Product Review

December 18, 2024

[1]:
```
import pandas as pd
from transformers import T5ForConditionalGeneration, T5Tokenizer, pipeline
import matplotlib.pyplot as plt
import os
import torch
from tabulate import tabulate
```

[2]:
```
device = 'cuda' if torch.cuda.is_available() else 'cpu'
print(device)
import warnings
warnings.filterwarnings('ignore')
```

```
cuda
```

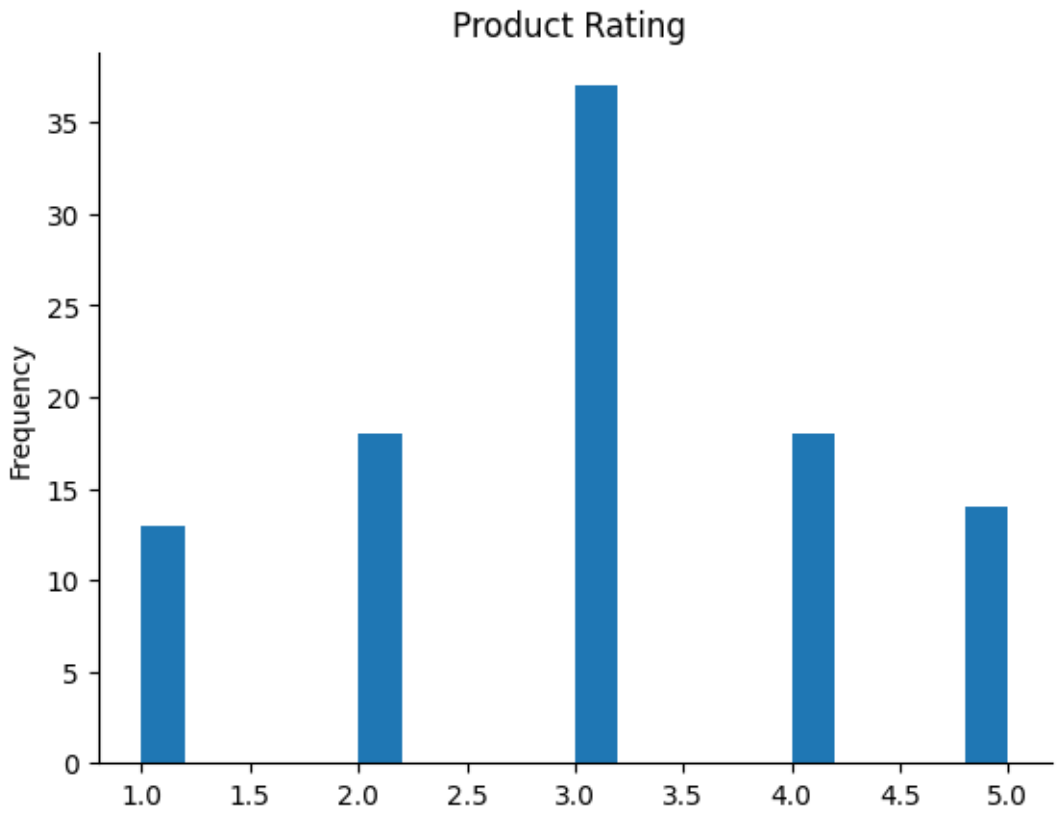
[3]:
```
df = pd.read_csv('/content/enhanced_product_reviews_dataset_100.csv')
```

[4]:
```
df.head()
```

[4]:

	review_id	product_name	review_text	rating	sentiment
0	R001	Headphones	The sound quality is crisp and clear, with gre…	4	Positive
1	R002	Camera	The camera works fine for beginners but not su…	3	Neutral
2	R003	Camera	The camera is bulky and the image quality is n…	1	Negative
3	R004	Tablet	The tablet freezes often and has limited stora…	1	Negative
4	R005	Laptop	The laptop is okay for its price but lacks adv…	3	Neutral

[5]:
```
# @title Plotting the rating

from matplotlib import pyplot as plt
df['rating'].plot(kind='hist', bins=20, title='Product Rating ')
plt.gca().spines[['top', 'right',]].set_visible(False)
```



##Text Summarization

[6]:
```
summarizer = pipeline("summarization", model="t5-base", device = device)
```

```
config.json:   0%|          | 0.00/1.21k [00:00<?, ?B/s]
model.safetensors:   0%|          | 0.00/892M [00:00<?, ?B/s]
generation_config.json:   0%|          | 0.00/147 [00:00<?, ?B/s]
spiece.model:   0%|          | 0.00/792k [00:00<?, ?B/s]
tokenizer.json:   0%|          | 0.00/1.39M [00:00<?, ?B/s]
```

[7]:
```
df['review_length'] = df['review_text'].apply(lambda x: len(str(x).split()))


average_length = df['review_length'].mean()
max_length = df['review_length'].max()
min_length = df['review_length'].min()

print(f"Average review length: {average_length:.2f} words")
print(f"Maximum review length: {max_length} words")
```

```
print(f"Minimum review length: {min_length} words")
```

```
Average review length: 10.88 words
Maximum review length: 13 words
Minimum review length: 9 words
```

[8]:
```
summary_min_length = int(average_length * 0.3)
summary_max_length = int(average_length * 0.7)
```

