

DAFTAR PUSTAKA

- Asrumi, Suharijadi, D., Setiari, A. D., & Wulanda, D. P. (2023). *Analisis Sentimen dan Penggalan Opini*. Eureka Media Aksara.
- Hardiany, R., & Saputra, A. (2022). Penerapan Algoritma Naive Bayes untuk Klasifikasi Sentimen Masyarakat terhadap Kualitas Layanan Pemerintah. *Jurnal Informatika dan Teknologi Sistem Informasi*.
- Hidayat, T. (2025). *Pedoman Teknis Pengolahan Data Kuantitatif untuk Skripsi Teknik Informatika*. Andi Offset.
- Hidayah, A. K., Sunardi, D., & Sahputra, E. (2025). *Analisis Sentimen di Era Digital: Konsep, Algoritma, dan Studi Kasus Media Sosial*. PT Literasi Nusantara Abadi Grup.
- IBM Education. (2023). *What is Artificial Intelligence (AI)?* Tersedia di: <https://www.ibm.com/topics/artificial-intelligence>.
- Indrajit, R. E. (2020). *Manajemen Sistem Informasi dan Teknologi Informasi*. Prehallindo.
- Jurafsky, D., & Martin, J. H. (2020). *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition*. Stanford University.
- Liu, B. (2020). *Sentiment Analysis: Mining Opinions, Sentiments, and Emotions*. Cambridge University Press.
- Pratama, R. (2021). *Evaluasi Pelayanan Publik di Era Digital: Tantangan dan Peluang*. Universitas Muhammadiyah Bengkulu Press.
- Sari, D. P., & Kurniawan, D. (2024). Analisis Kepuasan Masyarakat Menggunakan Data Media Sosial dengan Pendekatan Machine Learning. *Jurnal Sistem Informasi dan Bisnis*.
- Wicaksono, A. (2023). *Natural Language Processing untuk Bahasa Indonesia: Teori dan Implementasi Preprocessing*. Informatika.
- Zhang, H., Li, J., & Liu, Y. (2020). Exploring the Performance of Naive Bayes in Text Classification. *Journal of Computer Science and Technology*.

L

A

M

P

I

R

A

N



Dokumentasi Lokasi Penelitian di DPMPTSP Provinsi Bengkulu



Foto saat pengambilan data Pertama di kantor DPMPTSP Provinsi Bengkulu



Ruangan pengambilan data



Dokumentasi Pelaksanaan Pengambilan Data di DPMPTSP Provinsi Bengkulu



```
proses data set

[103] dataset = pd.read_csv(
✓ 0s /content/drive/MyDrive/400 data dpmtsp/data_menta_ dpmtsp.csv',
    sep=';',
    engine='python'
)

[104] #####
✓ 0s # PENILIHAN DATA PENELITIAN
    #####

    data = dataset[['Komentar / Saran']].copy()

    data.columns = ['komentar']

    print(data.shape)

    data.head()

> ... Show hidden output

Next steps: Generate code with data New interactive sheet
```

```
[105] # Menampilkan 10 data pertama dataset
✓ 0s tabel_dataset = data[['komentar']].copy()

    tabel_dataset.head(10)

> ... Show hidden output

Next steps: Generate code with tabel_dataset New interactive sheet

+ Code + Text

[106] data.to_csv(
✓ 0s "/content/drive/MyDrive/400 data dpmtsp/data_penelitian.csv",
    index=False,
    encoding="utf-8-sig"
)

    print("Data penelitian berhasil disimpan.")

> Show hidden output
```

