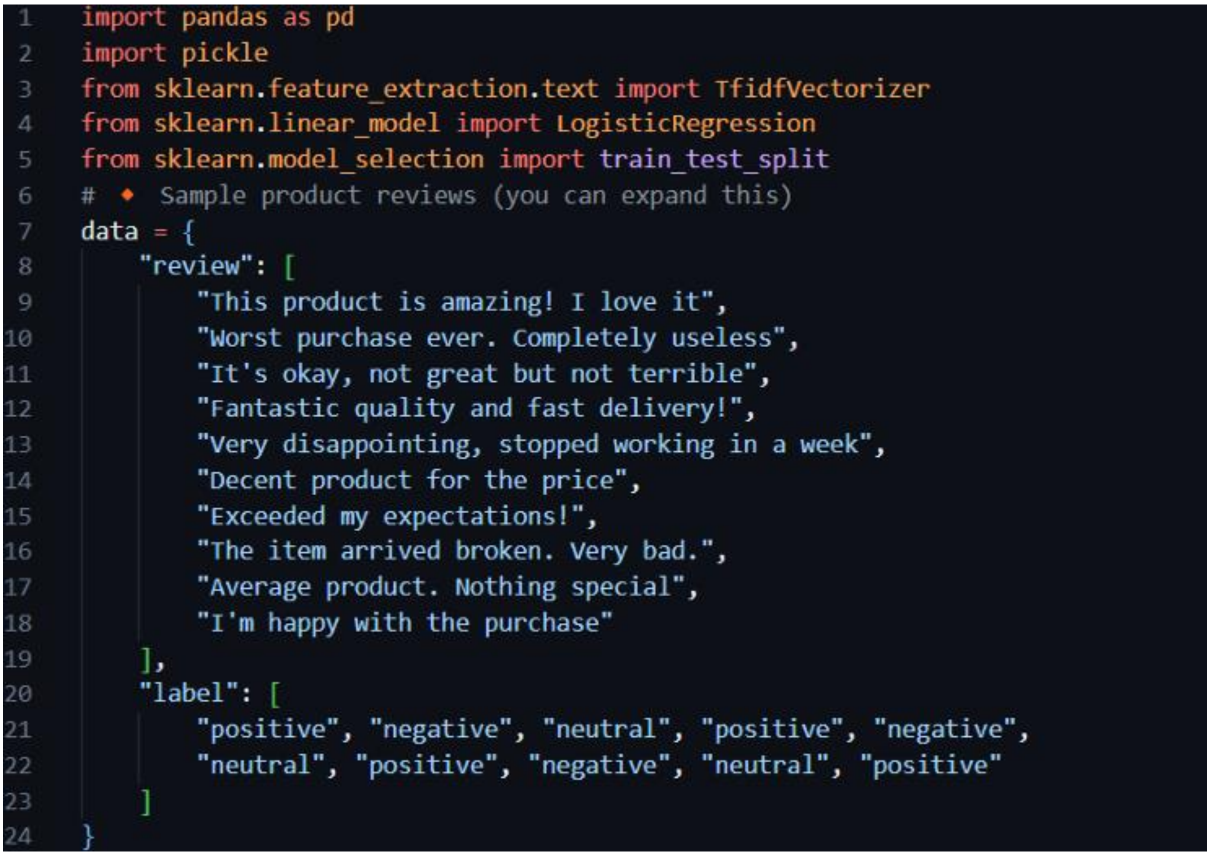
CHANDIGARH UNIVERSITY

Discover. Learn. Empower.

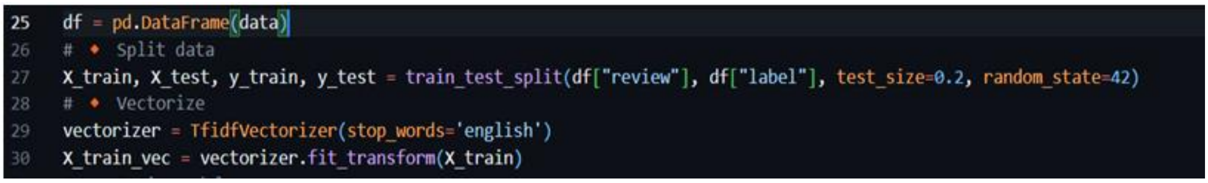
Name: Johnson Kumar **Uid:** 232bcs12654

Topic: Product Review Classifier (E-Commerce Focus): Scrape product reviews from Amazon or Flipkart, use NLP to clean and analyze the reviews, and classify the sentiment using a machine learning model. The results are displayed via a Flask web app.

