Import Necessary Dependencies

```
%pip install -q datasets
```

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from datasets import Dataset
```

Load the dataset

```
ds = pd.read_csv('/content/EcoPreprocessed.csv')
```

Perform EDA

Number of columns and tuples

```
tuples, columns = ds.shape
print(f'Number of Records: {tuples}\nNumber of features: {columns}')
```

```
Number of Records: 4084
Number of features: 4
```

Count of Null values

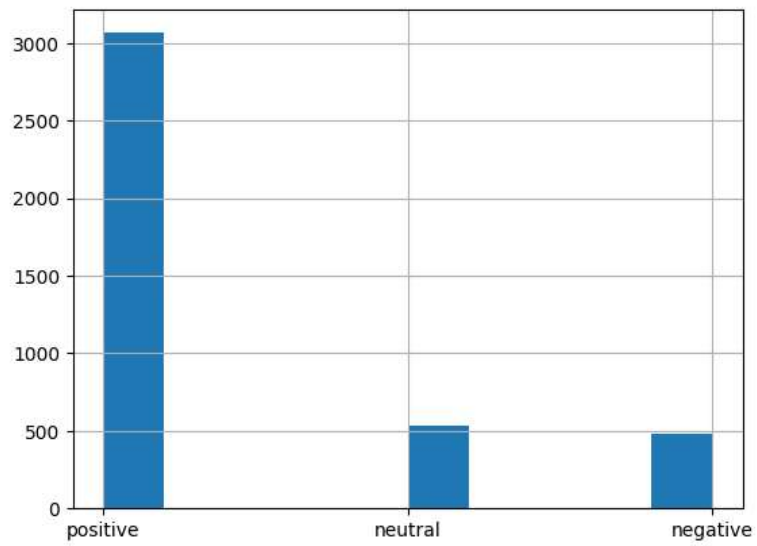
```
ds.isnull().sum() # no null value found
```

```
Unnamed: 0    0
review        0
polarity      0
division      0
dtype: int64
```

Distribution of the

```
import matplotlib.pyplot as plt
ds['division'].hist()
```

```
<Axes: >
```



Comment: As we can see above the database is highly biased

Utility objects and functions

```
labelencoder = LabelEncoder()
```

Preprocessing

Drop unnecessary columns

```
ds = ds.drop('Unnamed: 0', axis=1) # dropped Unnamed: 0
ds = ds.drop('polarity', axis=1) # dropped polarity
ds
```

	review	division
0	able play youtube alexa	positive
1	able recognize indian accent really well drop ...	positive
2	absolute smart device amazon connect external ...	positive
3	absolutely amaze new member family control hom...	positive
4	absolutely amaze previously sceptical invest m...	positive
...	...	...
4079	yo yo yo love go if want one smart speaker val...	positive
4080	youtube music	neutral
4081	youtube support nahi kartasong recognise achha...	neutral
4082	yup proscontrols wipro light amazinglysony bra...	neutral
4083	zero integration capabilities fire tv devices ...	negative

4084 rows × 2 columns

Encode labels accordingly

```
ds['division'] = labelencoder.fit_transform(ds['division'])
results = labelencoder.classes_
```

```
results
```

```
array(['negative', 'neutral', 'positive'], dtype=object)
```

```
ds.head()
```

	review	division
0	able play youtube alexa	2
1	able recognize indian accent really well drop ...	2
2	absolute smart device amazon connect external ...	2
3	absolutely amaze new member family control hom...	2
4	absolutely amaze previously sceptical invest m...	2

Create Huggingface Datasets

```
hds = Dataset.from_pandas(ds)
hds
```

```
Dataset({
    features: ['review', 'division'],
    num_rows: 4084
})
```

```
hds = hds.train_test_split(test_size=0.2, shuffle=True, seed=42)
```

```
hds
```

```
DatasetDict({
    train: Dataset({
        features: ['review', 'division'],
        num_rows: 3267
    })
    test: Dataset({
```

```
            features: ['review', 'division'],
            num_rows: 817
        })
    })
```

Fine tune the huggingface/setfit transformer based sentiment classifier

```
# %pip install -q setfit sentence_transformers # uncomment to download
```

```
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 45.9/45.9 kB 2.9 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 86.0/86.0 kB 5.6 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 81.4/81.4 kB 10.5 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.2/7.2 MB 78.6 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 53.1 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 7.8/7.8 MB 69.0 MB/s eta 0:00:00
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.3/1.3 MB 63.3 MB/s eta 0:00:00
  Building wheel for sentence_transformers (setup.py) ... done
```