[9]:
```
print(f"Suggested summary min_length: {summary_min_length} words")
print(f"Suggested summary max_length: {summary_max_length} words")
```

```
Suggested summary min_length: 3 words
Suggested summary max_length: 7 words
```

[10]:
```
def summarize_review(text):
    # Handle empty or short reviews directly
    if not isinstance(text, str) or len(text.split()) < 5:
        return text

    summary = summarizer(text, max_length=10, min_length=3, do_sample=False)
    return summary[0]['summary_text']
```

[11]:
```
df['summary'] = df['review_text'].apply(summarize_review)
print(df[['review_id', 'review_text', 'summary']].head())
```

```
You seem to be using the pipelines sequentially on GPU. In order to maximize
efficiency please use a dataset

  review_id                                        review_text  \
0      R001  The sound quality is crisp and clear, with gre…
1      R002  The camera works fine for beginners but not su…
2      R003  The camera is bulky and the image quality is n…
3      R004  The tablet freezes often and has limited stora…
4      R005  The laptop is okay for its price but lacks adv…

                                          summary
0         the sound quality is crisp and clear, with
1  the camera works fine for beginners but not su…
2          the camera is bulky and the image quality
3  tablet freezes often and has limited storage c…
4          the laptop is okay for its price but lack
```

[12]:
```
df_summary = df.copy()
df_summary['summary'].head(10)
```

[12]:
```
0           the sound quality is crisp and clear, with
1    the camera works fine for beginners but not su…
```

```
2            the camera is bulky and the image quality
3    tablet freezes often and has limited storage c…
4            the laptop is okay for its price but lack
5                                        lack of bass.
6                                        lack of bass.
7            the phone lags frequently and the battery
8                                        lack of bass.
9    the phone has an excellent camera and smooth p…
Name: summary, dtype: object
```

0.1 Sentiment Analysis

[13]:
```
sentiment_analyzer = pipeline(
    "text-classification",
    model="cardiffnlp/twitter-roberta-base-sentiment",
    tokenizer="cardiffnlp/twitter-roberta-base-sentiment", device = device)
```

```
config.json:   0%|          | 0.00/747 [00:00<?, ?B/s]
pytorch_model.bin:   0%|          | 0.00/499M [00:00<?, ?B/s]
vocab.json:   0%|          | 0.00/899k [00:00<?, ?B/s]
merges.txt:   0%|          | 0.00/456k [00:00<?, ?B/s]
special_tokens_map.json:   0%|          | 0.00/150 [00:00<?, ?B/s]
```

[14]:
```
def classify_sentiment_with_score(text):
    if not isinstance(text, str) or len(text) == 0:  # Handle empty reviews
        return "Neutral", 0.0

    result = sentiment_analyzer(text)[0]
    label = result['label']
    score = result['score']

    if label == "LABEL_0":  # Negative
        return "Negative", score
    elif label == "LABEL_1":  # Neutral
        return "Neutral", score
    elif label == "LABEL_2":  # Positive
        return "Positive", score
```

[15]:
```
df_sentiment = df.copy()
```

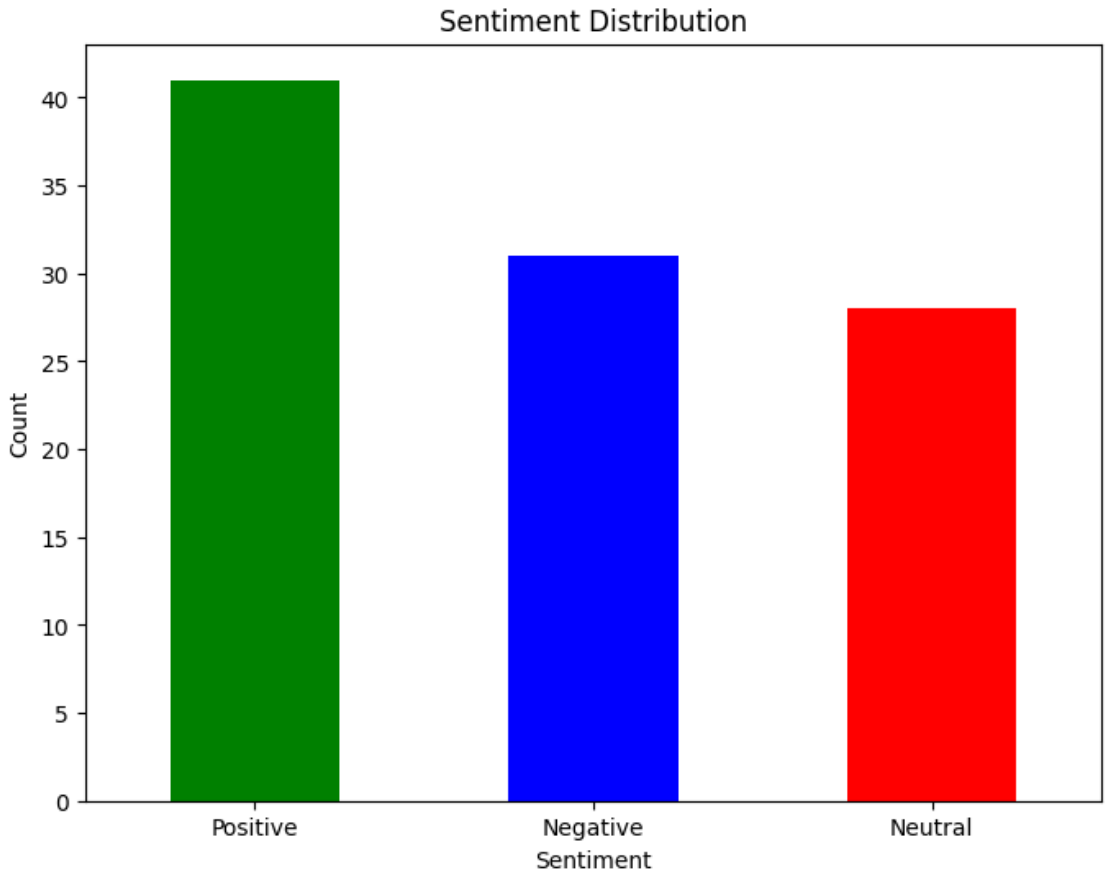
[16]:
```
df_sentiment[['predicted_sentiment', 'confidence']] =␣
 ↪df_sentiment['review_text'].apply(
    lambda x: pd.Series(classify_sentiment_with_score(x)))

confidence_threshold = 0.6
```

```
df_sentiment['flag_low_confidence'] = df_sentiment['confidence'] <␣
 ↪confidence_threshold
```