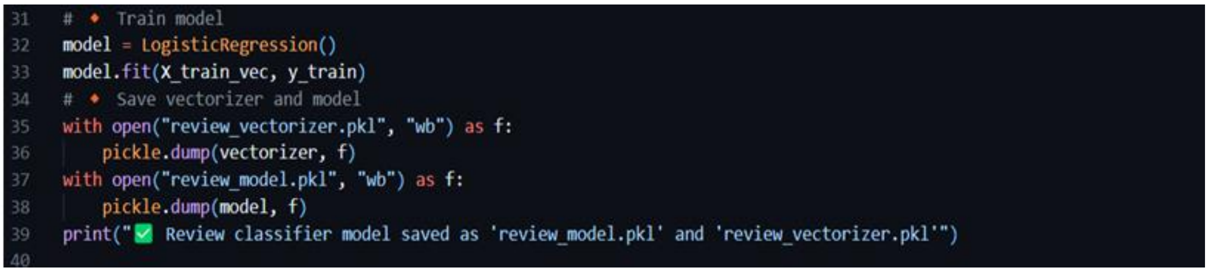
Model training code:



```python
import pandas as pd
import pickle
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
# 🔸 Sample product reviews (you can expand this)
data = {
    "review": [
        "This product is amazing! I love it",
        "Worst purchase ever. Completely useless",
        "It's okay, not great but not terrible",
        "Fantastic quality and fast delivery!",
        "Very disappointing, stopped working in a week",
        "Decent product for the price",
        "Exceeded my expectations!",
        "The item arrived broken. Very bad.",
        "Average product. Nothing special",
        "I'm happy with the purchase"
    ],
    "label": [
        "positive", "negative", "neutral", "positive", "negative",
        "neutral", "positive", "negative", "neutral", "positive"
    ]
}
```

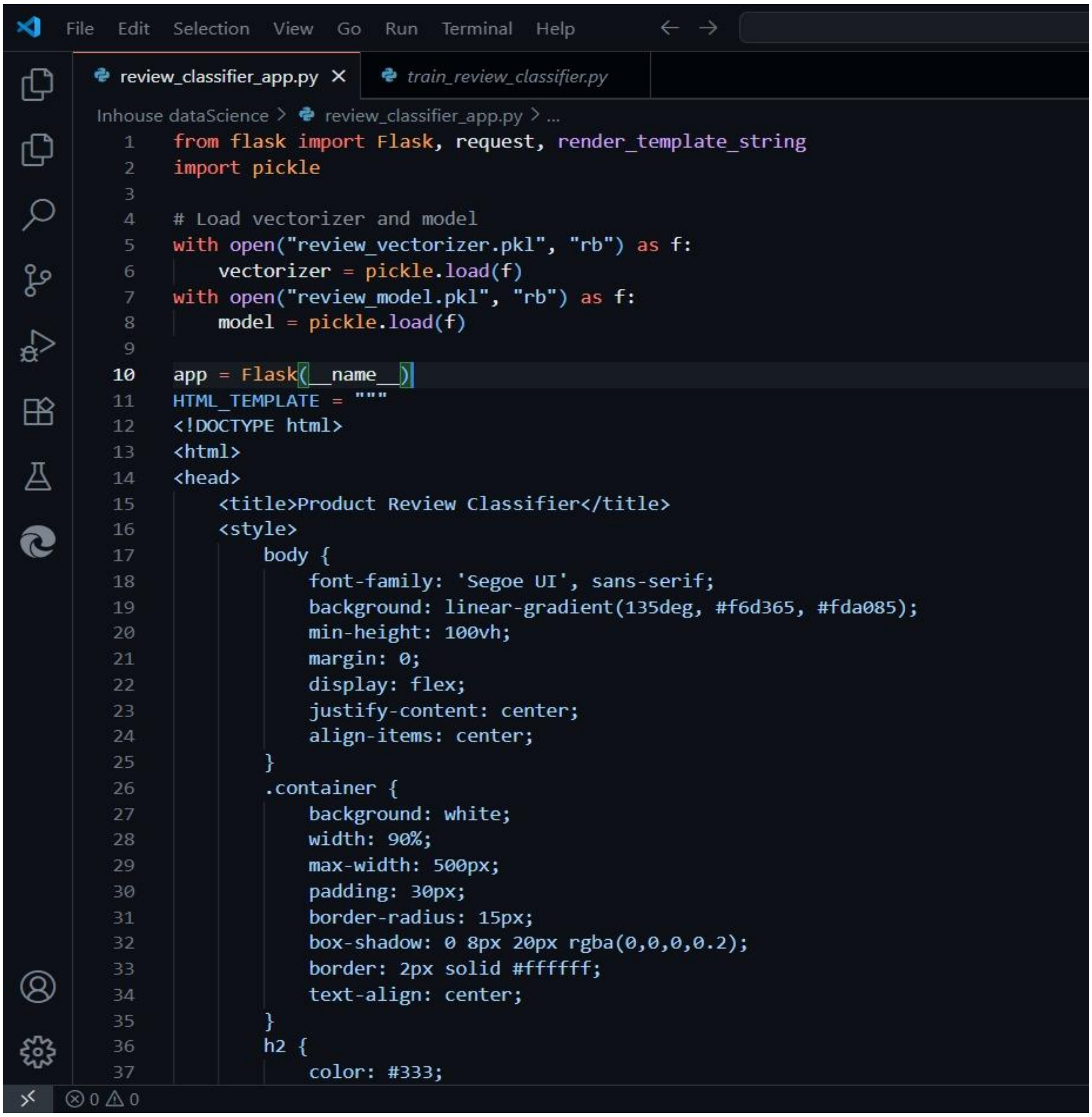


```python
df = pd.DataFrame(data)
# 🔸 Split data
X_train, X_test, y_train, y_test = train_test_split(df["review"], df["label"], test_size=0.2, random_state=42)
# 🔸 Vectorize
vectorizer = TfidfVectorizer(stop_words='english')
X_train_vec = vectorizer.fit_transform(X_train)
```

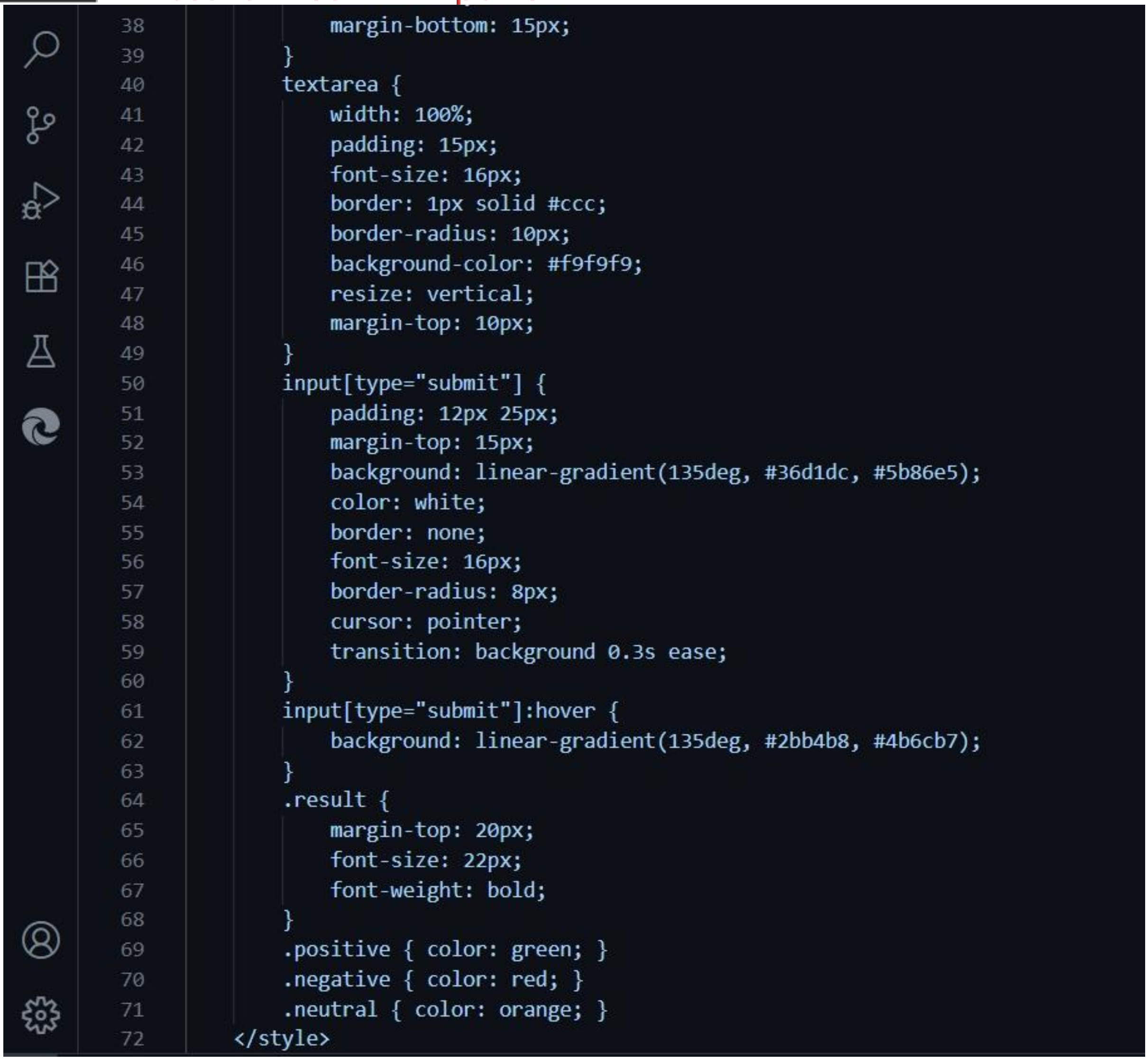


```python
# 🔹 Train model
model = LogisticRegression()
model.fit(X_train_vec, y_train)
# 🔹 Save vectorizer and model
with open("review_vectorizer.pkl", "wb") as f:
    pickle.dump(vectorizer, f)
with open("review_model.pkl", "wb") as f:
    pickle.dump(model, f)
print("✅ Review classifier model saved as 'review_model.pkl' and 'review_vectorizer.pkl'")
```

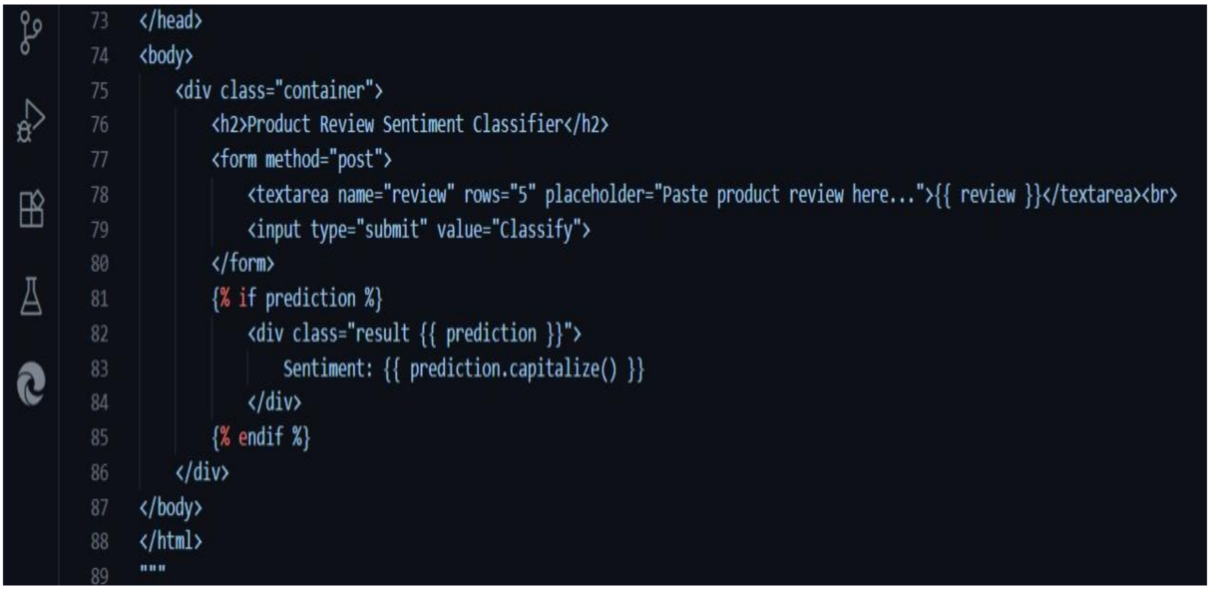
Flask Code:


File
Edit
Selection
View
Go
Run
Terminal
Help
review_classifier_app.py
train_review_classifier.py
Inhouse dataScience > review_classifier_app.py > ...
⊗ 0 ⚠ 0

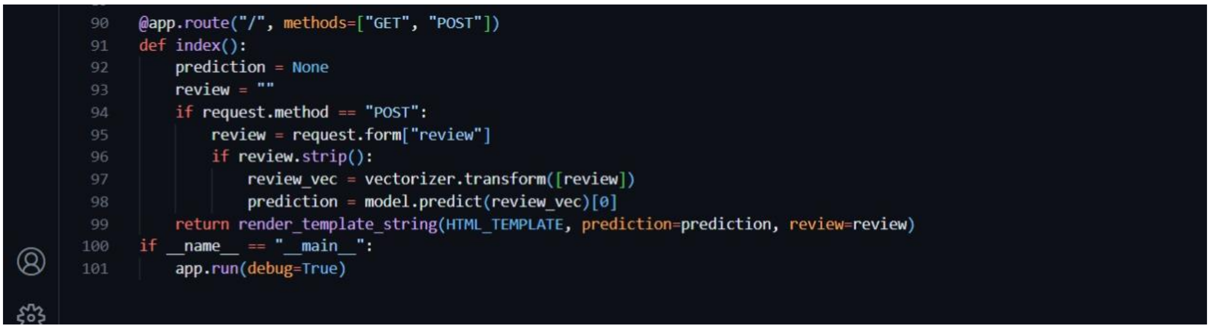
```python
from flask import Flask, request, render_template_string
import pickle

# Load vectorizer and model
with open("review_vectorizer.pkl", "rb") as f:
    vectorizer = pickle.load(f)
with open("review_model.pkl", "rb") as f:
    model = pickle.load(f)

app = Flask(__name__)
HTML_TEMPLATE = """
<!DOCTYPE html>
<html>
<head>
    <title>Product Review Classifier</title>
    <style>
        body {
            font-family: 'Segoe UI', sans-serif;
            background: linear-gradient(135deg, #f6d365, #fda085);
            min-height: 100vh;
            margin: 0;
            display: flex;
            justify-content: center;
            align-items: center;
        }
        .container {
            background: white;
            width: 90%;
            max-width: 500px;
            padding: 30px;
            border-radius: 15px;
            box-shadow: 0 8px 20px rgba(0,0,0,0.2);
            border: 2px solid #ffffff;
            text-align: center;
        }
        h2 {
            color: #333;
```



```
            margin-bottom: 15px;
        }
        textarea {
            width: 100%;
            padding: 15px;
            font-size: 16px;
            border: 1px solid #ccc;
            border-radius: 10px;
            background-color: #f9f9f9;
            resize: vertical;
            margin-top: 10px;
        }
        input[type="submit"] {
            padding: 12px 25px;
            margin-top: 15px;
            background: linear-gradient(135deg, #36d1dc, #5b86e5);
            color: white;
            border: none;
            font-size: 16px;
            border-radius: 8px;
            cursor: pointer;
            transition: background 0.3s ease;
        }
        input[type="submit"]:hover {
            background: linear-gradient(135deg, #2bb4b8, #4b6cb7);
        }
        .result {
            margin-top: 20px;
            font-size: 22px;
            font-weight: bold;
        }
        .positive { color: green; }
        .negative { color: red; }
        .neutral { color: orange; }
    </style>
```

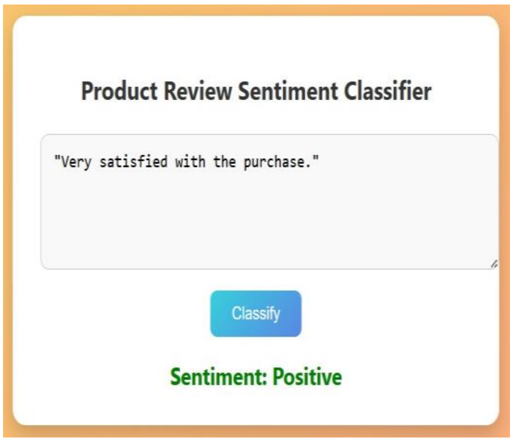


```
</head>
<body>
    <div class="container">
        <h2>Product Review Sentiment Classifier</h2>
        <form method="post">
            <textarea name="review" rows="5" placeholder="Paste product review here...">{{ review }}</textarea><br>
            <input type="submit" value="Classify">
        </form>
        {% if prediction %}
            <div class="result {{ prediction }}">
                Sentiment: {{ prediction.capitalize() }}
            </div>
        {% endif %}
    </div>
</body>
</html>
"""
```



```
@app.route("/", methods=["GET", "POST"])
def index():
    prediction = None
    review = ""
    if request.method == "POST":
        review = request.form["review"]
        if review.strip():
            review_vec = vectorizer.transform([review])
            prediction = model.predict(review_vec)[0]
    return render_template_string(HTML_TEMPLATE, prediction=prediction, review=review)
if __name__ == "__main__":
    app.run(debug=True)
```

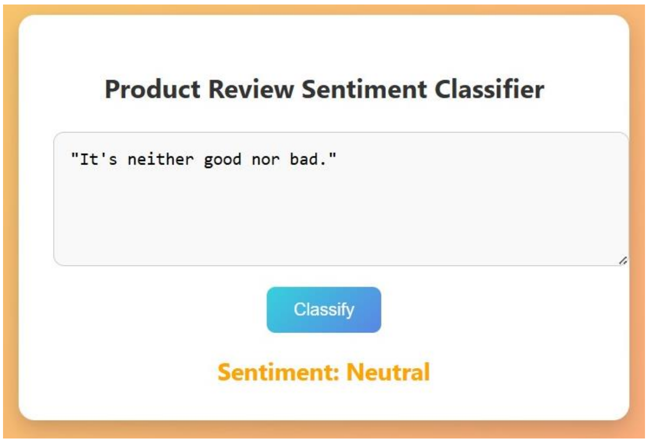
Output:



Product Review Sentiment Classifier

"Very satisfied with the purchase."

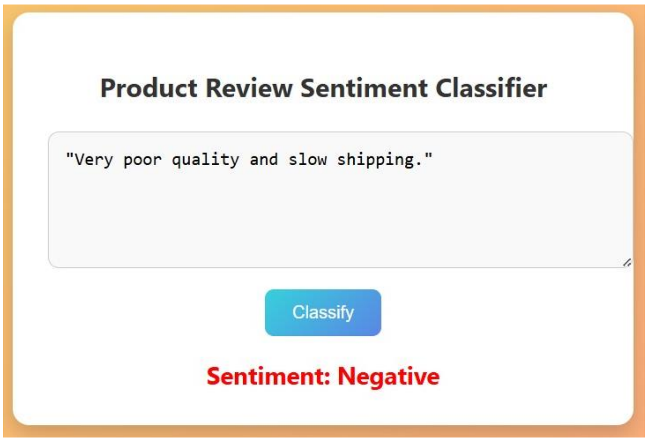
Sentiment: Positive



Product Review Sentiment Classifier

"It's neither good nor bad."

Sentiment: Neutral



Product Review Sentiment Classifier

"Very poor quality and slow shipping."

Sentiment: Negative