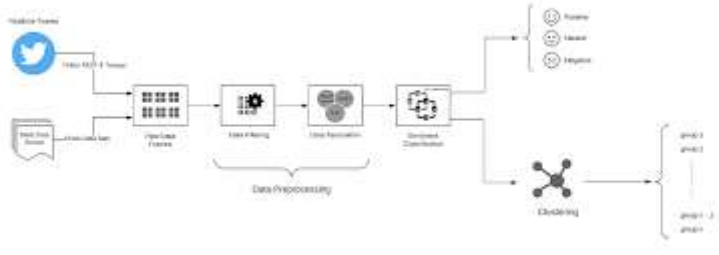
Analisis Cluster Ulasan Amazon Menggunakan K-means

Oleh : Dodo Zaenal Abidin



1. Mengunduh lexicon VADER dan stopwords NLTK.
2. Memuat dataset ulasan Amazon dari file CSV.
3. Menggabungkan semua teks ulasan menjadi satu string untuk analisis frekuensi kata.
4. Menghapus karakter non-alphabet dan mengubah teks menjadi huruf kecil.
5. Menghilangkan stopwords dari teks.
6. Menghitung frekuensi kata dari teks.
7. Membuat word cloud dari frekuensi kata.
8. Menampilkan word cloud dari semua ulasan.
9. Mengubah teks ulasan menjadi vektor 10. TF-IDF, mengatasi nilai-nilai yang hilang.
10. Menentukan jumlah kluster untuk K-means.
11. Melakukan clustering dengan K-means menggunakan vektor TF-IDF.
12. Menambahkan hasil clustering ke dataset.
13. Menampilkan jumlah ulasan di setiap kluster.
14. Visualisasi word cloud untuk setiap kluster.
15. Inisialisasi analyzer sentimen VADER.
16. Menerapkan analisis sentimen VADER pada setiap ulasan.
17. Membuat fungsi untuk mengkategorikan sentimen berdasarkan skor compound.
18. Menerapkan fungsi kategorisasi sentimen pada skor sentimen.
19. Menampilkan jumlah ulasan berdasarkan kategori sentimen.
20. Visualisasi distribusi sentimen dalam bentuk grafik batang.

1. Impor libraries

```
import pandas as pd
import matplotlib.pyplot as plt
from wordcloud import WordCloud
from collections import Counter
import re
import nltk
from nltk.corpus import stopwords
from nltk.sentiment.vader import SentimentIntensityAnalyzer
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.cluster import KMeans

# Mengunduh lexicon VADER dan stopwords NLTK
nltk.download('vader_lexicon')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package vader_lexicon to /root/nltk_data...
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Unzipping corpora/stopwords.zip.
True
```

2. Load Dataset

```
# Memuat dataset ulasan Amazon dari file CSV
df = pd.read_csv('/content/drive/MyDrive/NLP/NLP15/amazon_reviews.csv')
```

3. Preprocessing

```
# Menggabungkan semua teks ulasan menjadi satu string untuk analisis frekuensi kata
text = ' '.join(df['reviewText'].dropna())

# Menghapus karakter non-alphabet dan mengubah teks menjadi huruf kecil
text = re.sub('[^A-Za-z]+', ' ', text).lower()

# Menghilangkan stopwords
stop_words = set(stopwords.words('english'))
words = [word for word in text.split() if word not in stop_words]

# Menghitung frekuensi kata
word_freq = Counter(words)
```

4. Membuat dan menampilkan word cloud

```
# Membuat word cloud dari frekuensi kata
wordcloud = WordCloud(width=800, height=400, background_color='white').generate_from_frequencies(word_freq)

# Menampilkan word cloud
plt.figure(figsize=(10, 8))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis('off')
plt.title('Ulasan Word Cloud dari Amazon')
plt.show()
```