Proses Tahapan Dataset

Cf

```
cleaning dan case folding

[107] ✓ Os
# CLEANING DAN CASE FOLDING
# =====

import re

def cleaning_casefolding(text):

    text = str(text)

    # Case Folding
    text = text.lower()

    # Menghapus URL
    text = re.sub(r'http\S+|www\S+', ' ', text)

    # Menghapus mention dan hashtag
    text = re.sub(r'@\w+|\#\w+', ' ', text)

    # Menghapus angka
    text = re.sub(r'\d+', ' ', text)

    # Menghapus tanda baca
    text = re.sub(r'[^\w\s]', ' ', text)
```

```
[107] ✓ Os
# Menghapus spasi berlebih
text = re.sub(r'\s+', ' ', text).strip()

return text

data["cleaning_casefolding"] = data["komentar"].apply(cleaning_casefolding)

data.head(10)

... Show hidden output

Next steps: Generate code with data New interactive sheet

[108] ✓ Os
# SAVE CLEANING DAN CASE FOLDING
# =====

data.to_csv(
    "/content/drive/MyDrive/400 data dpmtsp/hasil_cleaning_casefolding.csv",
    index=False,
    encoding="utf-8-sig"
)

print("Hasil Cleaning dan Case Folding berhasil disimpan.")

... Show hidden output
```

```
[109] ✓ Os
# CEK NILAI KOSONG
# =====

print(data.isnull().sum())

... Show hidden output
```

Proses Tahapan Cleaning dan Case Folding

```

[110]
✓ Os
NORMALISASI

#=====
# MEMBUAT KAMUS NORMALISASI
#=====

import pandas as pd

kamus = {
    "slang": [
        "gk","ga","gak","nggak","ngga","tdk","tak",
        "blm","udh","sdh","aja","yg","dg","dgn","dn",
        "utk","krn","karna","tp","tpi","kalo","klau",
        "klu","bgt","banget","bnyk","byk","jd","jg",
        "smoga","moga","mksh","makasih","trimakasih",
        "terimakasih","thx","ok","oke","mantab",
        "mantabb","bgs","bgus","cepet","cpt","sip",
        "jos","top","responif","pelayananya",
        "ditingkatkan","dipertahankan","ramah",
        "memuaskan","puass"
    ],
    "formal": [
        "tidak","tidak","tidak","tidak","tidak","tidak","tidak",
        "belum","sudah","sudah","saja","yang","dengan","dengan","dari",
        "untuk","karena","karena","tapi","tapi","kalau","kalau",
        "kalau","sekali","sekali","banyak","banyak","jadi","juga",
    ]
}

```

```

[110]
✓ Os
    "semoga","semoga","terima kasih","terima kasih","terima kasih",
    "terima kasih","terima kasih","baik","baik","mantap",
    "mantap","bagus","bagus","cepat","cepat","baik",
    "baik","baik","responsif","pelayananya",
    "ditingkatkan","dipertahankan","ramah",
    "memuaskan","puas"
    ]
}

kamus = pd.DataFrame(kamus)

kamus.head()

... Show hidden output

Next steps: Generate code with kamus New interactive sheet

[111]
✓ Os
#=====
# SIMPAN KAMUS NORMALISASI
#=====

kamus.to_csv(
    "/content/drive/MyDrive/400 data dpmtsp/kamus_normalisasi.csv",
    index=False,
    encoding="utf-8-sig"
)

```

```

[113]
✓ Os
    for kata in text.split():
        hasil.append(kamus_dict.get(kata, kata))

    return " ".join(hasil)

data["normalisasi"] = data["cleaning_casefolding"].apply(normalisasi)

data.head(10)

... Show hidden output

Next steps: Generate code with data New interactive sheet

[116]
✓ Os
#=====
# SIMPAN HASIL NORMALISASI
#=====

data.to_csv(
    "/content/drive/MyDrive/400 data dpmtsp/hasil_normalisasi.csv",
    index=False,
    encoding="utf-8-sig"
)

print("Hasil normalisasi berhasil disimpan.")

... Show hidden output

```

Proses Tahapan Normalisasi

```
[116] import nltk
      nltk.download('punkt')
      nltk.download('punkt_tab')

... [nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data] Package punkt is already up-to-date!
[nltk_data] Downloading package punkt_tab to /root/nltk_data...
[nltk_data] Package punkt_tab is already up-to-date!
True

[117] #####
# TOKENIZING
#####

from nltk.tokenize import word_tokenize

data["tokenizing"] = data["normalisasi"].apply(word_tokenize)

data.head(10)
```

