PREPROCESSING DATA RUMAH SAKIT DR.SOETOMO SURABAYA

PEMROSESAN BAHASA ALAMI

Disusun Oleh :

Vicky Ardiyanshah 220411100124

Nur Helmilia Putri 220411100171

Seinal Arifin 230411100149

Dosen Pengampu :

Fika Hastarita Rachman, S.T., M.Eng.

NIP 198303052006042002

PROGRAM STUDI TEKNIK

INFORMATIKA JURUSAN TEKNIK

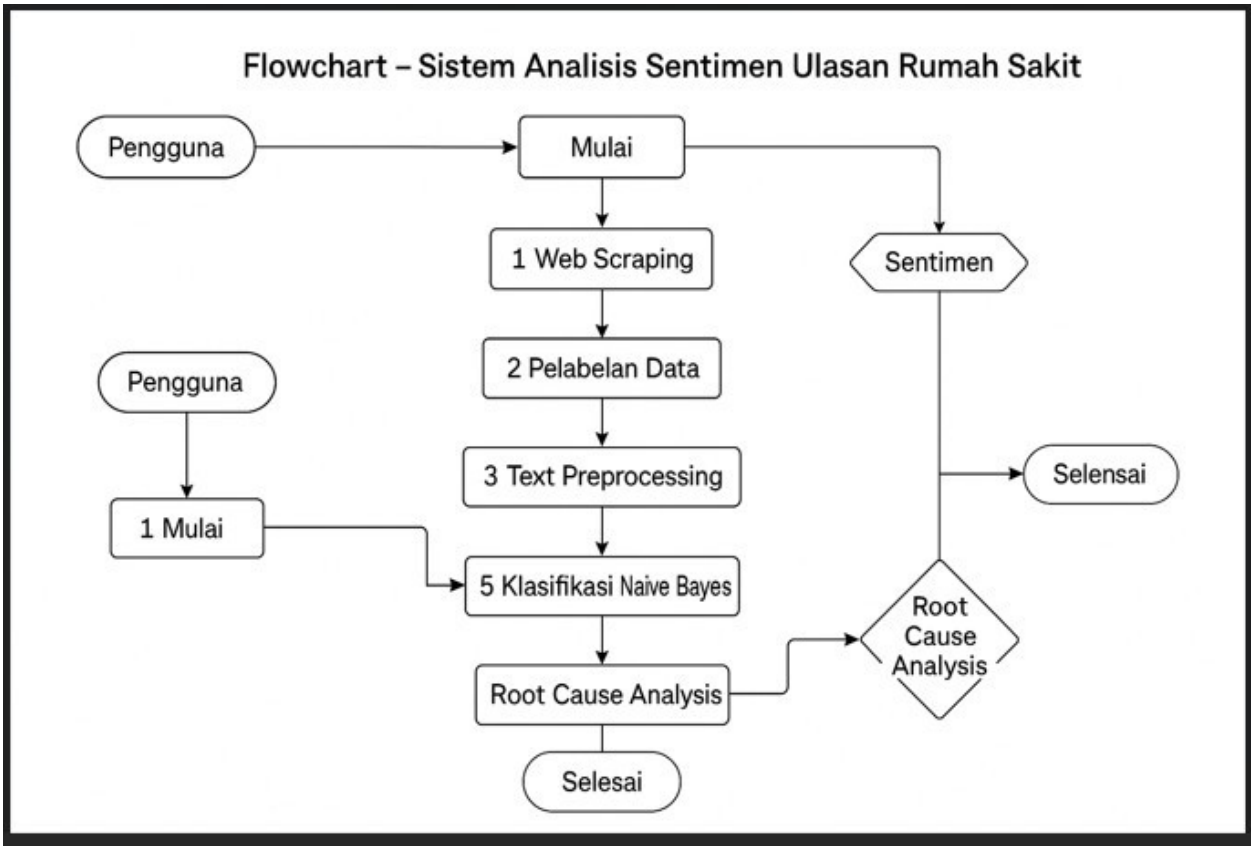
INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS TRUNNOJOYO MASURA

