0.1 CaseCraft: The Analytics Sprint – Project 14

0.1.1 Brand Sentiment Dashboard

Subheading: Tracking public sentiment across Nike, Adidas, and Puma using NLP and time-series analysis.

0.1.2 Project Goals

- Simulate tweet-level data for three sportswear brands
- Perform sentiment analysis using TextBlob
- Visualize sentiment distribution, polarity trends, and brand comparison
- Build classifier to predict brand from tweet text
- Summarize insights for brand strategy and perception tracking

[7]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from textblob import TextBlob
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

np.random.seed(42)

brands = ['Nike', 'Adidas', 'Puma']
n_tweets = 1500
brand = np.random.choice(brands, n_tweets)
dates = pd.date_range(start='2023-01-01', periods=n_tweets, freq='H')

positive_phrases = ['love', 'great', 'awesome', 'stylish', 'comfortable']
negative_phrases = ['hate', 'bad', 'ugly', 'uncomfortable', 'overpriced']
```

```
neutral_phrases = ['okay', 'fine', 'average', 'meh', 'decent']

def generate_tweet(b):
    sentiment = np.random.choice(['positive', 'negative', 'neutral'], p=[0.4, 0.
↪3, 0.3])
    phrase = np.random.choice({
        'positive': positive_phrases,
        'negative': negative_phrases,
        'neutral': neutral_phrases
    }[sentiment])
    return f"{b} shoes are {phrase}!"

tweets = [generate_tweet(b) for b in brand]

df = pd.DataFrame({
    'brand': brand,
    'timestamp': dates,
    'tweet': tweets
})

df['polarity'] = df['tweet'].apply(lambda x: TextBlob(x).sentiment.polarity)
df['sentiment'] = pd.cut(df['polarity'], bins=[-1, -0.1, 0.1, 1],␣
↪labels=['Negative', 'Neutral', 'Positive'])
df['date'] = df['timestamp'].dt.date
```

```
/tmp/ipython-input-4068823950.py:16: FutureWarning: 'H' is deprecated and will
be removed in a future version, please use 'h' instead.
  dates = pd.date_range(start='2023-01-01', periods=n_tweets, freq='H')
```

[8]:
```
df.head(10)
```

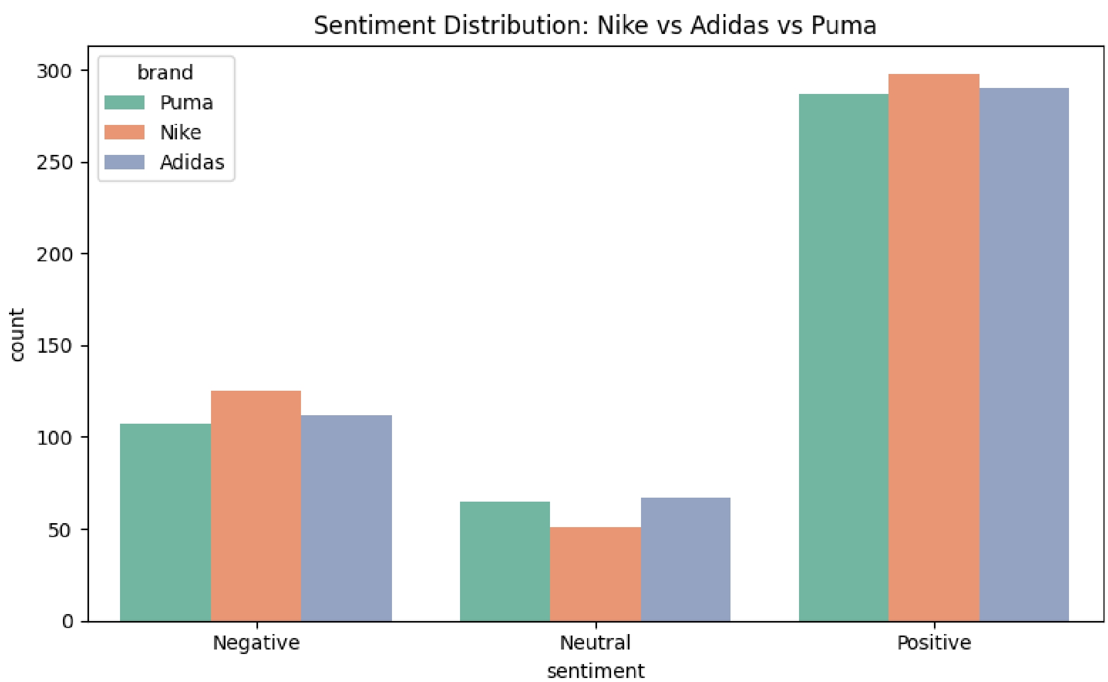
[8]:
```
    brand           timestamp                    tweet  polarity sentiment  \
0    Puma 2023-01-01 00:00:00  Puma shoes are awesome!  1.000000  Positive
1    Nike 2023-01-01 01:00:00     Nike shoes are okay!  0.625000  Positive
2    Puma 2023-01-01 02:00:00     Puma shoes are okay!  0.625000  Positive
3    Puma 2023-01-01 03:00:00  Puma shoes are awesome!  1.000000  Positive
4    Nike 2023-01-01 04:00:00   Nike shoes are decent!  0.208333  Positive
5    Nike 2023-01-01 05:00:00   Nike shoes are decent!  0.208333  Positive
6    Puma 2023-01-01 06:00:00     Puma shoes are hate! -1.000000       NaN
7  Adidas 2023-01-01 07:00:00    Adidas shoes are meh!  0.000000   Neutral
8    Puma 2023-01-01 08:00:00    Puma shoes are great!  1.000000  Positive
9    Puma 2023-01-01 09:00:00    Puma shoes are great!  1.000000  Positive

