```
# Step 1: Install and Import Libraries
# Install the datasets library
!pip install datasets

import pandas as pd
from sklearn.model_selection import train_test_split
from transformers import AutoTokenizer, AutoModelForSequenceClassification, TrainingArguments, Trainer
import numpy as np
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score, roc_auc_score
from datasets import Dataset  # Import the Dataset class
import io

# Step 2: Load the Dataset
# Load the dataset, handling potential DtypeWarning
from google.colab import files
uploaded = files.upload()

for filename in uploaded.keys():
    if filename.endswith(".csv"):
        try:
            df = pd.read_csv(io.BytesIO(uploaded[filename]), low_memory=False)  # Add low_memory=False
        except UnicodeDecodeError:
            df = pd.read_csv(io.BytesIO(uploaded[filename]), encoding='latin1', low_memory=False)  # Add low_memory=False
        break

# Print the first few rows and column names to understand the data structure
print(df.head())
print(df.columns)
# Rename 'Email Type' to 'label'
df = df.rename(columns={'Email Type': 'label', 'Email Text': 'text'})

# Drop the 'sn' column, as it's likely an index column.
if 'sn' in df.columns:
    df = df.drop('sn', axis=1)

# Convert labels to integers (if needed)
if df['label'].dtype == 'object':
    df["label"] = df["label"].map({"Safe Email": 0, "Phishing Email": 1})
elif df['label'].dtype == 'float64':
    df['label'] = df['label'].astype(int)

# Inspect column names after renaming
print("Columns after renaming:", df.columns)

# Preprocessing
# Drop rows with missing values
df.dropna(inplace=True)

# Convert labels to integers
df['label'] = df['label'].astype(int)

# Rename Columns if needed.
if "text" not in df.columns or "label" not in df.columns:
    print("Error: DataFrame must have 'text' and 'label' columns.")
    exit()

# Split the Dataset
train_df, test_df = train_test_split(df, test_size=0.15, random_state=42)
train_df, val_df = train_test_split(train_df, test_size=0.15, random_state=42)

train_dataset = Dataset.from_pandas(train_df)
val_dataset = Dataset.from_pandas(val_df)
test_dataset = Dataset.from_pandas(test_df)

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

def tokenize_function(examples):
    texts = [str(text) for text in examples["text"]]
    return tokenizer(texts, padding="max_length", truncation=True, max_length=512)

train_dataset = train_dataset.map(tokenize_function, batched=True, batch_size=16)
val_dataset = val_dataset.map(tokenize_function, batched=True, batch_size=16)
test_dataset = test_dataset.map(tokenize_function, batched=True, batch_size=16)
```

```
# Load Pre-trained LLM
model = AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)

# Training Arguments
training_args = TrainingArguments(
    output_dir="./results",
    learning_rate=2e-5,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    num_train_epochs=3,
    weight_decay=0.01,
    logging_dir="./logs",
    logging_steps=10,
    save_strategy="epoch",
    save_total_limit=2,
    metric_for_best_model="f1",
    load_best_model_at_end=True,
    eval_strategy="epoch"
)

# Metrics
def compute_metrics(eval_pred):
    logits, labels = eval_pred
    predictions = np.argmax(logits, axis=-1)
    accuracy = accuracy_score(labels, predictions)
    precision = precision_score(labels, predictions, zero_division=0)
    recall = recall_score(labels, predictions, zero_division=0)
    f1 = f1_score(labels, predictions, zero_division=0)
    try:
        auc = roc_auc_score(labels, logits[:, 1])
    except ValueError:
        auc = 0.5
    return {
        "accuracy": accuracy,
        "precision": precision,
        "recall": recall,
        "f1": f1,
        "auc": auc,
    }

# Train
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=val_dataset,
    compute_metrics=compute_metrics,
)

trainer.train()

# Evaluate on Test Set
test_results = trainer.evaluate(test_dataset)
print("Test Results:", test_results)

# Save Model
model.save_pretrained("./phishing_detection_model")
tokenizer.save_pretrained("./phishing_detection_tokenizer")

# Load Model for Inference
loaded_model = AutoModelForSequenceClassification.from_pretrained("./phishing_detection_model")
loaded_tokenizer = AutoTokenizer.from_pretrained("./phishing_detection_tokenizer")

def predict_phishing(email_text):
    inputs = loaded_tokenizer(email_text, return_tensors="pt", padding=True, truncation=True, max_length=512)
    outputs = loaded_model(**inputs)
    probs = outputs.logits.softmax(dim=-1)
    return "Phishing" if probs.argmax().item() == 1 else "Legitimate"

sample_email = "Congratulations! You've won a $1000 gift card. Click here to claim your prize."
print("Prediction:", predict_phishing(sample_email))

# Extract and Print Tables of Results
print("\nTraining Results:")
training_metrics = trainer.state.log_history
for entry in training_metrics:
    if "eval_loss" in entry:
        print(entry)
```

```
            print(entry)

    print("\nTest Results Table:")
    test_table = pd.DataFrame([test_results])
    print(test_table)

    # Create a sample table from the training history.
    eval_results = []
    for entry in training_metrics:
      if 'eval_loss' in entry:
        eval_results.append(entry)

    if eval_results:
      eval_df = pd.DataFrame(eval_results)
      print("\nEvaluation Results During Training:")
      print(eval_df)

    else:
      print("\nNo Evaluation Results During Training to display as a table.")
```

```
Requirement already satisfied: datasets in /usr/local/lib/python3.11/dist-packages (2.14.4)
Requirement already satisfied: numpy>=1.17 in /usr/local/lib/python3.11/dist-packages (from datasets) (2.0.2)
Requirement already satisfied: pyarrow>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (18.1.0)
Requirement already satisfied: dill<0.3.8,>=0.3.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (0.3.7)
Requirement already satisfied: pandas in /usr/local/lib/python3.11/dist-packages (from datasets) (2.2.2)
Requirement already satisfied: requests>=2.19.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (2.32.3)
Requirement already satisfied: tqdm>=4.62.1 in /usr/local/lib/python3.11/dist-packages (from datasets) (4.67.1)
Requirement already satisfied: xxhash in /usr/local/lib/python3.11/dist-packages (from datasets) (3.5.0)
Requirement already satisfied: multiprocess in /usr/local/lib/python3.11/dist-packages (from datasets) (0.70.15)
Requirement already satisfied: fsspec>=2021.11.1 in /usr/local/lib/python3.11/dist-packages (from fsspec[http]>=2021.11.1->datasets) (20
Requirement already satisfied: aiohttp in /usr/local/lib/python3.11/dist-packages (from datasets) (3.11.15)
Requirement already satisfied: huggingface-hub<1.0.0,>=0.14.0 in /usr/local/lib/python3.11/dist-packages (from datasets) (0.31.2)
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) (1.6.0)
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) (1.20.0)
Requirement already satisfied: filelock in /usr/local/lib/python3.11/dist-packages (from huggingface-hub<1.0.0,>=0.14.0->datasets) (3.18
Requirement already satisfied: typing-extensions>=3.7.4.3 in /usr/local/lib/python3.11/dist-packages (from huggingface-hub<1.0.0,>=0.14.
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->datasets) (3.
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->datasets) (3.10)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->datasets) (2.4.0)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.11/dist-packages (from requests>=2.19.0->datasets) (2025.4.2
Requirement already satisfied: python-dateutil>=2.8.2 in /usr/local/lib/python3.11/dist-packages (from pandas->datasets) (2.9.0.post0)
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
```