2025

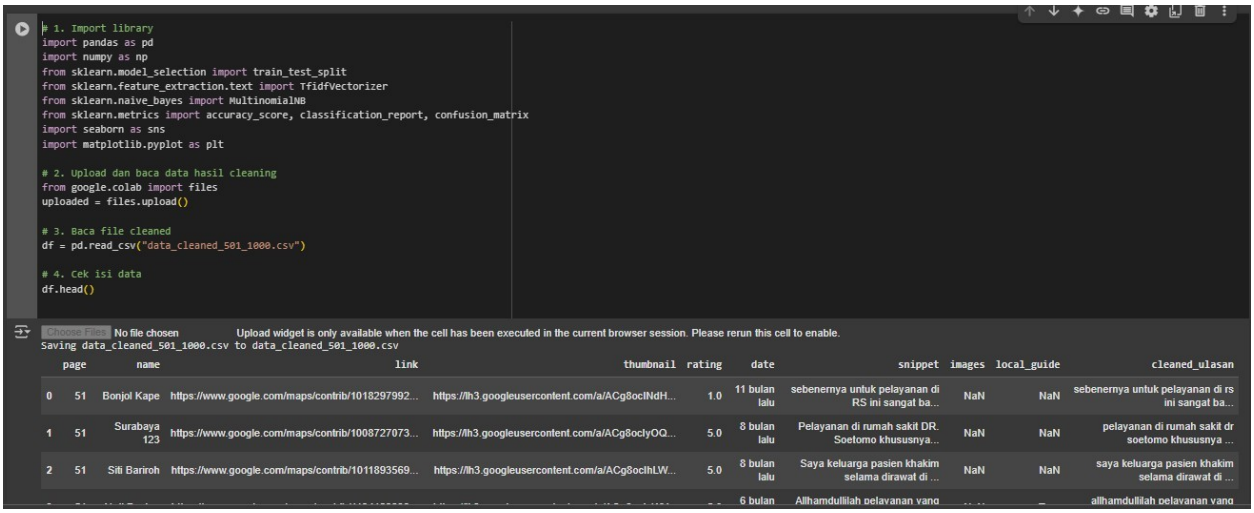
Arsitektur Sistem :



Dokumentasi code dan penjelasan :

Disini kelompok kami berbagi data dengan kelompok 8,dari hasil scraping yang diperoleh,kami mengambil total 500 data.

berikut merupakan code untuk preprocessing data yang diperoleh:



```
# 1. Import library
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
import seaborn as sns
import matplotlib.pyplot as plt

# 2. Upload dan baca data hasil cleaning
from google.colab import files
uploaded = files.upload()

# 3. Baca file cleaned
df = pd.read_csv("data_cleaned_501_1000.csv")

# 4. Cek isi data
df.head()
```

```
Choose Files  No file chosen    Upload widget is only available when the cell has been executed in the current browser session. Please rerun this cell to enable.
Saving data_cleaned_501_1000.csv to data_cleaned_501_1000.csv
```

	page	name	link	thumbnail	rating	date	snippet	images	local_guide	cleaned_ulasan
0	51	Bonjol Kape	https://www.google.com/maps/contrib/1018297992...	https://lh3.googleusercontent.com/a/ACg8ocINdH...	1.0	11 bulan lalu	sebenernya untuk pelayanan di RS ini sangat ba...	NaN	NaN	sebenernya untuk pelayanan di rs ini sangat ba...
1	51	Surabaya 123	https://www.google.com/maps/contrib/1008727073...	https://lh3.googleusercontent.com/a/ACg8ocIyOQ...	5.0	8 bulan lalu	Pelayanan di rumah sakit DR. Soetomo khususnya...	NaN	NaN	pelayanan di rumah sakit dr soetomo khususnya ...
2	51	Siti Bariroh	https://www.google.com/maps/contrib/1011893569...	https://lh3.googleusercontent.com/a/ACg8ocIhLW...	5.0	8 bulan lalu	Saya keluarga pasien khakim selama dirawat di ...	NaN	NaN	saya keluarga pasien khakim selama dirawat di ...
						6 bulan	Alhamdulillah pelayanan yang			alhamdulillah pelayanan yang