[17]:
```
def compare_sentiment_with_rating(row):
    rating = row['rating']
    sentiment = row['predicted_sentiment']

    if rating >= 4 and sentiment != "Positive":
        return "Mismatch"
    elif rating <= 2 and sentiment != "Negative":
        return "Mismatch"
    return "Match"
```

[18]:
```
df_sentiment['rating_sentiment_match'] = df_sentiment.
 ↪apply(compare_sentiment_with_rating, axis=1)
```

[19]:
```
# Visualize sentiment distribution
def visualize_sentiment_distribution(df):
    sentiment_counts = df['predicted_sentiment'].value_counts()

    plt.figure(figsize=(8, 6))
    sentiment_counts.plot(kind='bar', color=['green', 'blue', 'red'])
    plt.title("Sentiment Distribution")
    plt.xlabel("Sentiment")
    plt.ylabel("Count")
    plt.xticks(rotation=0)
    plt.show()

visualize_sentiment_distribution(df_sentiment)
```



[20]:
```
mismatches = df_sentiment[df_sentiment['rating_sentiment_match'] == "Mismatch"]
```

[21]:
```
def generate_report(df, mismatches, accuracy):
    print("\nSentiment Analysis Report")
    print("========================\n")
    print(f"Total reviews analyzed: {len(df)}")
    print(f"Accuracy: {accuracy:.2f}%\n")

    # Sentiment distribution
    sentiment_counts = df['predicted_sentiment'].value_counts()
    print("Sentiment Distribution:")
    print(sentiment_counts)
    print("\n")

    # Mismatched reviews
    print(f"Mismatched Reviews ({len(mismatches)}):")
    print(mismatches[['review_id', 'review_text', 'rating',␣
 ↪'predicted_sentiment']].to_string(index=False))
    print("\n")
```

```
    # Low-confidence predictions
    low_confidence = df[df['flag_low_confidence']]
    print(f"Low-Confidence Predictions ({len(low_confidence)}):")
    print(low_confidence[['review_id', 'review_text', 'confidence']].
 ↪to_string(index=False))
    print("\n")
```

[22]:
```
correct_predictions = (df_sentiment['predicted_sentiment'] ==␣
 ↪df_sentiment['sentiment']).sum()
total_reviews = len(df_sentiment)
accuracy = correct_predictions / total_reviews * 100

generate_report(df_sentiment, mismatches, accuracy)


print(f"Sentiment analysis accuracy: {accuracy:.2f}%")
```

```
Sentiment Analysis Report
=========================

Total reviews analyzed: 100
Accuracy: 91.00%

Sentiment Distribution:
predicted_sentiment
Positive    41
Negative    31
Neutral     28
Name: count, dtype: int64


Mismatched Reviews (0):
Empty DataFrame
Columns: [review_id, review_text, rating, predicted_sentiment]
Index: []


Low-Confidence Predictions (28):
review_id
review_text  confidence
     R002 The camera works fine for beginners but not suitable for
professionals.    0.470715
     R005           The laptop is okay for its price but lacks advanced
features.    0.454449
     R006             The headphones are fine for casual listening but lack
```

```
bass.    0.465944
     R007             The headphones are fine for casual listening but lack
bass.    0.465944
     R009             The headphones are fine for casual listening but lack
bass.    0.465944
     R013           The laptop is okay for its price but lacks advanced
features.    0.454449
     R015                        The tablet works well but feels a bit
overpriced.    0.414557
     R016             The headphones are fine for casual listening but lack
bass.    0.465944
     R019                        The tablet works well but feels a bit
overpriced.    0.414557
     R022                        The tablet works well but feels a bit
overpriced.    0.414557
     R030             The headphones are fine for casual listening but lack
bass.    0.465944
     R033 The camera works fine for beginners but not suitable for
professionals.    0.470715
     R038                        The tablet works well but feels a bit
overpriced.    0.414557
     R039 The camera works fine for beginners but not suitable for
professionals.    0.470715
     R052 The camera works fine for beginners but not suitable for
professionals.    0.470715
     R060           The laptop is okay for its price but lacks advanced
features.    0.454449
     R063           The laptop is okay for its price but lacks advanced
features.    0.454449
     R064 The camera works fine for beginners but not suitable for
professionals.    0.470715
     R069             The headphones are fine for casual listening but lack
bass.    0.465944
     R070                        The tablet works well but feels a bit
overpriced.    0.414557
     R073             The headphones are fine for casual listening but lack
bass.    0.465944
     R075                        The tablet works well but feels a bit
overpriced.    0.414557
     R077             The headphones are fine for casual listening but lack
bass.    0.465944
     R085           The laptop is okay for its price but lacks advanced
features.    0.454449
     R087             The headphones are fine for casual listening but lack
bass.    0.465944
     R090                        The tablet works well but feels a bit
overpriced.    0.414557
     R092           The laptop is okay for its price but lacks advanced
```

```
features.    0.454449
     R096 The camera works fine for beginners but not suitable for
professionals.    0.470715


Sentiment analysis accuracy: 91.00%
```