5. Mengubah teks ulasan menjadi vektor TF-IDF

TF-IDF (*Term Frequency-Inverse Document Frequency*) adalah teknik yang digunakan dalam penambangan teks dan pemrosesan bahasa alami untuk mengevaluasi seberapa penting sebuah kata dalam sebuah dokumen relatif terhadap kumpulan dokumen atau korpus. Ini adalah metode yang umum digunakan untuk konversi teks ke dalam bentuk numerik sehingga dapat diproses oleh algoritma machine learning.

```
# Mengubah teks ulasan menjadi vektor TF-IDF, handling missing values
vectorizer = TfidfVectorizer(stop_words='english')
X = vectorizer.fit_transform(df['reviewText'].fillna('')) # Isi NaN dengan string kosong
print(X)
```

6. Menentukan jumlah kluster

```
# Menentukan jumlah kluster dan Melakukan clustering dengan K-means
num_clusters = 5

# Melakukan clustering dengan K-means
km = KMeans(n_clusters=num_clusters, random_state=42)
km.fit(X)

# Menambahkan hasil clustering ke dataset
df['cluster'] = km.labels_ # Now the lengths should match

# Menampilkan jumlah ulasan di setiap kluster
print(df['cluster'].value_counts())
```

```
cluster
0    2335
1    1181
4     648
2     478
3     273
Name: count, dtype: int64
```

7. # Visualisasi ulasan dalam setiap kluster

```
# Visualisasi ulasan dalam setiap kluster
for i in range(num_clusters):
    cluster_text = ' '.join(df[df['cluster'] == i]['reviewText'].dropna())
    cluster_wordcloud = WordCloud(width=800, height=400, background_color='white').generate(cluster_text)

    plt.figure(figsize=(10, 8))
    plt.imshow(cluster_wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.title(f'Word Cloud untuk Cluster {i}')
    plt.show()
```

