Week 4 – Deployment on Flask – Book Genre predictor

Name: Jose Luis Castañeda Terrones

Batch code: LISUM 10

Submission date: 06/26/2022

Submitted to: https://github.com/JoseLuisCastanedaT/Book_genre_predictor

Dataset source: (https://www.cs.cmu.edu/~dbamman/booksummaries.html)

CMU Book Summary Dataset

The CMU Book Summary Dataset supports ongoing work described in:

David Bamman and Noah Smith (2013), "New Alignment Methods for Discriminative Book Summarization," [ArXiv]

booksummaries.tar.gz [17M]

This dataset contains plot summaries for 16,559 books extracted from Wikipedia, along with aligned metadata from Freebase, including book author, title, and genre.

All data is released under a Creative Commons Attribution-ShareAlike License. For questions or comments, please contact David Bamman (dbamman@cs.cmu.edu).

Example

The following example illustrates the data and metadata available for Don DeLillo's *White Noise*.

Book metadata

Wikipedia ID	1166383
Freebase ID	/m/04cvx9
Book title	White Noise
Book author	Don DeLillo
Publication date	1985-01-21
Genres	Novel, Postmodernism, Speculative fiction, Fiction

Plot summary

Set at a bucolic Midwestern college known only as The-College-on-the-Hill, White Noise follows a year in the life of Jack Gladney, a professor who has made his name by pioneering the field of Hitler Studies (though he hasn't taken German language lessons until this year). He has been married five times to four women and has a brood of children and stepchildren (Heinrich, Denise, Steffie, Wilder) with his current wife, Babette. Jack and Babette are both extremely afraid of death; they frequently wonder which of them will be the first to die. The first part of White Noise, called "Waves and Radiation," is a chronicle of contemporary family life combined with academic satire. There is little plot development in this section, which mainly serves as an introduction to the characters and themes that will dominate the rest of the book. For instance, the mysterious deaths of men in Mylex suits and the ashen, shaken survivors of a plane that went into free fall anticipate the catastrophe of the book's second part. Outside of the family, another important character introduced here is Murray, another college professor, who frequently discusses his theories, which relate to the rest of the book. In the second part, "The Airborne Toxic Event," a chemical spill from a rail car releases a black noxious cloud over Jack's home region, prompting an evacuation. Frightened by his exposure to the toxin, Gladney is forced to confront his mortality. An organization called SIMUVAC (short for "simulated evacuation") is also introduced in Part Two, an indication of simulations replacing reality. In part three of the book, "Dylarama," Gladney discovers that Babette has been cheating on him in order to gain access to a fictional drug called Dylar, an experimental treatment for the fear of death. The novel becomes a meditation on modern society's fear of death and its obsession with chemical cures as Gladney seeks to obtain his own black market supply of Dylar. However, Dylar does not work for Babette, and it has many possible side effects, including losing the ability to "distinguish words from things, so that if someone said 'speeding bullet,' I would fall to the floor to take cover." Jack continues to obsess over death. During a discussion about mortality, Murray hypothesizes that killing someone could perhaps alleviate the fear. Jack decides to test Murray's theory by tracking down and killing the man who had given Dylar to Babette in exchange for sex. After a black comedy scene of Jack driving and rehearsing, in his head, several ways in which their encounter might proceed, he successfully locates and shoots Willie Mink, who is delirious from his own Dylar addiction. He then puts the gun in Willie's hand to make the murder look like a suicide, but Willie then shoots Jack in the arm. Suddenly realizing the needless loss of life, Jack carries Willie to a hospital run by German nuns who do not believe in God or an afterlife. Having saved Willie, Jack returns home to watch his children sleep. The final chapter describes Wilder, Jack's youngest child, riding a tricycle across the highway and miraculously surviving.

Summary of the model: Given the summary of a book the model predicts its genre, it was trained on the CMU book dataset.