The sentiment analysis model achieved an impressive **91.00% accuracy** across 100 product reviews, showcasing its strong ability to correctly classify sentiments. Out of all the reviews, there were **no mismatched predictions** between the ratings and the predicted sentiments, reflecting a high level of alignment between the model's output and the actual sentiment labels. The sentiment distribution shows a balanced mix of classifications: **41 positive**, **31 negative**, and **28 neutral**, indicating the model's robustness in capturing diverse customer sentiments.

However, the analysis highlighted **28 low-confidence predictions**, which primarily involved reviews with nuanced or mixed sentiments. For instance, phrases such as "The camera works fine for beginners but not suitable for professionals" received low confidence scores (e.g., 0.47), suggesting that the model found it challenging to definitively classify borderline sentiments.

0.2 Synthetic Review Generation

[23]:
```
df.head()
```

[23]:
```
  review_id product_name                                        review_text  \
0      R001   Headphones  The sound quality is crisp and clear, with gre…
1      R002       Camera  The camera works fine for beginners but not su…
2      R003       Camera  The camera is bulky and the image quality is n…
3      R004       Tablet  The tablet freezes often and has limited stora…
4      R005       Laptop  The laptop is okay for its price but lacks adv…

   rating sentiment  review_length  \
0       4  Positive             11
1       3   Neutral             11
2       1  Negative             12
3       1  Negative              9
4       3   Neutral             11

                                            summary
0        the sound quality is crisp and clear, with
1  the camera works fine for beginners but not su…
2         the camera is bulky and the image quality
3  tablet freezes often and has limited storage c…
4         the laptop is okay for its price but lack
```

[24]:
```
from transformers import AutoModelForCausalLM, AutoTokenizer
```

[25]:
```
# Initialize Flan-T5 model and tokenizer
```

```
model_name = "google/flan-t5-xl"  # Use a larger, more advanced model like␣
↪Flan-T5-xl
model = T5ForConditionalGeneration.from_pretrained(model_name)
tokenizer = T5Tokenizer.from_pretrained(model_name)
```

```
config.json:   0%|          | 0.00/1.44k [00:00<?, ?B/s]
model.safetensors.index.json:   0%|          | 0.00/53.0k [00:00<?, ?B/s]
Downloading shards:   0%|          | 0/2 [00:00<?, ?it/s]
model-00001-of-00002.safetensors:   0%|          | 0.00/9.45G [00:00<?, ?B/s]
model-00002-of-00002.safetensors:   0%|          | 0.00/1.95G [00:00<?, ?B/s]
Loading checkpoint shards:   0%|          | 0/2 [00:00<?, ?it/s]
generation_config.json:   0%|          | 0.00/147 [00:00<?, ?B/s]
tokenizer_config.json:   0%|          | 0.00/2.54k [00:00<?, ?B/s]
spiece.model:   0%|          | 0.00/792k [00:00<?, ?B/s]
special_tokens_map.json:   0%|          | 0.00/2.20k [00:00<?, ?B/s]
tokenizer.json:   0%|          | 0.00/2.42M [00:00<?, ?B/s]
You are using the default legacy behaviour of the <class
'transformers.models.t5.tokenization_t5.T5Tokenizer'>. This is expected, and
simply means that the `legacy` (previous) behavior will be used so nothing
changes for you. If you want to use the new behaviour, set `legacy=False`. This
should only be set if you understand what it means, and thoroughly read the
reason why this was added as explained in
https://github.com/huggingface/transformers/pull/24565
```

[26]:
```
text_generator = pipeline("text2text-generation", model=model,␣
↪tokenizer=tokenizer, device = device)
```

[27]:
```
def generate_synthetic_review(prompt, max_length=100, num_return_sequences=1,␣
↪temperature=0.7):
    inputs = tokenizer(prompt, return_tensors="pt", truncation=True,␣
↪padding=True).to(device)
    outputs = model.generate(
        **inputs,
        max_length=max_length,
        num_return_sequences=num_return_sequences,
        do_sample=True,
        temperature=temperature,
        top_k=50,
        top_p=0.95,
        repetition_penalty=2.0
    )
```

```
    return [tokenizer.decode(output, skip_special_tokens=True) for output in␣
↪outputs]
```