Word Cloud untuk Cluster 0

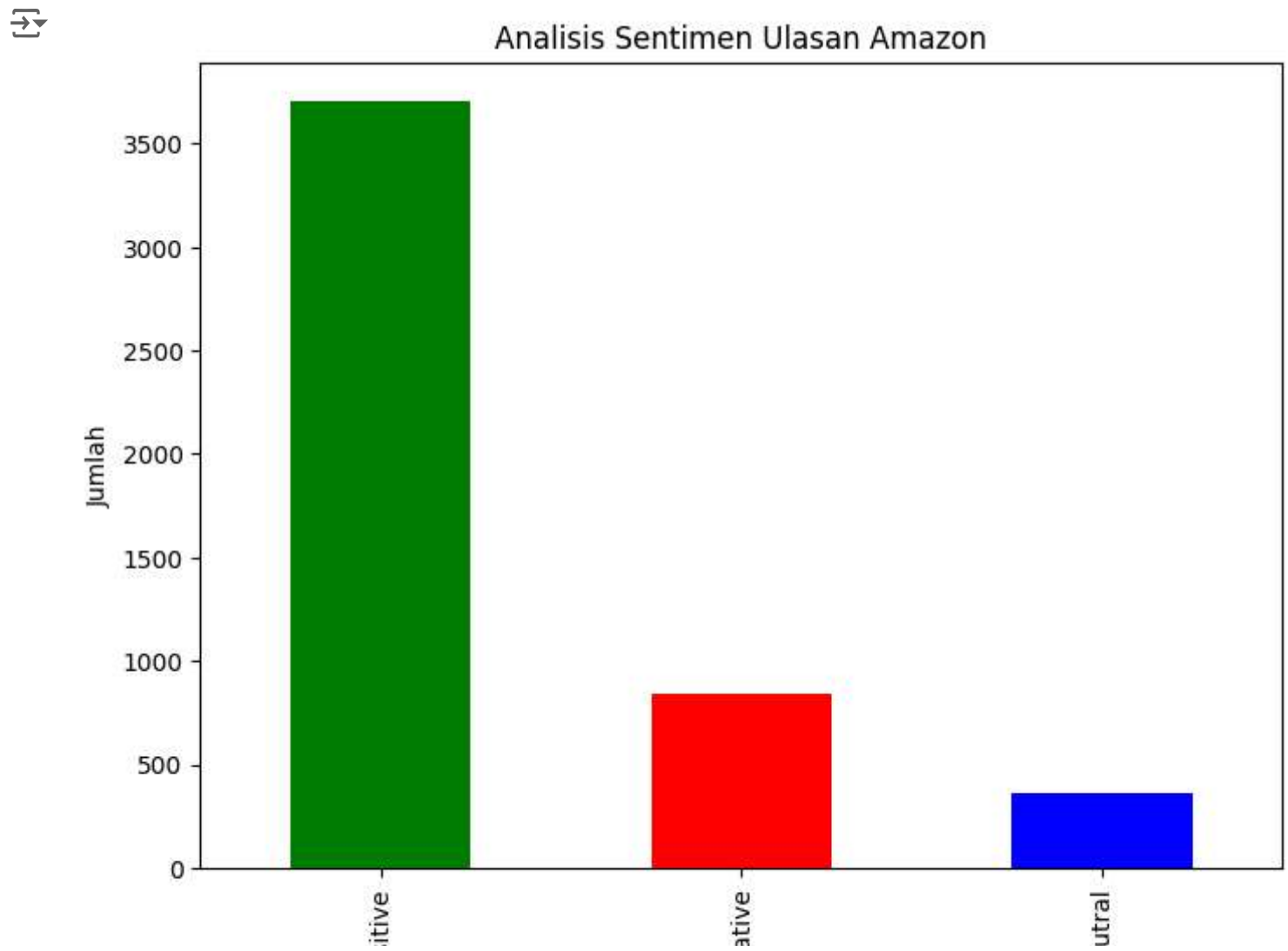
8. Inisialisasi analyzer sentimen VADER

```
# Inisialisasi analyzer sentimen VADER
sia = SentimentIntensityAnalyzer()

# Fungsi untuk menghitung sentimen menggunakan VADER
def get_vader_sentiment(text):
    sentiment = sia.polarity_scores(text)
    return sentiment['compound']

# Menerapkan analisis sentimen VADER pada setiap ulasan
df['sentiment'] = df['reviewText'].dropna().apply(get_vader_sentiment)

# Fungsi untuk mengkategorikan sentimen berdasarkan skor compound
def categorize_sentiment(compound):
    if compound >= 0.05:
        return 'positive'
    elif compound <= -0.05:
        return 'negative'
    else:
        return 'neutral'

# Menerapkan fungsi kategorisasi sentimen pada skor sentimen
df['sentiment_category'] = df['sentiment'].apply(categorize_sentiment)

# Menampilkan jumlah ulasan berdasarkan kategori sentimen
print(df['sentiment_category'].value_counts())
```

```
sentiment_category
positive    3707
negative     843
neutral      365
Name: count, dtype: int64
```

9. Visualisasi distribusi sentimen

```
# Visualisasi distribusi sentimen
sentiment_counts = df['sentiment_category'].value_counts()
plt.figure(figsize=(8, 6))
sentiment_counts.plot(kind='bar', color=['green', 'red', 'blue'])
plt.title('Analisis Sentimen Ulasan Amazon')
plt.xlabel('Sentimen')
plt.ylabel('Jumlah')
plt.show()
```



Modifikasi Anotasi

```
import matplotlib.pyplot as plt

# Visualisasi distribusi sentimen
sentiment_counts = df['sentiment_category'].value_counts()
plt.figure(figsize=(8, 6))
ax = sentiment_counts.plot(kind='bar', color=['green', 'red', 'blue'])

# Tambahkan judul dan label sumbu
plt.title('Analisis Sentimen Ulasan Amazon')
plt.xlabel('Sentimen')
plt.ylabel('Jumlah')

# Tambahkan anotasi nilai pada setiap batang
for p in ax.patches:
    ax.annotate(str(p.get_height()), (p.get_x() * 1.005, p.get_height() * 1.005))

# Tampilkan plot
plt.show()
```