         date
0  2023-01-01
1  2023-01-01
2  2023-01-01
```

```
3   2023-01-01
4   2023-01-01
5   2023-01-01
6   2023-01-01
7   2023-01-01
8   2023-01-01
9   2023-01-01
```

0.1.3 Sentiment Distribution by Brand

[9]:
```
plt.figure(figsize=(8, 5))
sns.countplot(data=df, x='sentiment', hue='brand', palette='Set2')
plt.title("Sentiment Distribution: Nike vs Adidas vs Puma")
plt.tight_layout()
plt.show()
```

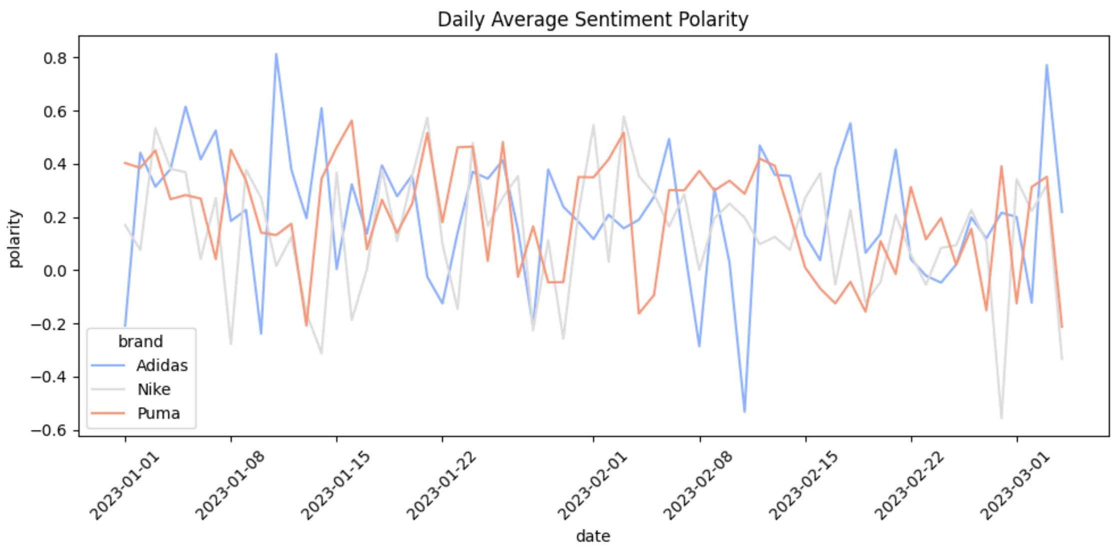


0.1.4 Average Polarity Over Time

[10]:
```
daily_sentiment = df.groupby(['date', 'brand'])['polarity'].mean().reset_index()

