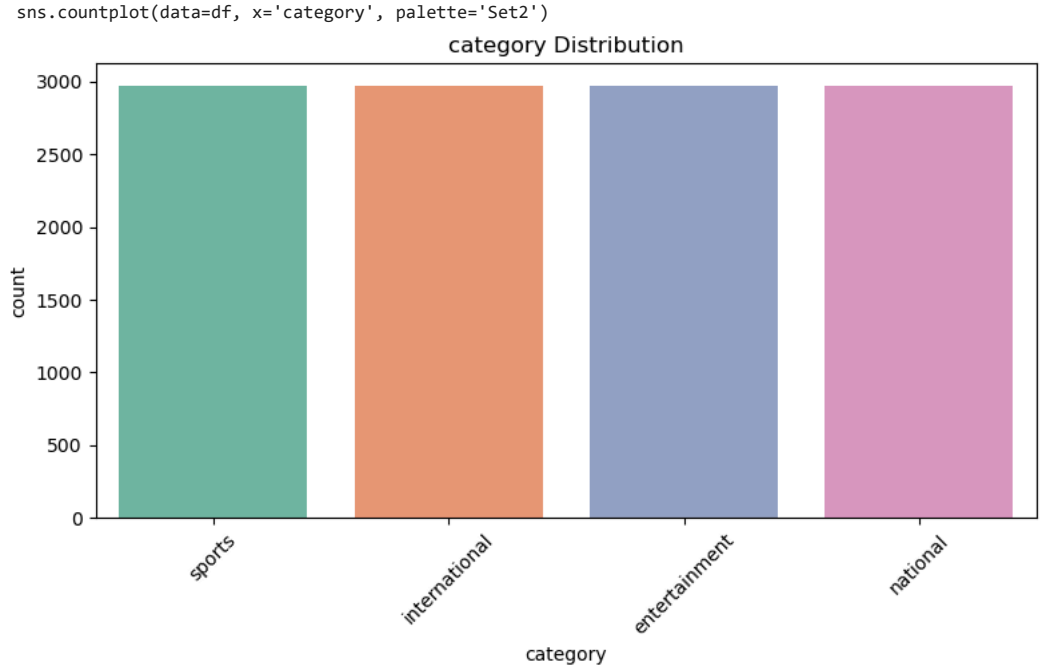
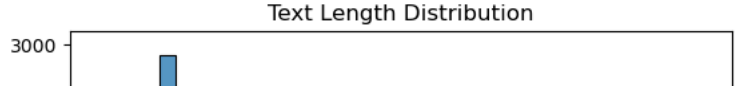
** Step 1: Exploratory Data Analysis (EDA)**

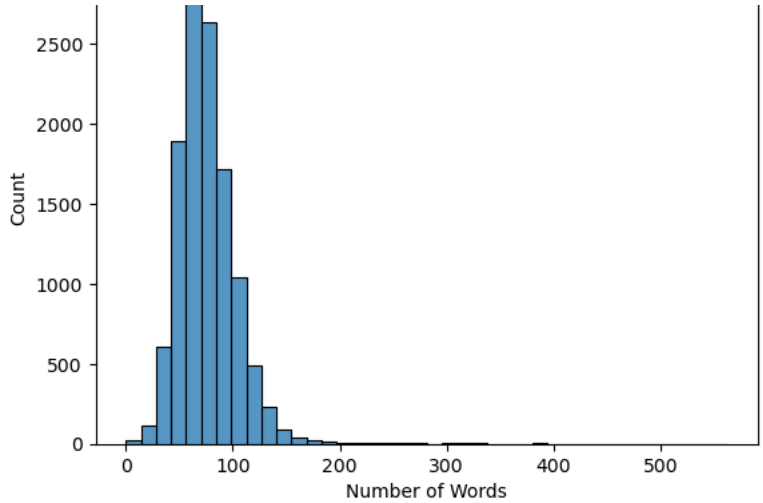
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Load the dataset
df = pd.read_csv("/Users/md.jubayerhossain/Downloads/Bangla_news.csv")

# Basic info
print(df.info())
print(df.head())

print(df['category'].value_counts())

