Document Vectors

Minling Zhou mz246, Meixiang Du md480

Task:
Complete LSA and Word2vec to distinguish documents from two authors “Austen”, “Carroll” and discuss the results.

Build model:

- LSA – implementing Scikit-learn built-in module to train LSA model, set “n_components = 300” to meet the requirement.
  ```
  svd = TruncatedSVD(n_components=300).fit(X_train)
  ```
- Word2vec– implementing gensim built-in modules and utilizing the data set `"word2vec-google-news-300”` to train the model.

Result:
TF-IDF (train: 0.989, test: 0.969) demonstrated the best performance, followed by the raw counts model (train: 0.999, test: 0.972), LSA model (train: 0.968, test: 0.949), and Word2Vec (train: 0.939, test: 0.913) in the respective order.

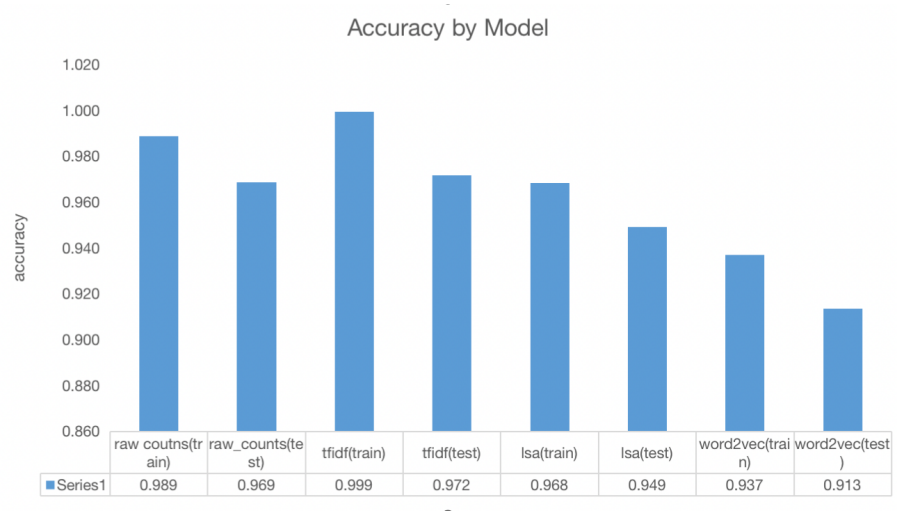
Insights:

1. **Vector size matters. Raw counts** and **TF-IDF** utilized the complete document size for training, whereas **LSA** and **Word2Vec** were trained on 300-size vectors. While gaining advantages in volume and computational efficiency, using truncated data may result in a negative impact on the overall result.

2. **Document similarity matters. Raw counts**, **TF-IDF**, and **LSA** utilize the "Austen" and "Carroll" documents for training, while **Word2Vec** uses "word2vec-google-news-300". The considerable difference in topic nature between Google News and the aforementioned Nobel authors may also result in a negative impact on the overall result.

Appendix:

1. accuracy compassion by bar graph.



2.Accuracy results from completed models.

```
Austen sentences: 4999
Carroll sentences: 1703
Vocabulary size: 8068
raw counts (train): 0.9887267904509284
raw_counts (test): 0.9686567164179104
tfidf (train): 0.9993368700265252
tfidf (test): 0.9716417910447761
lsa (train): 0.9671750663129973
lsa (test): 0.9507462686567164
word2vec (train): 0.9370026525198939
word2vec (test): 0.9134328358208955
```

3. Screenshot for LSA with vector size adjust to 6000, accuracy improved to close to raw data and TFIDF while it compromised in speed(vector size 300 run in 1 minute).

```
def generate_data_lsa(
    h0_documents: list[list[str]], h1_documents: list[list[str]]
) -> tuple[FloatArray, FloatArray, FloatArray, FloatArray]:
    """Generate training and testing data with LSA."""
    X_train, y_train, X_test, y_test = generate_data_token_count
        h0_documents, h1_documents
    )
    #lsa coding
    lsa = TruncatedSVD(n_components=6000, random_state=42).fit(X
    X_train = lsa.transform(X_train)
    X_test = lsa.transform(X_test)
    return X_train, y_train, X_test, y_test
```

```
run_experiment()
[1] ✓ 2m 6.2s
raw counts (train): 0.9887267904509284
raw_counts (test): 0.9686567164179104
tfidf (train): 0.9993368700265252
tfidf (test): 0.9716417910447761
lsa (train): 0.9892241379310345
lsa (test): 0.9626865671641791
Type 'python' code here and press Enter to run
```