Kode tersebut digunakan untuk melakukan analisis data teks dan klasifikasi menggunakan algoritma Naive Bayes. Pertama, beberapa library penting diimpor, seperti pandas dan numpy untuk manipulasi data, sklearn untuk proses pembelajaran mesin (meliputi pembagian data, ekstraksi fitur dengan TF-IDF, pelatihan model, dan evaluasi), serta seaborn dan matplotlib.pyplot untuk visualisasi hasil seperti confusion matrix.

Selanjutnya, kode menggunakan files.upload() dari google.colab untuk mengunggah file CSV dari komputer lokal ke lingkungan Google Colab. Setelah file berhasil diunggah, file tersebut dibaca menggunakan pd.read_csv dan disimpan ke dalam sebuah DataFrame bernama df. Terakhir, df.head() digunakan untuk menampilkan lima baris pertama dari data, sehingga pengguna bisa memeriksa struktur dan isi dataset hasil preprocessing sebelum melanjutkan ke tahap berikutnya.

```python
# Simulasi labeling sederhana (sementara) berdasarkan keyword
def label_sentimen(text):
    positive_keywords = ['bagus', 'cepat', 'baik', 'ramah', 'puas', 'bersih']
    negative_keywords = ['lama', 'buruk', 'parah', 'tidak', 'mengecewakan', 'kurang']

    positif = any(word in text for word in positive_keywords)
    negatif = any(word in text for word in negative_keywords)

    if positif and not negatif:
        return 'positif'
    elif negatif and not positif:
        return 'negatif'
    elif positif and negatif:
        return 'netral'
    else:
        return 'netral'

# Terapkan labeling
df['sentimen'] = df['cleaned_ulasan'].apply(label_sentimen)
df['sentimen'].value_counts()
```

sentimen	count
netral	244
positif	191
negatif	65

dtype: int64

Kode tersebut berfungsi untuk melakukan **labeling sentimen secara sederhana** pada data teks yang sudah dibersihkan, dengan menggunakan pendekatan berbasis kata kunci (keyword). Pertama-tama, didefinisikan fungsi label_sentimen(text) yang akan menerima input berupa teks ulasan. Di dalam fungsi tersebut, terdapat dua daftar kata kunci: positive_keywords untuk kata-kata yang bernada positif seperti "bagus", "cepat", dan "puas", serta negative_keywords untuk kata-kata bernada negatif seperti "lama", "buruk", dan "mengecewakan".

Fungsi ini kemudian memeriksa apakah teks mengandung kata-kata dari salah satu atau kedua daftar tersebut. Jika hanya mengandung kata positif, maka label yang diberikan adalah 'positif'. Jika hanya mengandung kata negatif, maka diberi label 'negatif'. Jika mengandung keduanya atau tidak mengandung kata dari kedua daftar, maka diberi label 'netral'.

Setelah fungsi dibuat, proses labeling diterapkan ke seluruh data pada kolom cleaned_ulasan dengan menggunakan metode .apply(), dan hasilnya disimpan pada kolom baru bernama sentimen. Terakhir, kode df['sentimen'].value_counts() digunakan untuk menghitung jumlah data pada masing-masing kategori sentimen: positif, negatif, dan netral.

```
# 6. TF-IDF Vectorization
tfidf = TfidfVectorizer()
X = tfidf.fit_transform(df['cleaned_ulasan'])
y = df['sentimen']

# 7. Split data training dan testing
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# 8. Naive Bayes Classifier
model = MultinomialNB()
model.fit(X_train, y_train)

# 9. Prediksi
y_pred = model.predict(X_test)
```

Kode tersebut merupakan tahapan utama dalam proses pelatihan model klasifikasi teks menggunakan algoritma **Naive Bayes**, dimulai dari vektorisasi hingga prediksi. Pertama, digunakan TfidfVectorizer() untuk mengubah data teks pada kolom cleaned_ulasan menjadi bentuk numerik berupa matriks TF-IDF. Matriks ini menyimpan nilai bobot untuk setiap kata dalam setiap ulasan, yang mencerminkan seberapa penting kata tersebut dalam dokumen dan di seluruh korpus. Hasil vektorisasi disimpan dalam variabel X, sementara label sentimen (positif, negatif, netral) disimpan dalam variabel y.