# Check category distribution
plt.figure(figsize=(8,5))
sns.countplot(data=df, x='category', palette='Set2')
plt.title("category Distribution")
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()

# Check missing values
print(df.isnull().sum())

# Text length distribution
df['text_length'] = df['content'].astype(str).apply(lambda x: len(x.split()))
sns.histplot(df['text_length'], bins=40)
plt.title("Text Length Distribution")
plt.xlabel("Number of Words")
plt.show()

```

```
<class> pandas.core.frame.DataFrame
RangeIndex: 11904 entries, 0 to 11903
Data columns (total 6 columns):
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   title           11904 non-null  object
 1   published_date  11904 non-null  object
 2   reporter        10914 non-null  object
 3   category        11904 non-null  object
 4   url             11904 non-null  object
 5   content         11904 non-null  object
dtypes: object(6)
memory usage: 558.1+ KB
None
                                                title  \
0  সিন নদীতে ব্যাপক দূষণ, স্থগিত করা হলো ট্রায়াথ...
1            এসিসির নতুন সভাপতি হচ্ছেন মহসিন নাকভি!
2  ২০৩০ ও ২০৩৪ বিশ্বকাপের হোস্ট জানা যাবে দুইদিন পর!
3  প্যারিস অলিম্পিক: কোয়ার্টার ফাইনালে ওঠার লক্ষ্...
4                আজ টিভিতে যা দেখবেন (৩০ জুলাই ২০২৪)

             published_date reporter category                          url  \
0   30th July, 2024 5:59 pm   আরআইএম   sports  https://jamuna.tv/news/552127
1   30th July, 2024 5:25 pm      NaN   sports  https://jamuna.tv/news/552123
2   30th July, 2024 1:22 pm   আরআইএম   sports  https://jamuna.tv/news/552066
3  30th July, 2024 11:54 am     এনকে   sports  https://jamuna.tv/news/552046
4   30th July, 2024 7:01 am     এনকে   sports  https://jamuna.tv/news/552016

                                             content
0  ছবি: সংগৃহীত সিন নদীতে অলিম্পিকের উদ্বোধনী অনু...
1   ২০২৫ সালের জানুয়ারিতে এশিয়ান ক্রিকেট কাউন্স...
2   ফিফা বিশ্বকাপ ২০৩০ ও ৩৪ সালের স্বাগতিক হবার ল...
3   প্যারিস অলিম্পিকে কোয়ার্টার ফাইনাল রেসে টিকে ...
4  আজ ভারত-শ্রীলঙ্কার তৃতীয় টি-টোয়েন্টি অলিম্পিকে...
category
sports           2976
international    2976
entertainment    2976
national         2976
Name: count, dtype: int64
/var/folders/nm/l7zhs8816hv5d_c6lvpbzn8m0000gn/T/ipykernel_22789/1749407327.py:16: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `leg

  sns.countplot(data=df, x='category', palette='Set2')
```




```
title               0
published_date      0
reporter          990
category            0
url                 0
content             0
dtype: int64
```





Step 2: Data Preprocessing

```
import re
import nltk
from nltk.corpus import stopwords
nltk.download('stopwords')

# Remove non-Bangla characters & stopwords
stop_words = set(stopwords.words('bengali'))

def clean_text(text):
    text = str(text).lower()
    text = re.sub(r'http\S+', '', text)
    text = re.sub(r'[^অ-ঔক-হড়-য়\s]', '', text)
    text = " ".join([word for word in text.split() if word not in stop_words])
    return text.strip()

# Apply cleaning
df['clean_content'] = df['content'].apply(clean_text)
df.dropna(subset=['clean_content', 'category'], inplace=True)
df = df[df['clean_content'].str.strip() != '']

```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     /Users/md.jubayerhossain/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Step 3: Feature Engineering (Label Encoding & Train-Test Split)

```
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split

# Encode labels
le = LabelEncoder()
df['label'] = le.fit_transform(df['category'])

# Split
X_train, X_test, y_train, y_test = train_test_split(
    df['clean_content'], df['label'], test_size=0.2, stratify=df['label'], random_state=42
)

```

Step 4: Model Creation (Transformer - BanglaBERT)