[28]:
```
def verify_sentiment(reviews, target_sentiment):
    verified_reviews = []
    sentiment_analyzer = pipeline("sentiment-analysis", model = 'cardiffnlp/
↪twitter-roberta-base-sentiment', device = device)

    for review in reviews:
        sentiment_result = sentiment_analyzer(review)[0]
        verified_reviews.append({
            "review": review,
            "predicted_sentiment": sentiment_result['label'],
            "confidence": sentiment_result['score'],
            "target_sentiment": target_sentiment,
            "sentiment_match": sentiment_result['label'].lower() ==␣
↪target_sentiment.lower()
        })
    return verified_reviews
```

[29]:
```
def generate_product_specific_reviews(df):
    synthetic_reviews = {}

    for _, row in df.iterrows():
        product = row['product_name']
        rating = row['rating']

        # Generating prompts based on product and rating
        if rating == 5:
            prompt = f"Write an enthusiastic and detailed positive review for a␣
↪5-star product about {product}. Highlight its best features, explain why it␣
↪exceeds expectations, and share your overall satisfaction. Mention aspects␣
↪such as value for money, quality, and performance in your review."
            synthetic_reviews[f"{product}_5star"] =␣
↪generate_synthetic_review(prompt, max_length=100, num_return_sequences=3)

        elif rating == 1:
            prompt = f"Write a detailed and constructive negative review for a␣
↪1-star product about {product}. Mention the major drawbacks and issues you␣
↪encountered, such as poor performance, bad quality, or lack of features.␣
↪Provide specific examples of why the product failed to meet expectations and␣
↪your overall disappointment."
            synthetic_reviews[f"{product}_1star"] =␣
↪generate_synthetic_review(prompt, max_length=150, num_return_sequences=3)

    return synthetic_reviews
```

[30]:
```
synthetic_reviews = generate_product_specific_reviews(df)
```

[31]:
```
display(synthetic_reviews)
```

```
{'Camera_1star': ["Honestly I was very disappointed, since the camera has been␣
 ↪released, but I didn't expect the quality to be this low. Apparently the␣
 ↪sensor is not so good as it's not even close to being good. I took the phone␣
 ↪for the first time with the camera on and had some issues. The shutter speed␣
 ↪would get terribly slow and it took a while to take a picture. It seemed to␣
 ↪take forever to click the button. Unless you have a lot of patience, you may␣
 ↪as well keep looking.",
  "This is a very cheap and bad camera. It has the worst quality pictures. I␣
 ↪have seen many camera that look much better than this. I haven't had this kind␣
 ↪of problem with any other camera I have used.",
  "I don't understand how you can make a camera with such poor quality. The␣
 ↪image was pixelated and the camera did not work at all. Also, the flash was␣
 ↪horrible. It didn't even turn on."],
 'Tablet_1star': ["I bought this tablet for my son and I am a big fan of tablets.
 ↪ I loved it when I was little but now I hate it. He has to use an extra␣
 ↪charger every time he uses it. The screen is always blurry. I tried it on a␣
 ↪normal table but it didn't work at all. The main thing is the battery life.␣
 ↪The unit is very small and there is no storage space. If you have lots of apps␣
 ↪on it then that is fine. But if you have a lot of movies or music files then␣
 ↪it is a huge problem. There are just too many options to choose from.",
  "This was a horrible tablet. I used it for two years and it just died. The␣
 ↪screen didn't work at all. It was always blue. No response to my emails or␣
 ↪phone calls. I tried it on a different computer. The cpu speed was much better.
 ↪ I got it to run Windows 8 on the same hardware. I am not going to use this␣
 ↪tablet.",
  'I don\'t care how nice the package is if the product doesn\'t meet␣
 ↪expectations. The most disappointing thing about this tablet was that the␣
 ↪display screen didn\'t seem to be working at all. The ""screen"" on the back␣
 ↪of the unit was so small and blurry that I couldn\'t read anything on it. It␣
 ↪also took awhile to turn on and shut off. It looked like the battery was dead␣
 ↪when I turned it on. I have no idea why the manufacturer would make a phone␣
 ↪with such poor quality for this price.'],
 'Phone_5star': ['Great phone. Just what I was looking for.',
  'Best phone ever',
  'I love it. It has the best screen I have ever seen. I am so impressed with␣
 ↪this phone. The design and quality of the build are fantastic. Great product.
 ↪'],
 'Laptop_5star': ['Best Laptop I have ever had. The price is great and the␣
 ↪quality is excellent.',
  "This is a great product. I really like it. It is very lightweight and feels␣
 ↪pretty sturdy, and has a decent battery life. I've only had it for a couple of␣
 ↪weeks now but it has held up very well so far.",
```

```
 "Great Laptop! I've been using a Dell Inspiron 15 for over three years and I␣
↪have yet to find anything that isn't fantastic. I think the one thing that␣
↪really sets it apart from other laptops is that it has an external hard drive.␣
↪That is great because sometimes you need to back up data and I love being able␣
↪to back up and save my files. But what makes the Dell Inspiron 15 so special␣
↪is the incredible battery life"],
'Headphones_1star': ['I purchased a pair of these for my daughter. When I got␣
↪them they were extremely uncomfortable and had poor sound quality. The earcups␣
↪are padded and are ok but the headband is very thin and it has no padding on␣
↪the inside. The headphone jack is small and there is no way to insert it into␣
↪the speaker. I bought them on Amazon and had a problem with one of the␣
↪speakers and had to replace it. I also tried to put them in my pocket but they␣
↪would not fit in the ear buds. I would not recommend this product to anyone.',
 "I got this for my ps3 and it's not working. It's not even an option in the␣
↪options. Basically what you get is an old design that won't work.",
 "I purchased this product because it was a good deal and I didn't know what I␣
↪was paying for. The sound quality was horrible. It was so low that it was␣
↪difficult to hear the music. The design was horrible. It's like a pair of old␣
↪and cheap headphones that doesn't fit well. I am glad I didn't buy this␣
↪product."],
'Laptop_1star': ["I'm a huge fan of ASUS, but this laptop is terrible! The␣
↪battery life was very disappointing. Almost slept through the entire day. And␣
↪the graphics are poor. It has a bad design that looks like a cheap knock-off.␣
↪I feel like the price was too high for a decent gaming laptop. I think it was␣
↪around $800.",
 "If you have no idea about technology and all it's wonders, don't buy this␣
↪product. It will cost you more than you think if you buy it online. The UI is␣
↪confusing and the keyboard doesn't work at all.",
 'I was so excited to get a new laptop and have been waiting for it for weeks.␣
↪I finally got it and I am very disappointed. The screen is too old and the␣
↪sound is not good. The battery life is also not up to par. It has a lot of␣
↪problems with connectivity and performance.'],
'Camera_5star': ["This is a very good camera. I bought it after buying another␣
↪refurbished one. And this camera is better than the refurbished. It has good␣
↪sensitivity, and high ISO sensitivity. But the biggest advantage is that it's␣
↪smaller. So you can use it anywhere. I like to keep it in my pocket.",
 "It's awesome! I bought this as a birthday gift and I was so happy that I have␣
↪it. The pictures are great, the flash is amazing, the lens is just what I␣
↪needed. It is easy to use and it has great controls. This camera is perfect␣
↪for beginners.",
 'Very good camera.'],
'Phone_1star': ['I had to return it because it was too small. They gave me a␣
↪replacement and the screen is smaller than what I wanted. They refunded me for␣
↪the new one.',
 "I bought this phone because it was supposedly good for watching movies. It␣
↪didn't work and the screen broke within the first week. They didn't even offer␣
↪to replace it.",
```

```
 "I was really excited to get a new phone. But I only have one complaint and␣
↪that is the screen size. It is too small for me to read anything at all and␣
↪it's hard to see what I'm looking at when I'm using my fingers to scroll␣
↪through websites. The only thing I can say about the screen is that it is not␣
↪very bright, so you're kind of missing out on everything."],
'Tablet_5star': ['Great tablet. The battery life is good for watching videos␣
↪and a lot of other things. The screen resolution is perfect for viewing videos␣
↪and reading. The touchpad is very responsive. It has 8GB of internal storage␣
↪and has a microSD card slot.',
 'Good product',
 "i think it's great"],
'Headphones_5star': ["Great sound quality and bass. I like the fact that the␣
↪ear cups are soft and won't fall off my head.",
 "Great headphone. I've tried many over the years and this is the best. It has␣
↪so much bass and clarity and is very comfortable. Great sound quality.",
 'i loved the sound quality of these headphones. they were very comfortable and␣
↪fit my head perfectly.']}
```

