Time to slice and dice

Install the Transformers, Datasets, and Evaluate libraries to run this notebook.

```
[ ] !pip install datasets evaluate transformers[sentencepiece]
```

```
[ ] !wget "https://archive.ics.uci.edu/ml/machine-learning-databases/00462/drugsCom_raw.zip"
    !unzip drugsCom_raw.zip
```

```
[ ] from datasets import load_dataset

    data_files = {"train": "drugsComTrain_raw.tsv", "test": "drugsComTest_raw.tsv"}
    # \t is the tab character in Python
    drug_dataset = load_dataset("csv", data_files=data_files, delimiter="\t")
```

```
[ ] drug_sample = drug_dataset["train"].shuffle(seed=42).select(range(1000))
    # Peek at the first few examples
    drug_sample[:3]
```

```
{'Unnamed: 0': [87571, 178045, 80482],
 'drugName': ['Naproxen', 'Duloxetine', 'Mobic'],
 'condition': ['Gout, Acute', 'ibromyalgia', 'Inflammatory Conditions'],
 'review': ['"like the previous person mention, I&#039;m a strong believer of aleve, it works faster for my gout than the presc
refills.....Aleve works!"',
  '"I have taken Cymbalta for about a year and a half for fibromyalgia pain. It is great\r\nas a pain reducer and an anti-depre
benefit I got from it. I had trouble with restlessness, being tired constantly,\r\ndizziness, dry mouth, numbness and tingling
off of it now. Went from 60 mg to 30mg and now to 15 mg. I will be\r\noff completely in about a week. The fibro pain is coming
effects."',
  '"I have been taking Mobic for over a year with no side effects other than an elevated blood pressure.  I had severe knee and
Mobic.  I attempted to stop the medication however pain returned after a few days."'],
 'rating': [9.0, 3.0, 10.0],
 'date': ['September 2, 2015', 'November 7, 2011', 'June 5, 2013'],
```

```
 'usefulCount': [36, 13, 128]}
```

```
[ ] for split in drug_dataset.keys():
        assert len(drug_dataset[split]) == len(drug_dataset[split].unique("Unnamed: 0"))
```

```
[ ] drug_dataset = drug_dataset.rename_column(
        original_column_name="Unnamed: 0", new_column_name="patient_id"
    )
    drug_dataset
```

```
DatasetDict({
    train: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount'],
        num_rows: 161297
    })
    test: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount'],
        num_rows: 53766
    })
})
```

```
[ ] def lowercase_condition(example):
        return {"condition": example["condition"].lower()}


    drug_dataset.map(lowercase_condition)
```

```
AttributeError: 'NoneType' object has no attribute 'lower'
```

```
[ ] def filter_nones(x):
        return x["condition"] is not None
```

```
[ ] (lambda x: x * x)(3)
```

```
9
```

```
[ ] (lambda base, height: 0.5 * base * height)(4, 8)
```

```
16.0
```

```
[ ] drug_dataset = drug_dataset.filter(lambda x: x["condition"] is not None)
```

```
[ ] drug_dataset = drug_dataset.map(lowercase_condition)
    # Check that lowercasing worked
    drug_dataset["train"]["condition"][:3]
```

```
['left ventricular dysfunction', 'adhd', 'birth control']
```

```
[ ] def compute_review_length(example):
        return {"review_length": len(example["review"].split())}
```

```
[ ] drug_dataset = drug_dataset.map(compute_review_length)
    # Inspect the first training example
    drug_dataset["train"][0]
```

```
{'patient_id': 206461,
 'drugName': 'Valsartan',
 'condition': 'left ventricular dysfunction',
 'review': '"It has no side effect, I take it in combination of Bystolic 5 Mg and Fish Oil"',
 'rating': 9.0,
 'date': 'May 20, 2012',
 'usefulCount': 27,
 'review_length': 17}
```

```
[ ] drug_dataset["train"].sort("review_length")[:3]
```

```
{'patient_id': [103488, 23627, 20558],
 'drugName': ['Loestrin 21 1 / 20', 'Chlorzoxazone', 'Nucynta'],
 'condition': ['birth control', 'muscle spasm', 'pain'],
 'review': ['"Excellent."', '"useless"', '"ok"'],
 'rating': [10.0, 1.0, 6.0],
 'date': ['November 4, 2008', 'March 24, 2017', 'August 20, 2016'],
 'usefulCount': [5, 2, 10],
 'review_length': [1, 1, 1]}
```

```
[ ] drug_dataset = drug_dataset.filter(lambda x: x["review_length"] > 30)
    print(drug_dataset.num_rows)
```

```
{'train': 138514, 'test': 46108}
```

```
[ ] import html

    text = "I&#039;m a transformer called BERT"
    html.unescape(text)
```

```
"I'm a transformer called BERT"
```

```
[ ] drug_dataset = drug_dataset.map(lambda x: {"review": html.unescape(x["review"])})
```

```
[ ] new_drug_dataset = drug_dataset.map(
        lambda x: {"review": [html.unescape(o) for o in x["review"]]}, batched=True
    )
```

```
[ ] from transformers import AutoTokenizer

    tokenizer = AutoTokenizer.from_pretrained("bert-base-cased")


    def tokenize_function(examples):
        return tokenizer(examples["review"], truncation=True)
```

```
[ ] %time tokenized_dataset = drug_dataset.map(tokenize_function, batched=True)
```

```
[ ] slow_tokenizer = AutoTokenizer.from_pretrained("bert-base-cased", use_fast=False)


    def slow_tokenize_function(examples):
        return slow_tokenizer(examples["review"], truncation=True)


    tokenized_dataset = drug_dataset.map(slow_tokenize_function, batched=True, num_proc=8)
```

```
[ ] def tokenize_and_split(examples):
        return tokenizer(
            examples["review"],
            truncation=True,
            max_length=128,
            return_overflowing_tokens=True,
        )
```

```
[ ] result = tokenize_and_split(drug_dataset["train"][0])
    [len(inp) for inp in result["input_ids"]]
```

```
[128, 49]
```

```
[ ] tokenized_dataset = drug_dataset.map(tokenize_and_split, batched=True)
```

```
ArrowInvalid: Column 1 named condition expected length 1463 but got length 1000
```

```
[ ] tokenized_dataset = drug_dataset.map(
        tokenize_and_split, batched=True, remove_columns=drug_dataset["train"].column_names
    )
```

```
[ ] len(tokenized_dataset["train"]), len(drug_dataset["train"])
```

```
(206772, 138514)
```

```
[ ] def tokenize_and_split(examples):
        result = tokenizer(
            examples["review"],
            truncation=True,
            max_length=128,
            return_overflowing_tokens=True,
        )
        # Extract mapping between new and old indices
        sample_map = result.pop("overflow_to_sample_mapping")
        for key, values in examples.items():
            result[key] = [values[i] for i in sample_map]
        return result
```

```
[ ] tokenized_dataset = drug_dataset.map(tokenize_and_split, batched=True)
    tokenized_dataset
```

```
DatasetDict({
    train: Dataset({
        features: ['attention_mask', 'condition', 'date', 'drugName', 'input_ids', 'patient_id', 'rating', 'review', 'review_le
        num_rows: 206772
    })
    test: Dataset({
        features: ['attention_mask', 'condition', 'date', 'drugName', 'input_ids', 'patient_id', 'rating', 'review', 'review_le
```

```
        num_rows: 68876
    })
})
```

```
[ ] drug_dataset.set_format("pandas")
```

```
[ ] drug_dataset["train"][:3]
```

```
[ ] train_df = drug_dataset["train"][:]
```

```
[ ] frequencies = (
        train_df["condition"]
        .value_counts()
        .to_frame()
        .reset_index()
        .rename(columns={"index": "condition", "condition": "frequency"})
    )
```

```
[ ] from datasets import Dataset

    freq_dataset = Dataset.from_pandas(frequencies)
    freq_dataset
```

```
Dataset({
    features: ['condition', 'frequency'],
    num_rows: 819
})
```

```
[ ] drug_dataset.reset_format()
```

```
[ ] drug_dataset_clean = drug_dataset["train"].train_test_split(train_size=0.8, seed=42)
    # Rename the default "test" split to "validation"
    drug_dataset_clean["validation"] = drug_dataset_clean.pop("test")
    # Add the "test" set to our `DatasetDict`
    drug_dataset_clean["test"] = drug_dataset["test"]
    drug_dataset_clean
```

```
DatasetDict({
    train: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount', 'review_length', 'review_c
        num_rows: 110811
```

```
    })
    validation: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount', 'review_length', 'review_c
        num_rows: 27703
    })
    test: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount', 'review_length', 'review_c
        num_rows: 46108
    })
})
```