Steps of deployment:

Step 1: Importing necessary packages in app.py

app.py

app.py > cleantext

```
1   import pickle
2   from nltk.stem import PorterStemmer
3   from nltk.stem import WordNetLemmatizer
4   from nltk.corpus import stopwords
5   import re
6   import nltk
7   from flask import Flask, request, render_template
8
9   nltk.download('stopwords')
10  nltk.download('wordnet')
```

Step 2: Model pipeline in app.py

Cleaning the text

```
def cleantext(text):

    # removing the "\"
    text = re.sub("'\''", "", text)
    # removing special symbols
    text = re.sub("[^a-zA-Z]", " ", text)
    # removing the whitespaces
    text = ' '.join(text.split())
    # convert text to lowercase
    text = text.lower()

    return text
```

Removing stopwords

```
def removestopwords(text):
    stop_words = set(stopwords.words('english'))
    removedstopword = [word for word in text.split() if word not in stop_words]
    return ' '.join(removedstopword)
```

Lemmatizing

```
def lemmatizing(text):
    lemma = WordNetLemmatizer()
    stemSentence = ""
    for word in text.split():
        stem = lemma.lemmatize(word)
        stemSentence += stem
        stemSentence += " "
    stemSentence = stemSentence.strip()
    return stemSentence
```

Stemming

```
def stemming(text):

    stemmer = PorterStemmer()
    stemmed_sentence = ""
    for word in text.split():
        stem = stemmer.stem(word)
        stemmed_sentence += stem
        stemmed_sentence += " "

    stemmed_sentence = stemmed_sentence.strip()
    return stemmed_sentence
```

Testing

```
def test(text, model, tfidf_vectorizer):

    text = cleantext(text)
    text = removestopwords(text)
    text = lemmatizing(text)
    text = stemming(text)
    text_vector = tfidf_vectorizer.transform([text])
    predicted = model.predict(text_vector)

    newmapper = {0: 'Fantasy', 1: 'Science Fiction', 2: 'Crime Fiction',
                 3: 'Historical novel', 4: 'Horror', 5: 'Thriller'}

    return newmapper[predicted[0]]
```

Loading the pretrained model:

```
file = open('bookgenremodel.pkl', 'rb')
model = pickle.load(file)
file.close()

file1 = open('tfdifvector.pkl', 'rb')
tfidf_vectorizer = pickle.load(file1)
file1.close()
```

Step 2: Defining the Flask app in main.py

```python
app = Flask(__name__)


@app.route('/', methods=['GET', 'POST'])
def hello_world():

    if request.method == 'POST':

        mydict = request.form
        text = mydict["summary"]
        prediction = test(text, model, tfidf_vectorizer)

        return render_template('index.html', genre=prediction, text=str(text)[:100], showresult=True)
    return render_template('index.html')


if __name__ == '__main__':
    app.run(host='0.0.0.0',port=5080)
```

Step 3: Defining the index.html

```html
<!doctype html>
<html lang="en">

<head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css"
    integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">

    <title>Genre Prediction</title>
</head>

<body>

    <div class="container my-4">


        <h2 class="my-4 text-center">Book Genre Prediction</h2>

        <form class="my-4" action="/" method="POST">
            <div class="form-group">
                <label for="exampleFormControlTextarea1">Enter the Summary of the Book</label>
                <textarea class="form-control" id="summary" name="summary" rows="3"></textarea>
```

```
        <!-- button -->
        <div class="text-center">
            <button type="submit" class="btn btn-primary">Predict</button>
        </div>

    </form>

    {%if showresult%}

        <div class = "container my-4">

          <div class="alert alert-success" role="alert">

              <p>Your Summary : {{text}}...</p>

              <p>Predicted Genre for the summary is : <span style = "color: red;font-size:1 rem;">{{genre}}</p>
              <hr>

          </div>
        </div>
    {%endif%}
```