0.3 Sentiment Consistency

[32]:
```python
sentiment_mapping = {
    "LABEL_0": "Negative",
    "LABEL_1": "Neutral",
    "LABEL_2": "Positive"
}

all_results = []

for product, reviews in synthetic_reviews.items():
    sentiment_type = "positive" if "5star" in product else "negative"
    verified = verify_sentiment(reviews, sentiment_type.upper())

    for idx, result in enumerate(verified):
        predicted_sentiment = sentiment_mapping.
 ↪get(result['predicted_sentiment'], result['predicted_sentiment'])

        sentiment_match = predicted_sentiment.lower() ==␣
 ↪result['target_sentiment'].lower()

        all_results.append({
            'review': result['review'],
            'predicted_sentiment': predicted_sentiment,
            'confidence': result['confidence'],
            'target_sentiment': result['target_sentiment'],
            'sentiment_match': sentiment_match,
            'product': product.split('_')[0],
            'rating': 5 if sentiment_type == "positive" else 1,
```

```
            'review_id': f"R{str(idx+1).zfill(3)}"
        })
```

[33]:
```
all_results_df = pd.DataFrame(all_results)
```

[34]:
```
all_results_df.head()
```

[34]:

	review	predicted_sentiment	confidence	target_sentiment	sentiment_match	product	rating	review_id
0	Honestly I was very disappointed, since the ca…	Negative	0.929525	NEGATIVE	True	Camera	1	R001
1	This is a very cheap and bad camera. It has th…	Negative	0.836290	NEGATIVE	True	Camera	1	R002
2	I don't understand how you can make a camera w…	Negative	0.977102	NEGATIVE	True	Camera	1	R003
3	I bought this tablet for my son and I am a big…	Negative	0.559030	NEGATIVE	True	Tablet	1	R001
4	This was a horrible tablet. I used it for two …	Negative	0.888483	NEGATIVE	True	Tablet	1	R002

