Đoạn code trên đang thực hiện các bước sau:

1 Import thư viện: Import các thư viện cần thiết như json, re, nltk, CountVectorizer và MultinomialNB từ sklearn.

2 Tải dữ liệu và stopwords: Đọc dữ liệu từ tệp JSON và danh sách stopwords từ tệp JSON khác.

```
with open('custom_stopwords.json') as file:
    stopwords_list = json.load(file)
```

3 Tiền xử lý dữ liệu: Định nghĩa hàm preprocess_text để tiền xử lý văn bản bằng cách loại bỏ ký tự không cần thiết, chuyển văn bản về chữ thường, loại bỏ stopwords và các ký tự đơn lẻ.

```
def preprocess_text(text):
    text = re.sub(r'\W', ' ', text)
    text = text.lower()
    text = re.sub(r'\s+[a-zA-Z]\s+', ' ', text)  # Loại bỏ ký tự đơn lẻ
    text = re.sub(r'\^[a-zA-Z]\s+', ' ', text)  # Loại bỏ ký tự đầu dòng
    text = re.sub(r'\s+', ' ', text, flags=re.I)  # Loại bỏ khoảng trắng thừa
    text = ' '.join(word for word in text.split() if word not in stop_words)  # Loại bỏ sto
    return text
```

4 Chuẩn bị dữ liệu huấn luyện: Tạo danh sách bình luận và nhãn từ dữ liệu đã đọc, trong đó nhãn là mức độ độc hại của bình luận.

5 Tiền xử lý và mã hóa dữ liệu: Sử dụng CountVectorizer để mã hóa bình luận thành các vectơ đặc trưng dựa trên tần suất xuất hiện của từng từ trong từ điển.

```
# Tiền xử lý và mã hóa dữ liệu
vectorizer = CountVectorizer(stop_words=stop_words)
X = vectorizer.fit_transform(comments)
```

6 Xây dựng và huấn luyện mô hình Naive Bayes: Tạo một mô hình Naive Bayes đa thức và huấn luyện mô hình trên dữ liệu đã chuẩn bị.

```
# Xây dựng mô hình Naive Bayes
model = MultinomialNB()

# Huấn luyện mô hình
model.fit(X, labels)
```

7 Dự đoán độ độc hại của bình luận: Dự đoán mức độ độc hại của từng bình luận trong tệp dữ liệu sử dụng mô hình đã huấn luyện.

8 Ghi dữ liệu đã được gán nhãn vào tệp JSON: Ghi dữ liệu đã được gán nhãn (bao gồm mức độ độc hại dự đoán) vào một tệp JSON mới có tên là DATA_GOOD_FINAL.json.

Nói tóm lại, đoạn mã này đang thực hiện quá trình tiền xử lý, huấn luyện một mô hình Naive Bayes và sử dụng mô hình này để dự đoán mức độ độc hại của các bình luận trong dữ liệu đầu vào, sau đó ghi kết quả dự đoán vào một tệp JSON mới.

Trong đoạn mã trên, mô hình Naive Bayes được áp dụng như sau:

1. Xây dựng mô hình Naive Bayes:

```
# Xây dựng mô hình Naive Bayes
model = MultinomialNB()
```

Ở đây, MultinomialNB() là một lớp trong thư viện sklearn để xây dựng mô hình Naive Bayes đa thức.

2. Huấn luyện mô hình:

```
# Huấn luyện mô hình
model.fit(X, labels)
```

Hàm fit được sử dụng để huấn luyện mô hình trên dữ liệu huấn luyện X và nhãn tương ứng labels. Trong trường hợp này, X là các vectơ đặc trưng của các bình luận được mã hóa bằng CountVectorizer, và labels là mức độ độc hại của từng bình luận.

3. Dự đoán độ độc hại của bình luận:

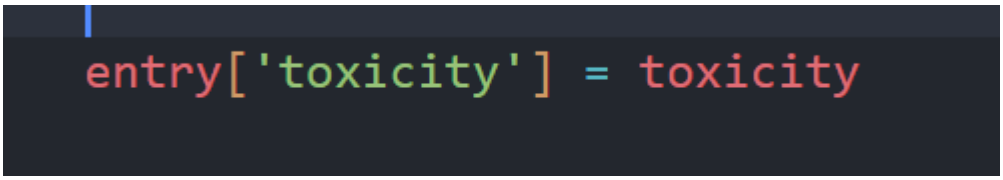
```
for entry in data:
    comment = entry['comment']
    preprocessed_comment = preprocess_text(comment)
    comment_vector = vectorizer.transform([preprocessed_comment])

    toxicity = model.predict_proba(comment_vector)[0][1]

    entry['toxicity'] = toxicity
```

predict_proba được sử dụng để dự đoán xác suất của từng lớp cho các mẫu mới. Trong trường hợp này, comment_vector là vectơ đặc trưng của một bình luận đã được tiền xử lý và mã hóa, và model.predict_proba(comment_vector) trả về một mảng 2 chiều với các giá trị xác suất cho mỗi lớp (trong trường hợp này, lớp 0 và lớp 1, tương ứng với không độc hại và độc hại). Đối với mỗi bình luận, ta quan tâm đến xác suất của lớp độc hại (lớp 1), do đó, [0][1] được sử dụng để lấy xác suất của lớp 1.

4. Gán kết quả dự đoán vào dữ liệu:



Kết quả dự đoán được gán vào thuộc tính 'toxicity' của từng mục trong dữ liệu đầu vào. Điều này cho phép lưu lại thông tin về mức độ độc hại được dự đoán của từng bình luận.