[ ]
```
drug_dataset_clean.save_to_disk("drug-reviews")
```

[ ]
```
from datasets import load_from_disk

drug_dataset_reloaded = load_from_disk("drug-reviews")
drug_dataset_reloaded
```

```
DatasetDict({
    train: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount', 'review_length'],
        num_rows: 110811
    })
    validation: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount', 'review_length'],
        num_rows: 27703
    })
    test: Dataset({
        features: ['patient_id', 'drugName', 'condition', 'review', 'rating', 'date', 'usefulCount', 'review_length'],
        num_rows: 46108
    })
})
```

[ ]
```
for split, dataset in drug_dataset_clean.items():
    dataset.to_json(f"drug-reviews-{split}.jsonl")
```

[ ]
```
!head -n 1 drug-reviews-train.jsonl
```

```
{"patient_id":141780,"drugName":"Escitalopram","condition":"depression","review":"\"I seemed to experience the regular side eff
during the day. I am taking it at night because my doctor said if it made me tired to take it at night. I assumed it would and
pleasant. I was diagnosed with fibromyalgia. Seems to be helping with the pain. Have had anxiety and depression in my family, a
worked. Only have been on it for two weeks but feel more positive in my mind, want to accomplish more in my life. Hopefully the
it from hearing others responses. Great medication.\"","rating":9.0,"date":"May 29, 2011","usefulCount":10,"review_length":125}
```

```
[ ]  data_files = {
         "train": "drug-reviews-train.jsonl",
         "validation": "drug-reviews-validation.jsonl",
         "test": "drug-reviews-test.jsonl",
     }
     drug_dataset_reloaded = load_dataset("json", data_files=data_files)
```