Next steps: [Generate code with data](#) [New interactive sheet](#)

```
[118] #####
# SIMPAN HASIL TOKENIZING
#####

data.to_csv(
    "/content/drive/MyDrive/480 data dpmtsp/hasil_tokenizing.csv",
    index=False,
    encoding="utf-8-sig"
)

print("Hasil tokenizing berhasil disimpan.")

Hasil tokenizing berhasil disimpan.

[119] data[["normalisasi", "tokenizing"]].head(10)
```

Proses Tahapan Tokenizing

```
stopword removal

[128] ✓ 0s #=====
# MEMBUAT STOPWORD KHUSUS ANALISIS SENTIMEN
#=====

from nltk.corpus import stopwords
import nltk

nltk.download('stopwords')

stop_words = set(stopwords.words('indonesian'))

# Kata yang harus dipertahankan
keep_words = {
    "tidak",
    "belum",
    "bukan",

    "baik",
    "bagus",
    "buruk",
    "jelek",

    "puas",
    "memuaskan",
```

```
"kecewa",
"cepat",
"lambat",

"mudah",
"sulit",

"ramah",
"responsif",

"semangat",
"mantap",

"seanget",
"kurang"
}

# Hapus dari daftar stopwords
stop_words = stop_words - keep_words

print("Jumlah stopwords :", len(stop_words))

> ... Show hidden output
```

```
[129] ✓ 0s #=====
# STOPWORD REMOVAL
#=====

def remove_stopwords(tokens):
    return [kata for kata in tokens if kata not in stop_words]

data["stopword"] = data["tokenizing"].apply(remove_stopwords)

data[["tokenizing", "stopword"]].head(10)

> Show hidden output

[129] ✓ 0s from nltk.corpus import stopwords
import nltk

nltk.download('stopwords')

stop_words = set(stopwords.words('indonesian'))

# Kata yang dipertahankan
keep_words = {
    "tidak",
    "belum",
    "bukan",
    "baik",
    "bagus",
```

Proses Tahapan Stopword Removal

```
Commands + Code + Text ▶ Run all RAM Disk
[122] ✓ Os
    "buruk",
    "jelek",
    "puas",
    "memuaskan",
    "kecewa",
    "cepat",
    "lambat",
    "mudah",
    "sulit",
    "ramah",
    "responsif",
    "semangat",
    "mantap",
    "sangat",
    "kurang"
    }

    stop_words = stop_words - keep_words

    def remove_stopwords(tokens):
        return [kata for kata in tokens if kata not in stop_words]

    data["stopword"] = data["tokenizing"].apply(remove_stopwords)
```

```
Commands + Code + Text ▶ Run all RAM Disk
[122] ✓ Os
    data[
        "komentar",
        "cleaning_casfolding",
        "normalisasi",
        "tokenizing",
        "stopword"
    ].head(10)
> ... Show hidden output
[134] ✓ Os
    #=====
    # SIMPAN HASIL STOPWORD
    #=====

    data.to_csv(
        "/content/drive/MyDrive/400 data dpmtsp/hasil_stopword.csv",
        index=False,
        encoding="utf-8-sig"
    )

    print("Hasil stopwords berhasil disimpan")
```

Lanjutan Proses Stopword Removal

```
Commands + Code + Text ▶ Run all RAM Disk
tahap Stemming

[125] ✓ 0s | pip install -q Sastrawi

[126] ✓ 31s # =====
# STEMMING
# =====

from Sastrawi.Stemmer.StemmerFactory import StemmerFactory

factory = StemmerFactory()
stemmer = factory.create_stemmer()

def stemming(tokens):
    hasil = []

    for kata in tokens:
        hasil.append(stemmer.stem(kata))

    return " ".join(hasil)

data["stemming"] = data["stopword"].apply(stemming)

data[["komentar",
      "stopword",
      "stemming",
```

```
Commands + Code + Text ▶ Run all RAM Disk

[126] ✓ 31s "stemming"
]]).head(10)
> ... Show hidden output

[127] ✓ 0s # =====
# MENAMPILKAN HASIL STEMMING
# =====

data[["komentar",
      "cleaning_casefolding",
      "normalisasi",
      "tokenizing",
      "stopword",
      "stemming"
      ]].head(10)
> ... Show hidden output
```

```
Commands + Code + Text ▶ Run all RAM Disk

[128] ✓ 0s # =====
# SIMPAN HASIL STEMMING
# =====

data.to_csv(
    "/content/drive/MyDrive/400 data dpmptsp/hasil_stemming.csv",
    index=False,
    encoding="utf-8-sig"
)

print("Hasil stemming berhasil disimpan di Google Drive.")
```

Proses Tahapan Stemming

```
proses pelabelan sentimen

[129] # =====
# MEMBACA LEXICON POSITIF DAN NEGATIF
# =====

import pandas as pd

positive = pd.read_csv(
    "/content/drive/MyDrive/400 data dpmtsp/positive.tsv",
    sep="\t"
)

negative = pd.read_csv(
    "/content/drive/MyDrive/400 data dpmtsp/negative.tsv",
    sep="\t"
)

positive_dict = dict(zip(positive["word"], positive["weight"]))
negative_dict = dict(zip(negative["word"], negative["weight"]))

print("Lexicon Positif :", len(positive_dict))
print("Lexicon Negatif :", len(negative_dict))

> ... Show hidden output
```

```
[130] # =====
# KATA KUNCI NETRAL
# =====

komentar_netral = [
    "",
    "_",
    "-",
    "tidak",
    "tidak ada",
    "belum",
    "belum ada",
    "ga ada",
    "gak ada",
    "tanpa saran",
    "tidak saran",
    "tidak komentar",
    "camkoha"
]

[131] # =====
# PELABELAN SENTIMEN 2 KELAS
# =====

def pelabelan_dua_kelas(teks):
    teks = str(teks).lower().strip()
```

```
[131] # komentar netral
if teks in komentar_netral:
    return "Netral"

# komentar berupa saran
if (
    "saran" in teks or
    "ditingkatkan" in teks or
    "ditambah" in teks or
    "mohon" in teks
):
    if (
        "ganggu" not in teks and
        "hambat" not in teks and
        "lambat" not in teks and
        "kecewa" not in teks and
        "buruk" not in teks
    ):
        return "Netral"

# Hitung skor positif
skor = 0

for kata in teks.split():
```

Proses Tahapan Pelabelan

```
Commands + Code + Text ▶ Run all RAM Disk
[131] ✓ Os
    if kata in positive_dict:
        skor += positive_dict[kata]

    if skor > 0:
        return "Positif"

    return "Netral"

[132] ✓ Os
    # =====
    # MENJALANKAN PELABELAN
    # =====

    data["label_sentimen"] = data["stemming"].apply(
        pelabelan_dua_kelas
    )

[133] ✓ Os
    # =====
    # HASIL PELABELAN
    # =====

    data[
        [
            "komentar",
            "cleaning_casefolding",
            "normalisasi",
            "tokenizing",
```

```
Commands + Code + Text ▶ Run all RAM Disk
[133] ✓ Os
    "stopword",
    "stemming",
    "label_sentimen"
    ].head(10)
    > Show hidden output

[134] ✓ Os
    # =====
    # JUMLAH LABEL
    # =====

    print(data["label_sentimen"].value_counts())
    > ... Show hidden output

[135] ✓ Os
    for teks in data[data["label_sentimen"]=="Netral"]["stemming"]:
        skor = 0
        for kata in str(teks).split():
            if kata in positive_dict:
                skor += positive_dict[kata]
            if skor > 0:
                print(teks, skor)
    > Show hidden output
```

Lanjutan Proses Tahapan Pelabelan

The screenshot shows a Jupyter Notebook with the title "tahap TF-IDF". The code is as follows:

```
[116]: !pip install scikit-learn
> Show hidden output

[117]: from sklearn.feature_extraction.text import TfidfVectorizer
import pandas as pd

[118]: tfidf = TfidfVectorizer()

X = tfidf.fit_transform(data["stemming"])

y = data["label_sentimen"]

[119]: print("Jumlah dokumen :", X.shape[0])
print("Jumlah kata :", X.shape[1])
> ... Show hidden output

[120]: print(tfidf.get_feature_names_out()[:30])
> Show hidden output
```

The screenshot shows the continuation of the Jupyter Notebook. The code is as follows:

```
[141]: tfidf_df = pd.DataFrame(
X.toarray(),
columns=tfidf.get_feature_names_out()
)

tfidf_df.head()
> Show hidden output

[142]: tfidf_df.to_csv(
"/content/drive/MyDrive/400 data dpmptsp/hasil_tfidf.csv",
index=False,
encoding="utf-8-sig"
)

print("TF-IDF berhasil disimpan")
> Show hidden output
```

Proses Tahapan TF- IDF

```
[143] ✓ 0s from sklearn.model_selection import train_test_split

[144] ✓ 0s X_train, X_test, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.20,
    random_state=42,
    stratify=y
)

[145] ✓ 0s print("Data Latih :", X_train.shape)
print("Data Uji   :", X_test.shape)

> Show hidden output

[146] ✓ 0s print("\nLabel Data Latih")
print(y_train.value_counts())

print("\nLabel Data Uji")
print(y_test.value_counts())

> Show hidden output
```

```
[147] ✓ 1s import pandas as pd

train_df = pd.DataFrame(X_train.toarray(), columns=tfidf.get_feature_names_out())
train_df["label"] = y_train.reset_index(drop=True)

test_df = pd.DataFrame(X_test.toarray(), columns=tfidf.get_feature_names_out())
test_df["label"] = y_test.reset_index(drop=True)

train_df.to_csv(
    "/content/drive/MyDrive/400 data dpemtpsp/data_train.csv",
    index=False,
    encoding="utf-8-sig"
)

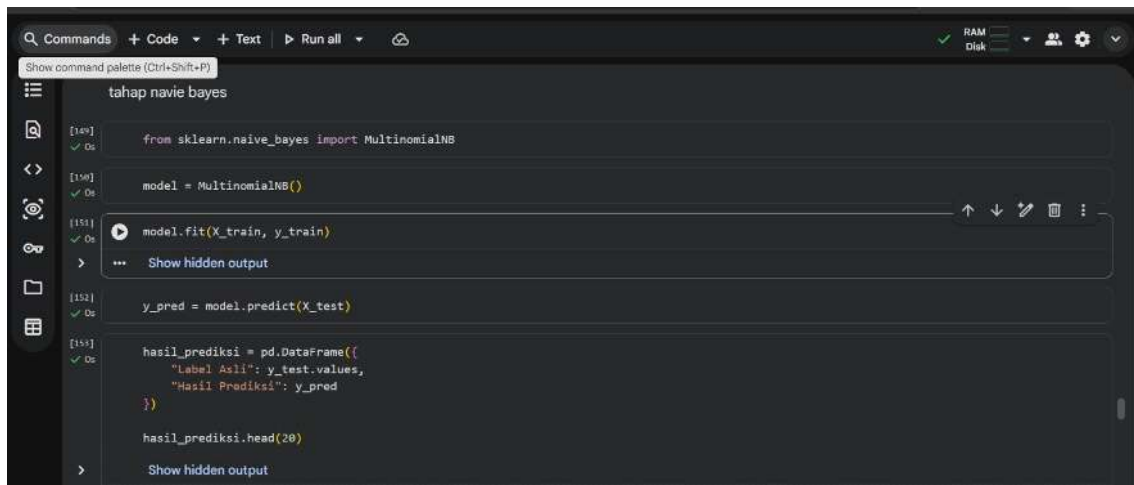
test_df.to_csv(
    "/content/drive/MyDrive/400 data dpemtpsp/data_test.csv",
    index=False,
    encoding="utf-8-sig"
)

print("Data train dan test berhasil disimpan.")

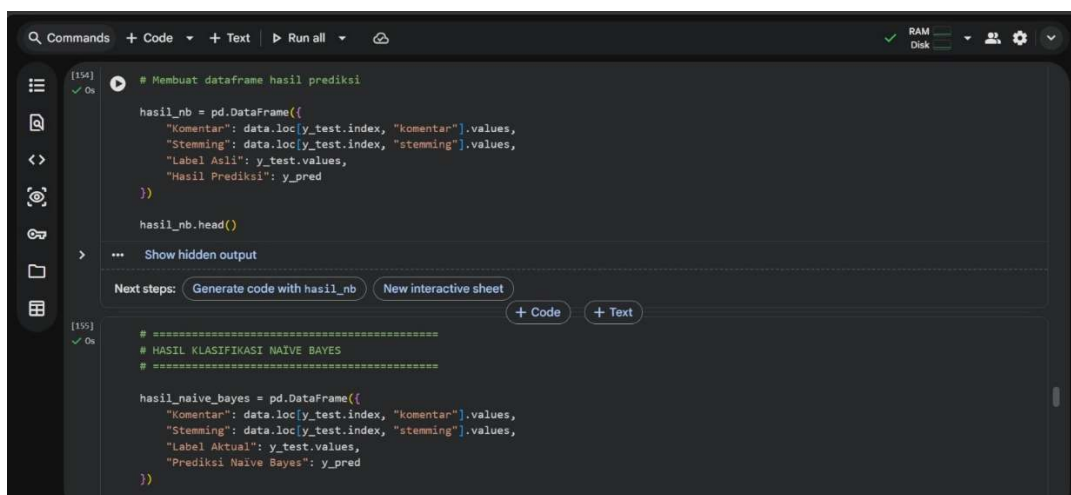
... Data train dan test berhasil disimpan.

[148] ✓ 0s print(y_train.value_counts())
print(y_test.value_counts())
```

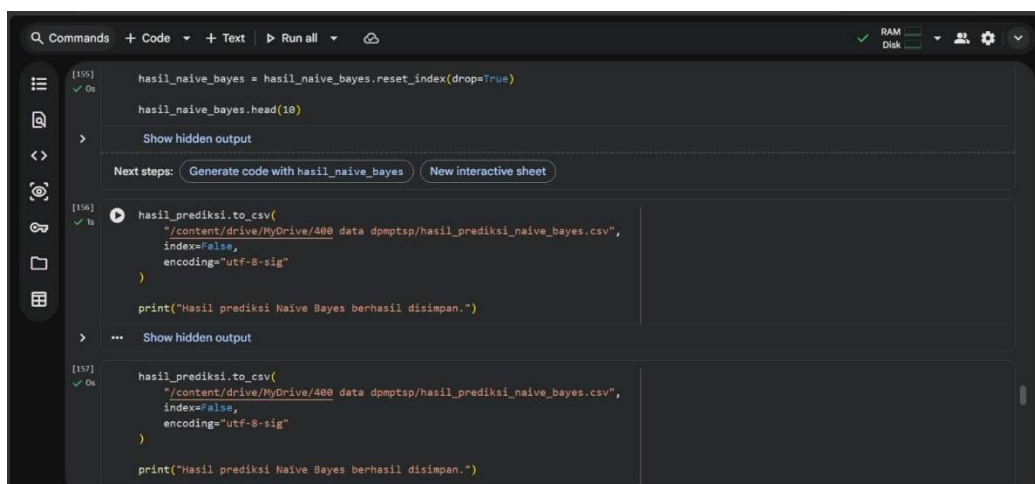
Proses Tahapan Split Data (80:20)



```
[149] from sklearn.naive_bayes import MultinomialNB
[150] model = MultinomialNB()
[151] model.fit(X_train, y_train)
[152] y_pred = model.predict(X_test)
[153] hasil_prediksi = pd.DataFrame({
    "Label Asli": y_test.values,
    "Hasil Prediksi": y_pred
})
hasil_prediksi.head(20)
```



```
[154] # Membuat dataframe hasil prediksi
hasil_nb = pd.DataFrame({
    "komentar": data.loc[y_test.index, "komentar"].values,
    "stemming": data.loc[y_test.index, "stemming"].values,
    "Label Asli": y_test.values,
    "Hasil Prediksi": y_pred
})
hasil_nb.head()
```



```
[155] hasil_naive_bayes = hasil_naive_bayes.reset_index(drop=True)
hasil_naive_bayes.head(10)
[156] hasil_prediksi.to_csv(
    "/content/drive/MyDrive/400 data dpmtsp/hasil_prediksi_naive_bayes.csv",
    index=False,
    encoding="utf-8-sig"
)
print("Hasil prediksi Naive Bayes berhasil disimpan.")
```

Proses Tahapan Naïve Bayes

```
evalulasi model

[158] ✓ 0s from sklearn.metrics import accuracy_score
akurasi = accuracy_score(y_test,y_pred)
print("Accuracy :",akurasi)
> Show hidden output

[159] ✓ 0s from sklearn.metrics import precision_score
precision = precision_score(
    y_test,
    y_pred,
    average="weighted"
)
print("Precision :",precision)
> ... Show hidden output

[160] ✓ 0s from sklearn.metrics import recall_score
recall = recall_score(
    y_test,
```

```

    y_pred,
    average="weighted"
)
print("F1-Score :",f1)
> Show hidden output

[162] ✓ 0s from sklearn.metrics import classification_report
print(classification_report(
    y_test,
    y_pred
))
> ... Show hidden output

[163] ✓ 0s evaluasi = pd.DataFrame([
    "Metode":["Naive Bayes"],
    "Accuracy":[round(akurasi*100,2)],
    "Precision":[round(precision*100,2)],
    "Recall":[round(recall*100,2)],
    "F1-Score":[round(f1*100,2)]
])

evaluasi
```

```
[164] ✓ 0s evaluasi.to_csv(
    "/content/drive/MyDrive/400 data dpmtsp/evaluasi_model_naive_bayes.csv",
    index=False,
    encoding="utf-8-sig"
)
```

Proses Tahapan Evaluasi Model

```
confusion matrix

[165] ✓ 0s
from sklearn.metrics import confusion_matrix

cm = confusion_matrix(
    y_test,
    y_pred,
    labels=["Positif", "Netral"]
)

print(cm)

Show hidden output

[171] ✓ 1s
from sklearn.metrics import ConfusionMatrixDisplay
import matplotlib.pyplot as plt

disp = ConfusionMatrixDisplay(
    confusion_matrix=cm,
    display_labels=["Positif", "Netral"]
)

disp.plot(cmap="Blues")

plt.title("Confusion Matrix Naive Bayes")
plt.show()
```

Proses Tahapan Confusion Matrix

```
wordcloud positif

[167] ✓ 3s
from wordcloud import WordCloud
import matplotlib.pyplot as plt

text_positif = " ".join(
    data[data["label_sentimen"]=="Positif"]["stemming"].astype(str)
)

wordcloud = WordCloud(
    width=1000,
    height=500,
    background_color='white'
).generate(text_positif)

plt.figure(figsize=(12,6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("WordCloud Sentimen Positif")
plt.show()

... Show hidden output
```

Wordcloud Positif

```
wordcloud netral

[168] ✓ 1s
text_netral = " ".join(
    data[data["label_sentimen"]=="Netral"]["stemming"].astype(str)
)

wordcloud = WordCloud(
    width=1000,
    height=500,
    background_color='white'
).generate(text_netral)

plt.figure(figsize=(12,6))
plt.imshow(wordcloud, interpolation='bilinear')
plt.axis("off")
plt.title("WordCloud Sentimen Netral")
plt.show()

Show hidden output
```

Wordcloud Netral