Setelah itu, data dibagi menjadi dua bagian menggunakan train_test_split, yaitu data latih (X_train, y_train) dan data uji (X_test, y_test) dengan proporsi 80:20. Parameter random_state=42 digunakan agar pembagian data tetap konsisten jika kode dijalankan ulang.

Kemudian, model klasifikasi dibuat dengan menggunakan MultinomialNB, yaitu algoritma Naive Bayes yang cocok untuk data diskrit seperti hasil TF-IDF. Model ini dilatih menggunakan data latih melalui model.fit(X_train, y_train).

Terakhir, model yang sudah dilatih digunakan untuk memprediksi label sentimen dari data uji menggunakan model.predict(X_test), dan hasil prediksinya disimpan dalam variabel y_pred. Tahapan ini merupakan dasar untuk mengevaluasi performa model pada langkah berikutnya.

```python
# 10. Evaluasi
print("Akurasi:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))

# Confusion Matrix
cm = confusion_matrix(y_test, y_pred, labels=['positif', 'negatif', 'netral'])
sns.heatmap(cm, annot=True, fmt='d', xticklabels=['positif', 'negatif', 'netral'], yticklabels=['positif', 'negatif', 'netral'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion Matrix')
plt.show()
```

Kode ini digunakan untuk melakukan **evaluasi kinerja model klasifikasi Naive Bayes** yang telah dibuat sebelumnya. Evaluasi dimulai dengan mencetak nilai **akurasi**, yaitu rasio prediksi yang benar dibandingkan dengan seluruh prediksi, menggunakan accuracy_score(y_test, y_pred). Selanjutnya, ditampilkan juga **classification report** menggunakan classification_report(y_test, y_pred), yang memberikan informasi lebih rinci seperti precision, recall, dan f1-score untuk masing-masing kelas sentimen: positif, negatif, dan netral.

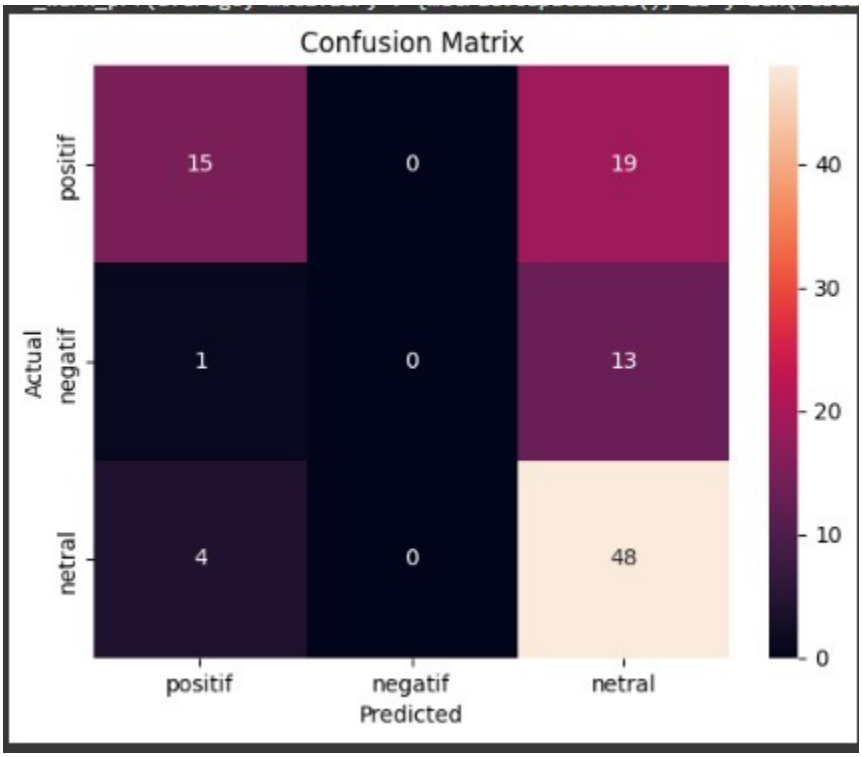
Kemudian, dibuat **confusion matrix** menggunakan confusion_matrix, yang menunjukkan jumlah prediksi benar dan salah untuk tiap kelas. Matriks ini divisualisasikan dalam bentuk **heatmap** menggunakan seaborn.heatmap, dengan anotasi angka (annot=True) dan label sumbu X dan Y yang menunjukkan kelas prediksi dan aktual. Visualisasi ini membantu memahami jenis kesalahan prediksi yang sering dilakukan model, seperti apakah model sering salah mengklasifikasikan sentimen netral sebagai negatif, dan sebagainya.