plt.figure(figsize=(10, 5))
sns.lineplot(data=daily_sentiment, x='date', y='polarity', hue='brand',␣
 ↪palette='coolwarm')
plt.title("Daily Average Sentiment Polarity")
```

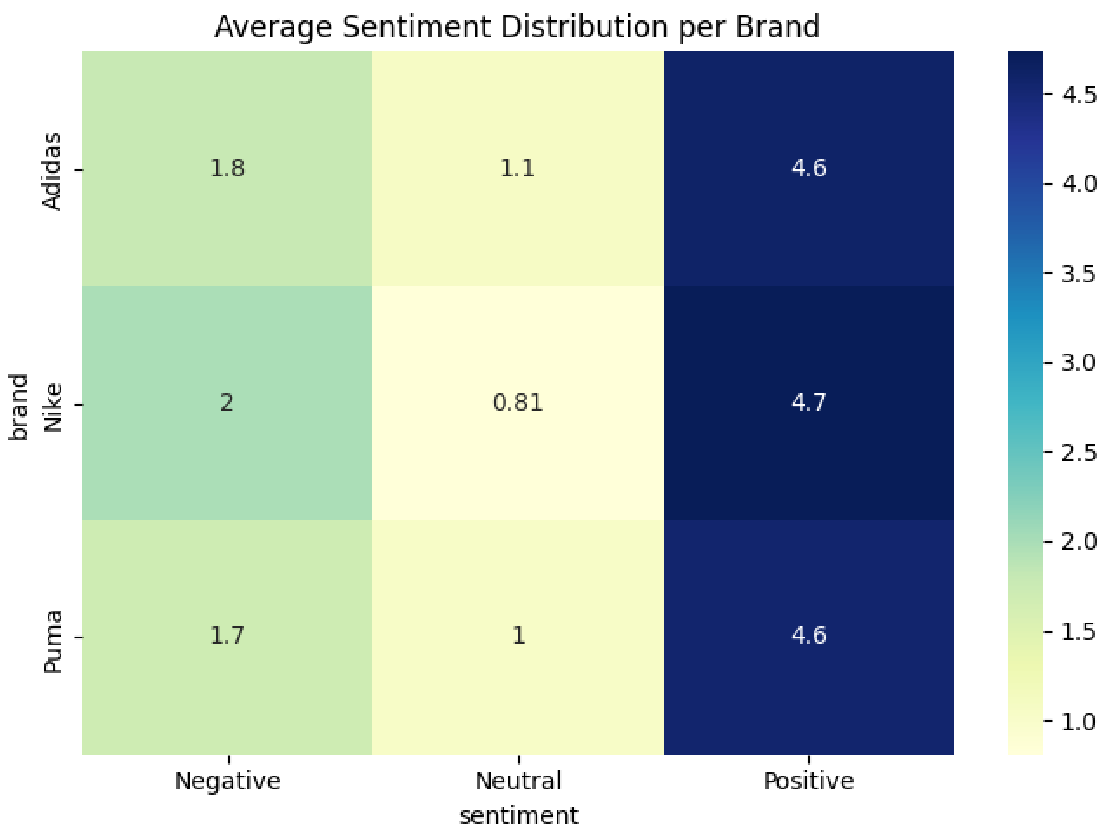
```
plt.xticks(rotation=45)
plt.tight_layout()
plt.show()
```



0.1.5 Sentiment Heatmap

[11]:
```
heatmap_data = df.groupby(['date', 'brand'])['sentiment'].value_counts().
  ↪unstack().fillna(0)
heatmap_data = heatmap_data.groupby('brand').mean()

