テキスト分類の前処理

目的

- モデルに入力できる形にテキストを変換する
- テキストの「ノイズ」を減らして学習効率を上げる

なぜ前処理が必要か

- コンピュータは **文字列のままでは理解できない**
- モデルに渡すには **数値（ベクトル）** に変換する必要がある
- ノイズ（例: 記号やHTMLタグなど）が多いと学習が不安定

前処理の一般的な流れ

ステップ	内容
① テキストのクリーニング	不要な記号・改行・空白の除去
② 小文字化	Hello → hello（一貫性のため）
③ トークン化	テキストを単語・サブワード単位に分割
④ 数値化	単語をIDに変換（Tokenizerで実施）
⑤ パディング	長さの違う文章を同じ長さに揃える

前処理のイメージ

Original text: "This is awesome!"
↓
Cleaning: "this is awesome"
↓
Tokenize: ["this", "is", "awesome"]
↓
Numericalize: [2023, 2003, 6200]
↓
Padding: [2023, 2003, 6200, 0, 0, 0]

クリーニングの例

```
import re

text = “This is a sample review! 😀 <br> It‘s AWESOME!!!”

text = text.lower() # 小文字化

text = re.sub(r“<.*?>”, “”, text)    # HTMLタグ除去

text = re.sub(r“[^a-zA-Z0-9¥s]”, “”, text) # 記号・絵文字除去

print(text)
```

トークン化とテンソル化

```
from transformers import BertTokenizer

tokenizer = BertTokenizer.from_pretrained('bert-base-uncased’)

tokens = tokenizer("This is a sample review!", padding="max_length", max_length=10, truncation=True)

print(tokens['input_ids'])
```

実際のデータセットで行う

・データセットのロード

```
from datasets import load_dataset

# IMDBレビューのデータセットをロード
dataset = load_dataset("imdb")

print(dataset)
```

実際のデータセットで行う

・データクリーニング

```
import re


def clean_text(text):
    # HTMLタグ除去
    text = re.sub(r"<.*?>", "", text)
    # 記号除去
    text = re.sub(r"[^a-zA-Z0-9¥s]", "", text)
    # 小文字化
    text = text.lower()
    return text

# サンプル適用
sample_text = dataset["train"][0]["text"]
print("Before:", sample_text[:300])
print("After:", clean_text(sample_text)[:300])
```

実際のデータセットで行う

・トークナイズしてテンソル化

```
from transformers import BertTokenizer

# BERTトークナイザーをロード
tokenizer = BertTokenizer.from_pretrained('bert-base-uncased')

# サンプルデータに適用
encoded = tokenizer(
    clean_text(sample_text),
    padding="max_length",  # 長さを揃える
    truncation=True,       # 長すぎる文章をカット
    max_length=128,        # 最大長さを指定
    return_tensors="pt"    # PyTorchテンソルとして返す
)

print("Token IDs:", encoded["input_ids"])
```

実際のデータセットで行う

- データセットへの適応

```
def preprocess_function(examples):
    return tokenizer(
        [clean_text(text) for text in examples['text']],
        padding="max_length",
        truncation=True,
        max_length=128
    )

# データセットのトークナイズ
tokenized_datasets = dataset.map(preprocess_function, batched=True)
print(tokenized_datasets)
```