[35]:
```
all_results_df['sentiment_match'].value_counts()
```

[35]:
```
sentiment_match
True    30
Name: count, dtype: int64
```

[36]:
```
mismatched_reviews = all_results_df[all_results_df['sentiment_match'] == False]


print("Mismatched Reviews:")
mismatched_reviews[['review_id', 'review', 'predicted_sentiment',␣
 ↪'target_sentiment', 'confidence']]
```

```
Mismatched Reviews:
```

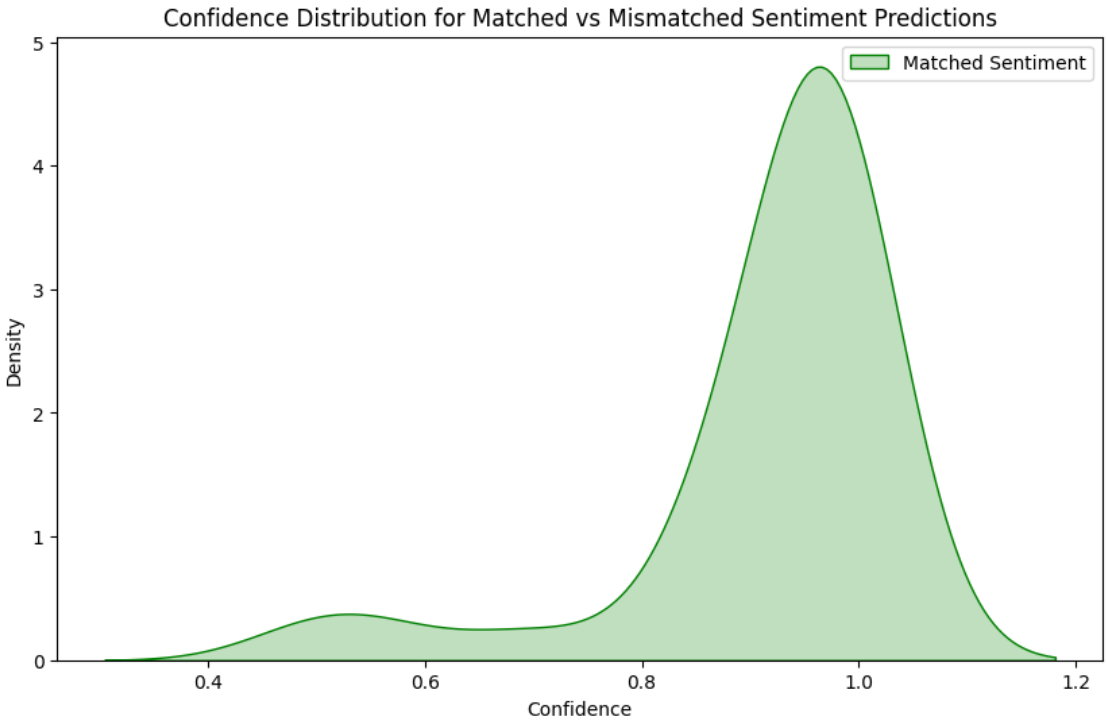
[36]:
```
Empty DataFrame
Columns: [review_id, review, predicted_sentiment, target_sentiment, confidence]
Index: []
```

[37]:
```
import seaborn as sns
```

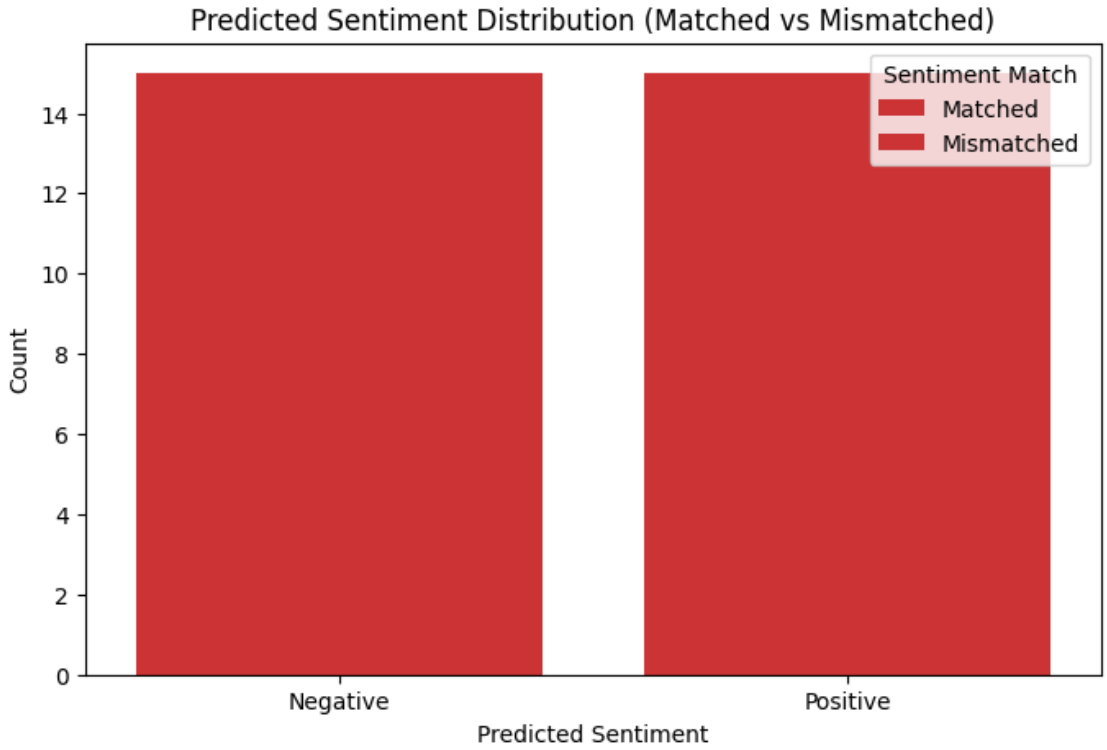
[38]:
```
matched_reviews = all_results_df[all_results_df['sentiment_match'] == True]
mismatched_reviews = all_results_df[all_results_df['sentiment_match'] == False]
```