Choose Files No file chosen Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.

```
Saving Phishing_Email 4.csv to Phishing_Email 4.csv
    sn                                         Email Text      Email Type
0  0.0  re : 6 . 1100 , disc : uniformitarianism , re ...      Safe Email
1  1.0  the other side of * galicismos * * galicismo *...      Safe Email
2  2.0  re : equistar deal tickets are you still avail...      Safe Email
3  3.0  \nHello I am your hot lil horny toy.\n    I am...  Phishing Email
4  4.0  software at incredibly low prices ( 86 % lower...  Phishing Email
Index(['sn', 'Email Text', 'Email Type'], dtype='object')
Columns after renaming: Index(['text', 'label'], dtype='object')
/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 secre
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, 2.56kB/s]

config.json: 100% 483/483 [00:00<00:00, 20.0kB/s]

vocab.txt: 100% 232k/232k [00:00<00:00, 5.45MB/s]

tokenizer.json: 100% 466k/466k [00:00<00:00, 22.1MB/s]

Map: 100% 2495/2495 [00:04<00:00, 569.26 examples/s]

Map: 100% 441/441 [00:00<00:00, 553.88 examples/s]

Map: 100% 519/519 [00:00<00:00, 593.26 examples/s]

```
Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to regular HTTP download. For better perfo
WARNING:huggingface_hub.file_download:Xet Storage is enabled for this repo, but the 'hf_xet' package is not installed. Falling back to r
```

model.safetensors: 100% 268M/268M [00:01<00:00, 201MB/s]

```
Some weights of DistilBertForSequenceClassification were not initialized from the model checkpoint at distilbert-base-uncased and are ne
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
wandb: WARNING The `run_name` is currently set to the same value as `TrainingArguments.output_dir`. If this was not intended, please spe
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?ref=models
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 nc
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: ucheignatiusopara (ucheignatiusopara-uwe) to https://api.wandb.ai. Use `wandb login --relogin` to force r
Tracking run with wandb version 0.19.11
Run data is saved locally in /content/wandb/run-20250518_150755-6nef8vj7
Syncing run ./results to Weights & Biases (docs)
View project at https://wandb.ai/ucheignatiusopara-uwe/huggingface
View run at https://wandb.ai/ucheignatiusopara-uwe/huggingface/runs/6nef8vj7
```

[468/468 6:10:08, Epoch 3/3]

Epoch	Training Loss	Validation Loss	Accuracy	Precision	Recall	F1	Auc
1	0.294200	0.163791	0.934240	0.944134	0.898936	0.920981	0.989151

2	0.021600	0.181323	0.938776	0.949721	0.904255	0.926431	0.988374

[23/28 04:42 < 01:04, 0.08 it/s]

[468/468 6:16:19, Epoch 3/3]

Epoch	Training Loss	Validation Loss	Accuracy	Precision	Recall	F1	Auc
1	0.294200	0.163791	0.934240	0.944134	0.898936	0.920981	0.989151
2	0.021600	0.181323	0.938776	0.949721	0.904255	0.926431	0.988374
3	0.007000	0.151350	0.947846	0.945946	0.930851	0.938338	0.993104

[33/33 06:48]

Test Results: {'eval_loss': 0.14528727531433105, 'eval_accuracy': 0.9537572254335526, 'eval_precision': 0.9348837209302325, 'eval_recall'