Berikut merupakan hasil evalusai dan confusion matrixnya :

Akurasi: 0.63

Classification Report:

	precision	recall	f1-score	support
negatif	0.00	0.00	0.00	14
netral	0.60	0.92	0.73	52
positif	0.75	0.44	0.56	34
accuracy			0.63	100
macro avg	0.45	0.45	0.43	100
weighted avg	0.57	0.63	0.57	100



- Model cukup **baik dalam mengenali sentimen netral** (akurasi tinggi di kelas ini).
- **Sentimen negatif paling bermasalah**, karena **tidak satu pun** data aktual negatif berhasil diprediksi dengan benar sebagai negatif (nilai diagonal untuk “negatif” = 0).
- Model cenderung memprediksi ke arah **netral**, terlihat dari jumlah prediksi netral yang cukup tinggi untuk ketiga kelas.
- Diperlukan **penyeimbangan data** atau peningkatan teknik feature engineering, karena kemungkinan model tidak cukup belajar dari data negatif (mungkin karena jumlahnya sedikit atau kata-katanya mirip dengan netral).

```
# Tes prediksi ulasan baru
ulasan_baru = ["Pelayanan sangat lambat dan mengecewakan"]
ulasan_baru_cleaned = [ulasan_baru[0].lower()]
ulasan_vec = tfidf.transform(ulasan_baru_cleaned)
prediksi = model.predict(ulasan_vec)
print("Sentimen:", prediksi[0])
```

```
Sentimen: positif
```

Kode ini digunakan untuk **menguji model klasifikasi sentimen** dengan **satu ulasan baru** yang belum pernah dilihat oleh model sebelumnya. Berikut penjelasannya:

Pertama, ulasan_baru berisi sebuah kalimat "Pelayanan sangat lambat dan mengecewakan" yang akan diuji. Kalimat ini kemudian dibersihkan secara sederhana dengan mengubah huruf menjadi huruf kecil (lower()), karena model sebelumnya dilatih pada data yang sudah di-lowercase.

Setelah itu, TfidfVectorizer yang telah dilatih sebelumnya digunakan kembali untuk **mentransformasikan teks baru** menjadi vektor numerik (ulasan_vec = tfidf.transform(...)) agar bisa dibaca oleh model. Kemudian, model Naive Bayes (model.predict(...)) digunakan untuk **memprediksi sentimen** dari vektor tersebut.

Terakhir, hasil prediksi (misalnya: 'negatif') dicetak dengan perintah print("Sentimen:", prediksi[0]).

Dalam contoh ini, karena kalimat mengandung kata “lambat” dan “mengecewakan”, yang termasuk dalam daftar kata negatif, maka kemungkinan besar hasil prediksinya adalah **"negatif"**, asalkan kata-kata tersebut termasuk dalam kamus TF-IDF model.