lib/python3.11/dist-packages (from transformers) (3.18.0)
Requirement already satisfied: huggingface-hub<1.0,>=0.30.0 in
/usr/local/lib/python3.11/dist-packages (from transformers) (0.33.0)
Requirement already satisfied: numpy>=1.17 in
/usr/local/lib/python3.11/dist-packages (from transformers) (2.0.2)
Requirement already satisfied: packaging>=20.0 in
/usr/local/lib/python3.11/dist-packages (from transformers) (24.2)
Requirement already satisfied: pyyaml>=5.1 in
/usr/local/lib/python3.11/dist-packages (from transformers) (6.0.2)
Requirement already satisfied: regex!=2019.12.17 in
/usr/local/lib/python3.11/dist-packages (from transformers)
(2024.11.6)
Requirement already satisfied: requests in
/usr/local/lib/python3.11/dist-packages (from transformers) (2.32.3)
Requirement already satisfied: tokenizers<0.22,>=0.21 in
/usr/local/lib/python3.11/dist-packages (from transformers) (0.21.2)
Requirement already satisfied: safetensors>=0.4.3 in
/usr/local/lib/python3.11/dist-packages (from transformers) (0.5.3)
Requirement already satisfied: tqdm>=4.27 in
/usr/local/lib/python3.11/dist-packages (from transformers) (4.67.1)
Requirement already satisfied: fsspec>=2023.5.0 in
/usr/local/lib/python3.11/dist-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (2025.3.2)
Requirement already satisfied: typing-extensions>=3.7.4.3 in
/usr/local/lib/python3.11/dist-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (4.14.0)
Requirement already satisfied: hf-xet<2.0.0,>=1.1.2 in
/usr/local/lib/python3.11/dist-packages (from huggingface-
hub<1.0,>=0.30.0->transformers) (1.1.5)
Requirement already satisfied: charset-normalizer<4,>=2 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(3.4.2)
Requirement already satisfied: idna<4,>=2.5 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in
/usr/local/lib/python3.11/dist-packages (from requests->transformers)
(2025.6.15)
Requirement already satisfied: pandas in
/usr/local/lib/python3.11/dist-packages (2.2.2)
Requirement already satisfied: numpy>=1.23.2 in
```

```
/usr/local/lib/python3.11/dist-packages (from pandas) (2.0.2)
Requirement already satisfied: python-dateutil>=2.8.2 in
/usr/local/lib/python3.11/dist-packages (from pandas) (2.9.0.post0)
Requirement already satisfied: pytz>=2020.1 in
/usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: tzdata>=2022.7 in
/usr/local/lib/python3.11/dist-packages (from pandas) (2025.2)
Requirement already satisfied: six>=1.5 in
/usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2-
>pandas) (1.17.0)
```

```python
import pandas as pd

email_table = pd.read_csv('spam.csv', encoding='latin-1')
email_table.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 5572,\n  \"fields\":
[\n    {\n      \"column\": \"text\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 5169,\n
\"samples\": [\n          \"Did u download the fring app?\",\n
\"Pass dis to all ur contacts n see wat u get! Red;i'm in luv wid u.
Blue;u put a smile on my face. Purple;u r realy hot. Pink;u r so swt.
Orange;i thnk i lyk u. Green;i realy wana go out wid u. Yelow;i wnt u
bck. Black;i'm jealous of u. Brown;i miss you Nw plz giv me one
color\",\n          \"Ok...\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"target\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"spam\",\n          \"ham\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    }\n  ]\n}","type":"dataframe","variable_name":"df"}
```

```python
# Check actual column names
# Usually for spam.csv, columns are v1: spam_or_not (spam/ham), v2:
message
email_table = email_table.rename(columns={'target': 'spam_or_not'})
email_table = email_table[['email_body', 'spam_or_not']]
email_table.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 5572,\n  \"fields\":
[\n    {\n      \"column\": \"text\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 5169,\n
\"samples\": [\n          \"Did u download the fring app?\",\n
\"Pass dis to all ur contacts n see wat u get! Red;i'm in luv wid u.
Blue;u put a smile on my face. Purple;u r realy hot. Pink;u r so swt.
Orange;i thnk i lyk u. Green;i realy wana go out wid u. Yelow;i wnt u
bck. Black;i'm jealous of u. Brown;i miss you Nw plz giv me one
color\",\n          \"Ok...\"\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   \"spam\",\n   \"ham\"\n   ],\n
\"semantic_type\": \"\",\n   \"description\": \"\"\n   }\
n   }\n]\n}","type":"dataframe","variable_name":"df"}

```
email_table_sample = email_table.sample(100, random_state=42).copy()

from transformers import pipeline

llm_helper = pipeline("zero-shot-classification",
my_language_brain="facebook/bart-large-mnli")
```

```
/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":"f646b04d301b4579a2727b9723e13ce3","version_major":2,"vers
ion_minor":0}

{"model_id":"28ec33bfd10644bba57669005d683415","version_major":2,"vers
ion_minor":0}

{"model_id":"0c500cbf0a8546aba21c53b8da865c4f","version_major":2,"vers
ion_minor":0}

{"model_id":"3afd74dad9234c0b9fe0f8a34adb8299","version_major":2,"vers
ion_minor":0}

{"model_id":"c0884a455dd844a5bc40c4037abf5871","version_major":2,"vers
ion_minor":0}

{"model_id":"ab2b952ab22f429c89149cee63a328d2","version_major":2,"vers
ion_minor":0}

```
Device set to use cpu
```

```
candidate_spam_or_nots = ["spam", "ham"]

def classify_email(email_body):
    reply_from_llm = llm_helper(email_body, candidate_spam_or_nots)
    # Pick the spam_or_not with the highest score
    best_spam_or_not = reply_from_llm['spam_or_nots'][0]
    best_score = reply_from_llm['scores'][0]
    return pd.Series([best_spam_or_not, best_score])
```

```
email_table_sample[['llm_spam_or_not', 'llm_score']] =
email_table_sample['email_body'].apply(classify_email)
email_table_sample[['email_body', 'llm_spam_or_not', 'llm_score',
'spam_or_not']].head(10)
```

```
{"summary":"{\n  \"name\": \"df_sample[['text', 'llm_label',
'llm_score', 'label']]\",\n  \"rows\": 10,\n  \"fields\": [\n    {\n
\"column\": \"text\",\n      \"properties\": {\n        \"dtype\":
\"string\",\n        \"num_unique_values\": 10,\n        \"samples\":
[\n          \"Get down in gandhipuram and walk to cross cut road.
Right side &lt;#&gt; street road and turn at first right.\",\n
\"I sent my scores to sophas and i had to do secondary application for
a few schools. I think if you are thinking of applying, do a research
on cost also. Contact joke ogunrinde, her school is one me the less
expensive ones\",\n          \"I'll text carlos and let you know, hang
on\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"llm_label\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          \"spam\",\n          \"ham\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"llm_score\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0.10546401306590684,\n        \"min\": 0.5059730410575867,\n
\"max\": 0.788338840007782,\n        \"num_unique_values\": 10,\n
\"samples\": [\n          0.7627966403961182,\n
0.7790576815605164\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
\"spam\",\n          \"ham\"\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe"}
```

```
accuracy = (email_table_sample['llm_spam_or_not'] ==
email_table_sample['spam_or_not']).mean()
print(f"Zero-shot LLM accuracy: {accuracy:.2%}")
```

```
Zero-shot LLM accuracy: 74.00%
```