```
from sentence_transformers.losses import CosineSimilarityLoss
from setfit import SetFitModel, SetFitTrainer, sample_dataset


train_dataset = sample_dataset(hds['train'], label_column="division", num_samples=8)
eval_dataset = hds['test']
```

Get the base model

```
model = SetFitModel.from_pretrained("sentence-transformers/paraphrase-mpnet-base-v2")
```

```
Downloading (…)lve/main/config.json: 100%  594/594 [00:00<00:00, 11.8kB/s]
Downloading (…)f39ef/.gitattributes: 100%  690/690 [00:00<00:00, 16.4kB/s]
Downloading (…)_Pooling/config.json: 100%  190/190 [00:00<00:00, 3.91kB/s]
Downloading (…)0182ff39ef/README.md: 100%  3.70k/3.70k [00:00<00:00, 139kB/s]
Downloading (…)82ff39ef/config.json: 100%  594/594 [00:00<00:00, 29.2kB/s]
Downloading (…)ce_transformers.json: 100%  122/122 [00:00<00:00, 3.34kB/s]
Downloading pytorch_model.bin: 100%  438M/438M [00:06<00:00, 73.8MB/s]
Downloading (…)nce_bert_config.json: 100%  53.0/53.0 [00:00<00:00, 860B/s]
Downloading (…)cial_tokens_map.json: 100%  239/239 [00:00<00:00, 5.79kB/s]
Downloading (…)f39ef/tokenizer.json: 100%  466k/466k [00:00<00:00, 4.60MB/s]
Downloading (…)okenizer_config.json: 100%  1.19k/1.19k [00:00<00:00, 49.9kB/s]
Downloading (…)0182ff39ef/vocab.txt: 100%  232k/232k [00:00<00:00, 3.40MB/s]
Downloading (…)2ff39ef/modules.json: 100%  229/229 [00:00<00:00, 4.03kB/s]
model_head.pkl not found on HuggingFace Hub, initialising classification head with random weights. You
```

```
trainer = SetFitTrainer(
    model=model,
    train_dataset=train_dataset,
    eval_dataset=eval_dataset,
    loss_class=CosineSimilarityLoss,
    metric="accuracy",
    batch_size=16,
    num_iterations=20, # The number of text pairs to generate for contrastive learning
    num_epochs=10, # The number of epochs to use for contrastive learning
    column_mapping={"review": "text", "division": "label"} # Map dataset columns to text/label expected by trainer
)


trainer.train()
```

```
Applying column mapping to training dataset
Generating Training Pairs: 100%  20/20 [00:00<00:00, 278.04it/s]
***** Running training *****
  Num examples = 960
  Num epochs = 10
  Total optimization steps = 600
  Total train batch size = 16
Epoch: 100%  10/10 [02:10<00:00, 13.06s/it]
Iteration: 100%  60/60 [00:13<00:00, 4.50it/s]
Iteration: 100%  60/60 [00:13<00:00, 4.66it/s]
Iteration: 100%  60/60 [00:13<00:00, 4.60it/s]
Iteration: 100%  60/60 [00:13<00:00, 4.60it/s]
Iteration: 100%  60/60 [00:12<00:00, 4.56it/s]
```

```
metrics = trainer.evaluate()
```

```
Applying column mapping to evaluation dataset
***** Running evaluation *****
Iteration: 100%  60/60 [00:12<00:00, 4.69it/s]
```

```
metrics
```

```
{'accuracy': 0.6719706242350061}
```

```
# from huggingface_hub import notebook_login

# notebook_login()
```

```
Token is valid (permission: write).
ır token has been saved in your configured git credential helpers (sto
Your token has been saved to /root/.cache/huggingface/token
Login successful
```

```
trainer.push_to_hub("AmitPress/bestsenti", )
```

```
Upload 2 LFS files: 100%  2/2 [00:09<00:00, 5.19s/it]
model_head.pkl: 100%  19.3k/19.3k [00:00<00:00, 35.2kB/s]
pytorch_model.bin: 100%  438M/438M [00:08<00:00, 36.1MB/s]
'https://huggingface.co/AmitPress/bestsenti/tree/main/'
```

✓ 0s completed at 1:01 PM