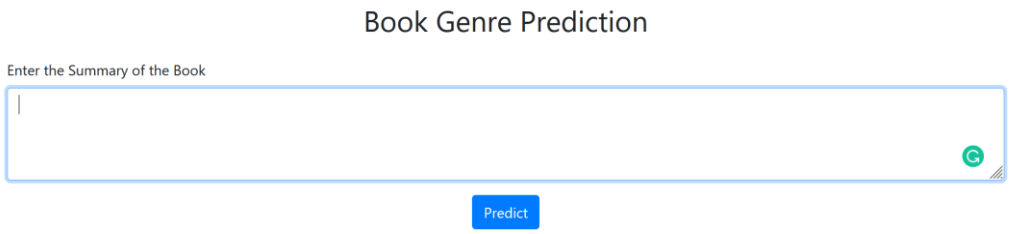
```
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbz
    <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqd
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB
</body>

</html>
```

Step 4: Running the model

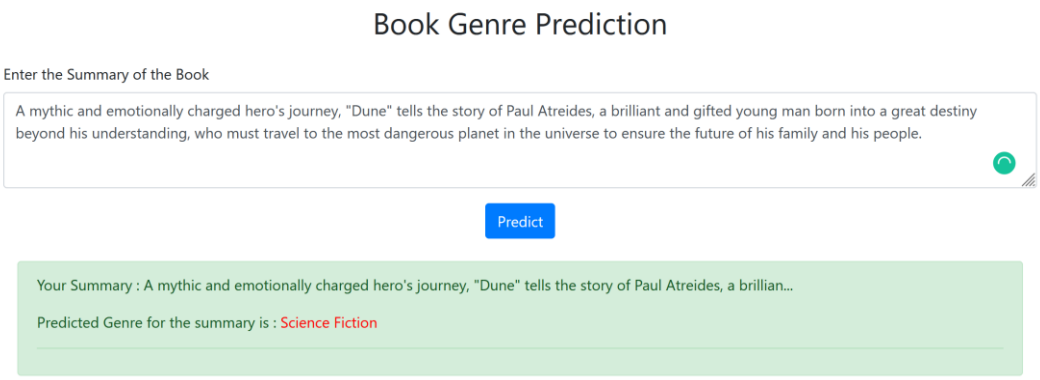
```
https://scikit-learn.org/stable/modules/model_persistence.html#security-maintainability-limitations
  warnings.warn(
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: off
 * Running on all addresses.
   WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://192.168.1.9:5080/ (Press CTRL+C to quit)
```

User Interface:

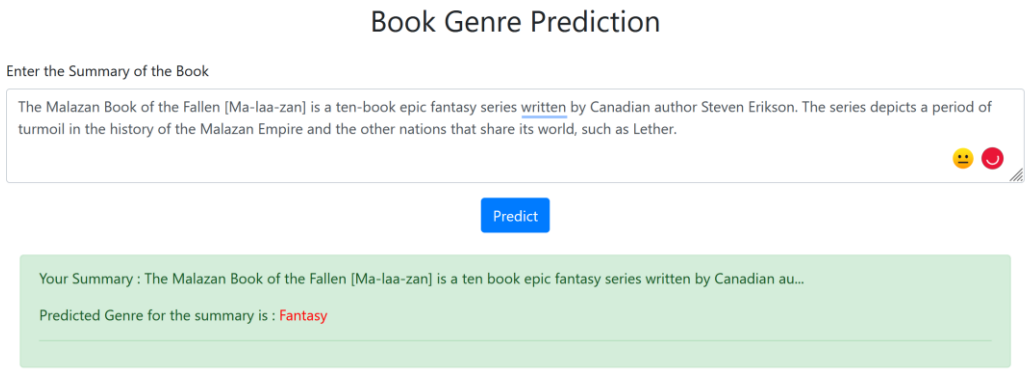


Step 5: Some examples with different books

Book 1: Dune (Genre: Science Fiction)



Book 2: Malazan Book of the Fallen (Genre: Fantasy)



Book 3: Wolf Hall (Genre: Historical novel)