EXP 8

October 13, 2025

[ ]:
```
# Text Mining & LDA Topic Modeling on Multiple Articles
    # Extracts insights, common terms, and hidden topics from news articles from bs4
    import BeautifulSoup
    from sklearn.feature_extraction.text import CountVectorizer
    from sklearn.decomposition import LatentDirichletAllocation
    import re, html, numpy as np

    # 1) --- List of news article files (add or remove easily) --- files = ["The Indian
    Express.html", "The Economic Times.html", "TIE.html"]
```

[ ]:
```
# 2) --- Read and clean each article ---
    texts = []
    for FILE in files:
        with open(FILE, encoding="utf-8", errors="ignore") as f:
            data = f.read()
            s = BeautifulSoup(data, "html.parser")
            for t in s(["script", "style"]): t.decompose()
            text = " ".join(p.get_text(" ", strip=True) for p in s.find_all("p")) text =
            html.unescape(re.sub(r"https?://\S+|www\.\S+", " ", text)) text = re.sub(r"\s+", " ",
            text).strip()
            if len(text) > 50:
                texts.append(text)

    assert texts, "No valid text found in the given articles."
```

[ ]:
```
# 3) --- Create Bag-of-Words model ---
    cv = CountVectorizer(stop_words="english", min_df=2, max_df=0.95) X =
    cv.fit_transform(texts)
    vocab = np.array(cv.get_feature_names_out())

    # 4) --- Find common terms ---
    freq = np.asarray(X.sum(axis=0)).ravel()
    top = freq.argsort()[::-1][:15]
    print("\nTop Common Terms Across All Articles:")
    for w, c in zip(vocab[top], freq[top]):
        print(f"{w:20s} {int(c)}")
```

[ ]:
```
# 5) --- Apply LDA Topic Modeling ---
```

```
k = min(3, len(texts)) # number of topics (up to 3) lda =
LatentDirichletAllocation(n_components=k, random_state=0) lda.fit(X)

print("\nHidden Topics (LDA):")
for i, comp in enumerate(lda.components_):
        terms = vocab[comp.argsort()[::-1][:10]]
        print(f"Topic {i+1}: {' | '.join(terms)}")
```