```
import torch
from transformers import AutoTokenizer, AutoModelForSequenceClassification, TrainingArguments, Trainer
from datasets import Dataset
from tensorflow import keras

# Load tokenizer
tokenizer = AutoTokenizer.from_pretrained("sagorsarker/bangla-bert-base")

```

```
# Prepare datasets
train_dataset = Dataset.from_dict({'text': X_train.tolist(), 'label': y_train.tolist()})
test_dataset = Dataset.from_dict({'text': X_test.tolist(), 'label': y_test.tolist()})

# Tokenization
def tokenize(batch):
    return tokenizer(batch['text'], padding=True, truncation=True, max_length=256)

train_dataset = train_dataset.map(tokenize, batched=True)
test_dataset = test_dataset.map(tokenize, batched=True)

# Load model
model = AutoModelForSequenceClassification.from_pretrained("sagorsarker/bangla-bert-base", num_labels=len(le.classes_))

```

```
The OrderedVocab you are attempting to save contains holes for indices [1015, 1016, 1017, 1018, 1053, 1054, 1055, 1056, 1057, 1060, 1061
Map:   0%|          | 0/9521 [00:00<?, ? examples/s]
The OrderedVocab you are attempting to save contains holes for indices [1015, 1016, 1017, 1018, 1053, 1054, 1055, 1056, 1057, 1060, 1061
Map:   0%|          | 0/2381 [00:00<?, ? examples/s]
Some weights of BertForSequenceClassification were not initialized from the model checkpoint at sagorsarker/bangla-bert-base and are new
You should probably TRAIN this model on a down-stream task to be able to use it for predictions and inference.
```

Step 5: Model Evaluation

```
import evaluate
import numpy as np

# Load metrics once
accuracy_metric = evaluate.load("accuracy")
f1_metric = evaluate.load("f1")
precision_metric = evaluate.load("precision")
recall_metric = evaluate.load("recall")

def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=-1)
    return {
        "accuracy": accuracy_metric.compute(predictions=preds, references=labels)["accuracy"],
        "f1": f1_metric.compute(predictions=preds, references=labels, average="weighted")["f1"],
        "precision": precision_metric.compute(predictions=preds, references=labels, average="weighted")["precision"],
        "recall": recall_metric.compute(predictions=preds, references=labels, average="weighted")["recall"]
    }
```

Step 6: Model Tuning & **Training**

```
print(type(model))
print(len(train_dataset))
print(len(test_dataset))
```

```
<class 'transformers.models.bert.modeling_bert.BertForSequenceClassification'>
9521
2381
```

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

```
4.46.0
```

```
import numpy as np
import evaluate
from transformers import Trainer, TrainingArguments

# Load evaluation metrics
accuracy = evaluate.load("accuracy")
f1 = evaluate.load("f1")
precision = evaluate.load("precision")
recall = evaluate.load("recall")

# Define compute_metrics function
def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = np.argmax(logits, axis=-1)
    return {
```

```
        "accuracy": accuracy.compute(predictions=preds, references=labels)["accuracy"],
        "f1": f1.compute(predictions=preds, references=labels, average="weighted")["f1"],
        "precision": precision.compute(predictions=preds, references=labels, average="weighted")["precision"],
        "recall": recall.compute(predictions=preds, references=labels, average="weighted")["recall"]
    }

# Set training arguments
training_args = TrainingArguments(
    output_dir="./results",
    evaluation_strategy="epoch",        # ← this works in your version
    save_strategy="epoch",
    learning_rate=2e-5,
    per_device_train_batch_size=32,
    per_device_eval_batch_size=32,
    num_train_epochs=1,
    metric_for_best_model="f1",
    load_best_model_at_end=True,
    logging_dir='./logs',
    logging_steps=100
)

# Define Trainer
trainer = Trainer(
    model=model,                         # ← make sure your model is loaded
    args=training_args,
    train_dataset=train_dataset,         # ← make sure train_dataset is ready
    eval_dataset=test_dataset,           # ← make sure test_dataset is ready
    tokenizer=tokenizer,                 # ← ensure tokenizer is passed correctly
    compute_metrics=compute_metrics
)

# Train the model
trainer.train()

# Evaluate on test set
results = trainer.evaluate()
print("Evaluation results:", results)
```