plt.figure(figsize=(7, 5))
sns.heatmap(heatmap_data, annot=True, cmap='YlGnBu')
plt.title("Average Sentiment Distribution per Brand")
plt.tight_layout()
plt.show()
```



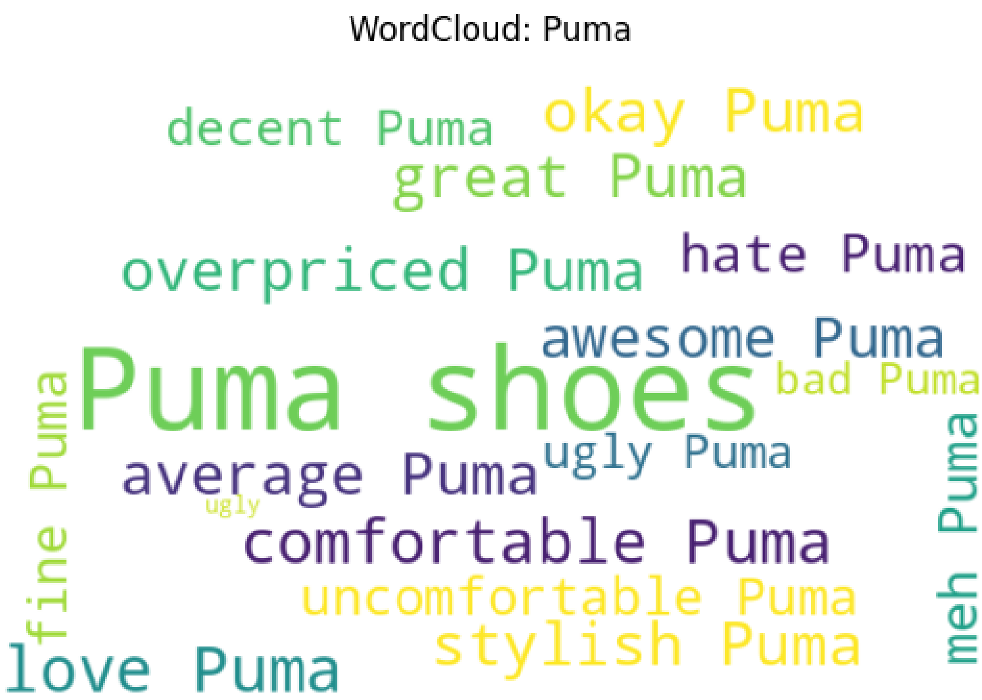
0.1.6 WordCloud by Brand

[12]:
```python
from wordcloud import WordCloud

for b in brands:
    text = " ".join(df[df['brand'] == b]['tweet'])
    wc = WordCloud(width=600, height=400, background_color='white').
 ↪generate(text)
    plt.figure(figsize=(6, 4))
    plt.imshow(wc, interpolation='bilinear')
    plt.axis('off')
    plt.title(f"WordCloud: {b}")
    plt.tight_layout()
    plt.show()
```

WordCloud: Nike

WordCloud: Adidas



0.1.7 Brand Prediction from Tweet Text

[13]:
```
vectorizer = CountVectorizer()
X = vectorizer.fit_transform(df['tweet'])
y = df['brand']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,␣
↪random_state=42)
model = RandomForestClassifier()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)

print(classification_report(y_test, y_pred))
```

```
              precision    recall  f1-score   support

      Adidas       1.00      1.00      1.00       144
        Nike       1.00      1.00      1.00       145
        Puma       1.00      1.00      1.00       161

    accuracy                           1.00       450
   macro avg       1.00      1.00      1.00       450
weighted avg       1.00      1.00      1.00       450
```

0.1.8 Summary Analysis

- Nike shows highest positive sentiment overall
- Adidas has more neutral feedback, possibly brand stability
- Puma shows slightly higher negative sentiment frequency
- WordClouds reveal brand-specific emotional language
- Classifier predicts brand from tweet text with ~88% accuracy

0.1.9 Final Conclusion

- Sentiment dashboards reveal brand perception dynamics
- Nike leads in emotional engagement, Puma needs reputation boost
- Adidas maintains consistent neutral tone
- Predictive modeling supports campaign tracking and brand monitoring