[39]:
```
plt.figure(figsize=(10, 6))
```

```
sns.kdeplot(matched_reviews['confidence'], label='Matched Sentiment',␣
 ↪shade=True, color='g')
sns.kdeplot(mismatched_reviews['confidence'], label='Mismatched Sentiment',␣
 ↪shade=True, color='r')
plt.title('Confidence Distribution for Matched vs Mismatched Sentiment␣
 ↪Predictions')
plt.xlabel('Confidence')
plt.ylabel('Density')
plt.legend()
plt.show()
```

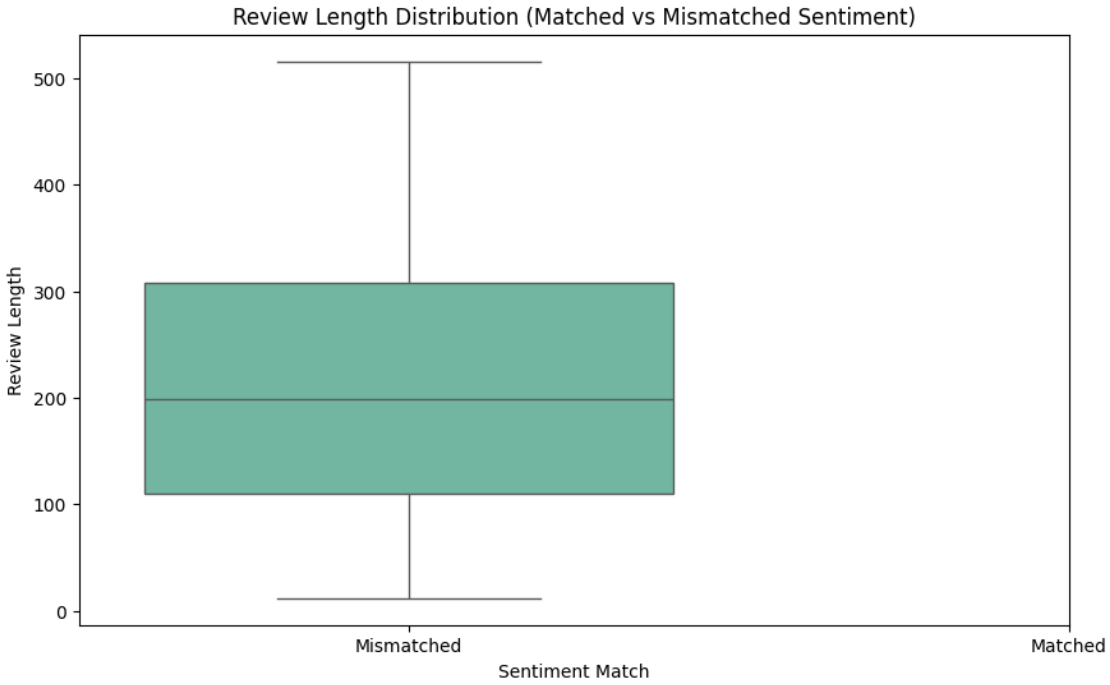


[40]:
```
plt.figure(figsize=(8, 5))
sns.countplot(data=all_results_df, x='predicted_sentiment',␣
 ↪hue='sentiment_match', palette='Set1')
plt.title('Predicted Sentiment Distribution (Matched vs Mismatched)')
plt.xlabel('Predicted Sentiment')
plt.ylabel('Count')
plt.legend(title="Sentiment Match", labels=["Matched", "Mismatched"])
plt.show()
```



[41]:
```
product_sentiment_inconsistency = mismatched_reviews.groupby('product').size().
  ↪reset_index(name='mismatches')
product_sentiment_inconsistency_sorted = product_sentiment_inconsistency.
  ↪sort_values('mismatches', ascending=False)
```

[42]:
```
all_results_df['review_length'] = all_results_df['review'].apply(len)
plt.figure(figsize=(10, 6))
sns.boxplot(data=all_results_df, x='sentiment_match', y='review_length',␣
  ↪palette='Set2')
plt.title('Review Length Distribution (Matched vs Mismatched Sentiment)')
plt.xlabel('Sentiment Match')
plt.ylabel('Review Length')
plt.xticks([0, 1], ['Mismatched', 'Matched'])
plt.show()
```


Review Length Distribution (Matched vs Mismatched Sentiment)
Review Length
500
400
300
200
100
0
Mismatched
Matched
Sentiment Match

[42]: