LAMPIRAN

Lampiran 1. Dataset CNBC Indonesia

Berikut ini merupakan link dataset video dari *channel* youtube CNBC Indonesia: https://www.youtube.com/watch?v=yByY5eoZ_To

Lampiran 2. Dataset Kompas Pontianak

Berikut ini merupakan link dataset video dari *channel* youtube Kompas TV Pontianak : https://www.youtube.com/watch?v=GLleXgKiUW8

Lampiran 3. Dataset Kumparan

Berikut ini merupakan link dataset video dari *channel* youtube Kumparan : https://www.youtube.com/watch?v=Xy6FoBEP1XE

Lampiran 4. *Crawling* Data Komentar Youtube Pada Python

```
def video_comments(video_id):
    # empty list for storing reply
    replies = []
    # creating youtube resource object
    youtube = build('youtube', 'v3', developerKey=api_key)
    # retrieve youtube video results
    video_response =
youtube.commentThreads().list(part='snippet,replies',
videoId=video_id).execute()
    # iterate video response
    while video_response:
       # extracting required info
       # from each result object
       for item in video_response['items']:
          # Extracting comments ()
          published =
item['snippet']['topLevelComment']['snippet']['publishedAt']
          user =
item['snippet']['topLevelComment']['snippet']['authorDisplayN
ame']
          # Extracting comments
          comment =
item['snippet']['topLevelComment']['snippet']['textDisplay']
          replies.append([published, user, comment])
          # counting number of reply of comment
          replycount = item['snippet']['totalReplyCount']
```

```
# if reply is there
       if replycount>0:
          # iterate through all reply
          for reply in item['replies']['comments']:

             # Extract reply
             published = reply['snippet']['publishedAt']
             user = reply['snippet']['authorDisplayName']
             repl = reply['snippet']['textDisplay']

             # Store reply is list
             #replies.append(reply)
             replies.append([published, user, repl])

    # Again repeat
    if 'nextPageToken' in video_response:
       video_response = youtube.commentThreads().list(
             part = 'snippet,replies',
             pageToken = video_response['nextPageToken'],
             videoId = video_id
          ).execute()
    else:
       break
#endwhile
return replies

# isikan dengan api key Anda
api_key = 'AIzaSyDVfqrgqthl96cL7YPm2OSLBjET6fiRzJs'

# Enter video id
# contoh url video =
https://www.youtube.com/watch?v=jETDZzSVMNM
video_id = "yByY5eoZ_To" #isikan dengan kode / ID video

# Call function
comments = video_comments(video_id)
```

Lampiran 5. Cleansing dan Case Folding Pada Python

```
import re
import string
import nltk
from nltk.corpus import stopwords

def preprocess_text(text):
    # Lowercase
    text = str(text).lower()
     # Hapus tag HTML
    text = re.sub(r'<.*?>', ' ', text)
    # Hapus URL
    text = re.sub(r'http\S+|www\S+|https\S+', ' ', text)
       # Hapus username (@username)
    text = re.sub(r'@\w+', ' ', text)
     # Hapus emoticon / emoji
    emoji_pattern = re.compile("["
        u"\U0001F600-\U0001F64F"  # emotikon wajah
        u"\U0001F300-\U0001F5FF"  # simbol & pictograf
        u"\U0001F680-\U0001F6FF"  # transportasi & simbol peta
        u"\U0001F1E0-\U0001F1FF"  # bendera (flag)
        u"\U00002700-\U000027BF"  # simbol dingin & dingin
        u"\U000024C2-\U0001F251"
        "]+", flags=re.UNICODE)
    text = emoji_pattern.sub(r'', text)
    # Hapus non ASCII (emoticon, chinese word, .etc)
    text = text.encode('ascii', 'replace').decode('ascii')
    # Hapus tanda baca
    text = re.sub(f"[{re.escape(string.punctuation)}]", " ", text)
    # Hapus angka
    text = re.sub(r'\d+', ' ', text)
    # Hapus spasi berlebih
    text = text.strip()
    text = re.sub(r'\s+', ' ', text)
    return text

# Contoh: jika ingin subset, gunakan copy()
df = df.copy()  # atau data[some_condition].copy()

# Lalu assign dengan .loc
df['Pre'] = df['Comment'].apply(preprocess_text)

# Hapus baris dengan komentar kosong atau hanya spasi
df = df[df['Pre'].str.strip() != '']
df
```

Lampiran 6. Tokenize Pada Python

```
from nltk.tokenize import word_tokenize


# Tokenization
def word_tokenize_wrapper(text):
    return word_tokenize(text)

df['Pre'] = df['Pre'].apply(word_tokenize_wrapper)
```

Lampiran 7. Normalize Pada Python

```
import pandas as pd
import swifter
import ast

# Bersihkan nama kolom dari spasi jika ada
normalizad_word.columns = normalizad_word.columns.str.strip()

# Buat dictionary normalisasi: kata_tidak_baku -> kata_baku
normalizad_word_dict =
dict(zip(normalizad_word['kata_tidak_baku'],
normalizad_word['kata_baku']))

# Fungsi untuk normalisasi token list
def normalized_term(tokens):
    return [normalizad_word_dict.get(token, token) for token
in tokens]

df['Normalize'] = df['Pre'].apply(normalized_term)
```

Lampiran 8. Stopword Pada Python

```
import pandas as pd
import nltk
from nltk.corpus import stopwords
import ast
nltk.download('stopwords')

# Jika kolom token masih berupa string
df['Normalize'] = df['Normalize'].apply(lambda x:
ast.literal_eval(x) if isinstance(x, str) else x)

# Stopword dasar Bahasa Indonesia
stop_words = set(stopwords.words('indonesian'))

stop_words.update(custom_stopwords)

stop_words = stop_words - whitelist

# Fungsi stopword removal
def remove_stopwords(tokens):
    return [word for word in tokens if word not in
stop_words]

# Terapkan stopword removal
df['Stop'] = df['Normalize'].apply(remove_stopwords)
```

Lampiran 9. Stemming Pada Python

```
import pandas as pd
import swifter
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
import ast

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

# Daftar kata yang tidak ingin di-stem (semua lowercase)
whitelist =
['setuju','senang','tujuan','nilai','kurang','kurangi','senil
ai','sekedar','mentri','pemangku','rekening','bali']

# Fungsi stemming per token dengan pengecualian whitelist
def stem_tokens(tokens):
    return [token if token.lower() in whitelist else
stemmer.stem(token) for token in tokens]

# Terapkan fungsi dengan swifter
df['Stemmed'] = df['Stemmed'].swifter.apply(stem_tokens)
```

Lampiran 10. Ekstraksi Fitur TF-IDF Pada Python

```
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer,
CountVectorizer

X = df['Stemmed']  # Kolom teks
y = df['Sentimen']  # Kolom label

# 2. TF-IDF Vectorizer
tfidf = TfidfVectorizer()
X_tfidf = tfidf.fit_transform(X)
```

Lampiran 11. Split Data Pada Python

```
import pandas as pd
from sklearn.model_selection import train_test_split

# 3. Split data (80:20)
X_train, X_test, y_train, y_test = train_test_split(
    X_tfidf, y, test_size=0.2, random_state=42
)
```

Lampiran 12. Balancing Data Dengan SMOTE Pada Python

```
import pandas as pd
from imblearn.over_sampling import SMOTE

smote = SMOTE(random_state=42)
X_train_smote, y_train_smote = smote.fit_resample(X_train,
y_train)

)
```

Lampiran 13. Fungsi Klasifikasi Pada Python

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.naive_bayes import MultinomialNB
from sklearn.svm import SVC
from sklearn.metrics import (
    classification_report, confusion_matrix,
    accuracy_score, precision_score, recall_score, f1_score
)

def tampilkan_evaluasi(y_true, y_pred):
    print(f"Accuracy:  {accuracy_score(y_true, y_pred):.4f}")
    print(f"Precision: {precision_score(y_true, y_pred,
average='weighted'):.4f}")
    print(f"Precision positif: {precision_score(y_true,
y_pred, pos_label='Positif'):.4f}")
    print(f"Precision negatif: {precision_score(y_true,
y_pred, pos_label='Negatif'):.4f}")
    print(f"Recall:    {recall_score(y_true, y_pred,
average='weighted'):.4f}")
    print(f"Recall positif:    {recall_score(y_true, y_pred,
pos_label='Positif'):.4f}")
    print(f"Recall negatif:    {recall_score(y_true, y_pred,
pos_label='Negatif'):.4f}")
    print(f"F1-Score:  {f1_score(y_true, y_pred,
average='weighted'):.4f}")
    print(f"F1-Score positif:  {f1_score(y_true, y_pred,
pos_label='Positif'):.4f}")
    print(f"F1-Score negatif:  {f1_score(y_true, y_pred,
pos_label='Negatif'):.4f}")

def tampilkan_confusion_matrix(y_true, y_pred,
title='Confusion Matrix'):
    # Tentukan urutan label secara eksplisit
    labels = ['Positif', 'Negatif']
    cm = confusion_matrix(y_true, y_pred, labels=labels)

    plt.figure(figsize=(5, 4))
    sns.heatmap(cm, annot=True, fmt='d', cmap='Blues',
                xticklabels=labels, yticklabels=labels)
    plt.title(title)
    plt.xlabel('Predicted')
    plt.ylabel('Actual')
    plt.tight_layout()
    plt.show()
```

```
# 4. ============================
#    Tanpa SMOTE (data asli)
#    ============================

print("\n=== Naive Bayes (tanpa SMOTE) ===")
nb = MultinomialNB()
nb.fit(X_train, y_train)
y_pred_nb = nb.predict(X_test)
tampilkan_confusion_matrix(y_test, y_pred_nb, title="Naive
Bayes - Tanpa SMOTE")
print(classification_report(y_test, y_pred_nb))
tampilkan_evaluasi(y_test, y_pred_nb)

print("\n=== SVM (tanpa SMOTE) ===")
svm = SVC()
svm.fit(X_train, y_train)
y_pred_svm = svm.predict(X_test)
tampilkan_confusion_matrix(y_test, y_pred_svm, title="SVM -
Tanpa SMOTE")
print(classification_report(y_test, y_pred_svm))
tampilkan_evaluasi(y_test, y_pred_svm)

# ============================
# 5. Dengan SMOTE
# ============================

print("\n=== Naive Bayes (dengan SMOTE) ===")
nb_sm = MultinomialNB()
nb_sm.fit(X_train_smote, y_train_smote)
y_pred_nb_sm = nb_sm.predict(X_test)
tampilkan_confusion_matrix(y_test, y_pred_nb_sm, title="Naive
Bayes - Dengan SMOTE")
print(classification_report(y_test, y_pred_nb_sm))
tampilkan_evaluasi(y_test, y_pred_nb_sm)

print("\n=== SVM (dengan SMOTE) ===")
svm_sm = SVC()
svm_sm.fit(X_train_smote, y_train_smote)
y_pred_svm_sm = svm_sm.predict(X_test)
tampilkan_confusion_matrix(y_test, y_pred_svm_sm, title="SVM
- Dengan SMOTE")
print(classification_report(y_test, y_pred_svm_sm))
tampilkan_evaluasi(y_test, y_pred_svm_sm)
```

```
print("\n=== K-FOLD CROSS VALIDATION (Tanpa SMOTE) ===")
kf = KFold(n_splits=5, shuffle=True, random_state=42)

def evaluasi_kfold(model, X, y, name='Model'):
    acc_list, prec_list, rec_list, f1_list = [], [], [], []
    fold = 1
    for train_index, test_index in kf.split(X):
        X_tr, X_te = X[train_index], X[test_index]
        y_tr, y_te = y.iloc[train_index], y.iloc[test_index]

        model.fit(X_tr, y_tr)
        y_pred = model.predict(X_te)

        acc = accuracy_score(y_te, y_pred)
        prec = precision_score(y_te, y_pred,
average='weighted')
        rec = recall_score(y_te, y_pred, average='weighted')
        f1 = f1_score(y_te, y_pred, average='weighted')

        acc_list.append(accuracy_score(y_te, y_pred))
        prec_list.append(precision_score(y_te, y_pred,
average='weighted'))
        rec_list.append(recall_score(y_te, y_pred,
average='weighted'))
        f1_list.append(f1_score(y_te, y_pred,
average='weighted'))

        print(f"\nFold {fold} - {name}")
        print(classification_report(y_te, y_pred))
        print(f"Akurasi Fold {fold}: {acc:.4f}")
        fold += 1

    print(f"\n{name} - Rata-rata Skor:")
    print(f"Accuracy:  {np.mean(acc_list):.4f}")
    print(f"Precision: {np.mean(prec_list):.4f}")
    print(f"Recall:    {np.mean(rec_list):.4f}")
    print(f"F1-Score:  {np.mean(f1_list):.4f}")

# Jalankan K-Fold
evaluasi_kfold(MultinomialNB(), X_tfidf, y, name='Naive Bayes
(Tanpa SMOTE)')

evaluasi_kfold(SVC(), X_tfidf, y, name='SVM (Tanpa SMOTE)')
```

Lampiran 16. Pengujian K-Fold Dengan SMOTE Pada Python

```
# ====================================
# 7. K-Fold Dengan SMOTE
# ====================================
print("\n=== K-FOLD CROSS VALIDATION (Dengan SMOTE) ===")

def evaluasi_kfold_smote(model, X, y, name='Model + SMOTE'):
    acc_list, prec_list, rec_list, f1_list = [], [], [], []
    fold = 1
    for train_index, test_index in kf.split(X):
        X_tr, X_te = X[train_index], X[test_index]
        y_tr, y_te = y.iloc[train_index], y.iloc[test_index]

        X_tr_res, y_tr_res =
SMOTE(random_state=42).fit_resample(X_tr, y_tr)

        model.fit(X_tr_res, y_tr_res)
        y_pred = model.predict(X_te)

        acc = accuracy_score(y_te, y_pred)
        prec = precision_score(y_te, y_pred,
average='weighted')
        rec = recall_score(y_te, y_pred, average='weighted')
        f1 = f1_score(y_te, y_pred, average='weighted')

        acc_list.append(accuracy_score(y_te, y_pred))
        prec_list.append(precision_score(y_te, y_pred,
average='weighted'))
        rec_list.append(recall_score(y_te, y_pred,
average='weighted'))
        f1_list.append(f1_score(y_te, y_pred,
average='weighted'))

        print(f"\nFold {fold} - {name}")
        print(classification_report(y_te, y_pred))
        print(f"Akurasi Fold {fold}: {acc:.4f}")
        fold += 1

    print(f"\n{name} - Rata-rata Skor:")
    print(f"Accuracy:  {np.mean(acc_list):.4f}")
    print(f"Precision: {np.mean(prec_list):.4f}")
    print(f"Recall:    {np.mean(rec_list):.4f}")
    print(f"F1-Score:  {np.mean(f1_list):.4f}")

# Jalankan K-Fold + SMOTE
evaluasi_kfold_smote(MultinomialNB(), X_tfidf, y, name='Naive
Bayes + SMOTE')

evaluasi_kfold_smote(SVC(), X_tfidf, y, name='SVC RBF +
SMOTE')
```

Lampiran 17. WordCloud Pada Python

```
from wordcloud import WordCloud

def tampilkan_wordcloud(teks, judul):
    wordcloud = WordCloud(
        width=800,
        height=400,
        background_color='white',
        max_words=200
    ).generate(" ".join(teks))

    plt.figure(figsize=(10, 5))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis("off")
    plt.title(judul, fontsize=16)
    plt.tight_layout()
    plt.show()


tampilkan_wordcloud(df['Stemmed'], "WordCloud - Semua Data")

# WordCloud Positif
teks_positif = df[df['Sentimen'] == 'Positif']['Stemmed']
tampilkan_wordcloud(teks_positif, "WordCloud - Sentimen
Positif")

# WordCloud Negatif
teks_positif = df[df['Sentimen'] == 'Negatif']['Stemmed']
tampilkan_wordcloud(teks_positif, "WordCloud - Sentimen
Negatif")
```

Lampiran 18. Pengujian Overfitting Data Pada Python

```
def check_overfitting(model, X_train, y_train, X_test,
y_test):
    train_acc = accuracy_score(y_train,
model.predict(X_train))
    test_acc = accuracy_score(y_test, model.predict(X_test))

    print(f"Akurasi Training : {train_acc:.4f}")
    print(f"Akurasi Testing  : {test_acc:.4f}")

# Buat model
modelnb = MultinomialNB()

# Training model
modelnb.fit(X_train, y_train)

# Cek overfitting
check_overfitting(modelnb, X_train, y_train, X_test, y_test)
```

Lampiran 20. Dataset yang digunakan

Date	UserName	Comment	Ahli 1	Ahli 2	Ahli 3	Sentimen
2025-05-02T06:57:36Z	@SupriyadiYadi-h7l	Klau ingin cpat maju,hrs cpat dilaksanakan,biar sama negara2 maju,karena ini akan mempermudah	Negatif	Negatif	Positif	Negatif
2025-03-30T17:47:21Z	@WawanHariono-j8j	Setuju sja asalkan desainnya tetap sma biar Masyarakat tdk bingung	Positif	Positif	Negatif	Positif
2025-03-28T01:41:57Z	@MrIhwan-z5h	Kasian uang yg jadi korban , sepatutnya yg dipotong para mafia	Negatif	Negatif	Negatif	Negatif
2025-03-27T15:09:12Z	@nurhasanaana3129	jangan lama segera	Negatif	Positif	Negatif	Negatif
2025-03-24T17:26:30Z	@sopyansopyan8552	Gong apa h gong kematian datang baru terjadi ini dah 2025 belum juga terjadi berbagai alasan yg di beri dan di cerita kan namun tetap juga semua nya dusta rakyat dah jenuh menanti keputusan BI namun apa hasil nya semua nya pendusta dan penipu	Negatif	Negatif	Positif	Negatif
2025-03-21T11:39:55Z	@fajarismanto2063	Omong doang ga rilis2	Negatif	Negatif	Negatif	Negatif
2025-03-19T10:39:08Z	@maharimahar2625	Hanya harapan doang / harapan hanya tinggal harapan. Redenominasi bisa terwujut setelah hari kiamat.	Negatif	Negatif	Negatif	Negatif
2025-03-19T05:54:51Z	@yusraalro710	Rupiah kembali ke masa jayanya sejak dari uang Golden Belanda,menjadi Rupiah yg sederhana dan sehat.dgn nilai sederHana. Contohnya Ringit mata uang Nusantara yang nilai Ringit 1 Ringit 2 1/2 Rupiah.berarti Sekarang Rp.1000..menjadi Rp 1,_ sdh final kembali ke awal .jadi Ringgit bukan mata uang Malaysia aslinya Uang Nusantara..dulu ada Rupiah emas,Ringgit Emas yg nilainya besar kalau di dolarkan.	Positif	Positif	Positif	Positif

2025-03-18T21:13:56Z	@JosephPasali	Omong kosong suda bagus d kotak Katik mau d blg pintar tp menyusahkan org banyak apapun alasannya itu akal2an apa urusannya dgn barang n yg lain bahasanya saja suda salah nanti bermasalah minta maaf klu yg mau d rubah adalah upah d naikkan itu baru hebat ini aman2 saja d ganggu cari kerjaan mau d blg hebat oh itu ide saya klu mau jago yg blm ada d adakan itu baru jago contoh air asin jadi tawar itu baru jago/hebat ini suda bagus mau d rubah makasih masukan saya	Negatif	Negatif	Negatif	Negatif
2025-03-11T08:48:06Z	@dianringo-hr8oz	TOLAK REDENOMINASI,PENIPUAN Pemerintah Kepada RAKYAT	Negatif	Negatif	Negatif	Negatif
2025-03-11T08:42:52Z	@dianringo-hr8oz	REDENOMINASI Inilah Yang Membuat IHSG MeMeRAH	Negatif	Negatif	Positif	Negatif
2025-03-05T13:19:18Z	@mudakiras8737	AH NGOMONG DOANG DARI 2TAHUN YG LALU SUDAH DI BERITAKAN SAMPAI SAAT INI BELUM TER REALISASI , DI LAKSANAKAN ! .	Negatif	Negatif	Negatif	Negatif
2025-03-01T19:14:26Z	@sukrasukrabae	Cuma wacana doang. Padahal masarakat sudah bosan menungu kebijakan ini	Negatif	Negatif	Negatif	Negatif
2025-03-01T01:15:35Z	@NajlaNajla-c4f	Di daerah Kediri sudah biasa rakyat mengatakan satu juta mengatakan seribu,dan seratus ribu ,mengatakan seratus saja dlm jual beli	Positif	Positif	Positif	Positif
2025-02-27T02:56:37Z	@AgusBru-ck6wu	LEBIH CEPAT LEBIH BAIK JANGAN DITUNDA TUNDA DONG BOSS	Positif	Positif	Positif	Positif