

LAPORAN PRAKTIKUM MANDIRI 6

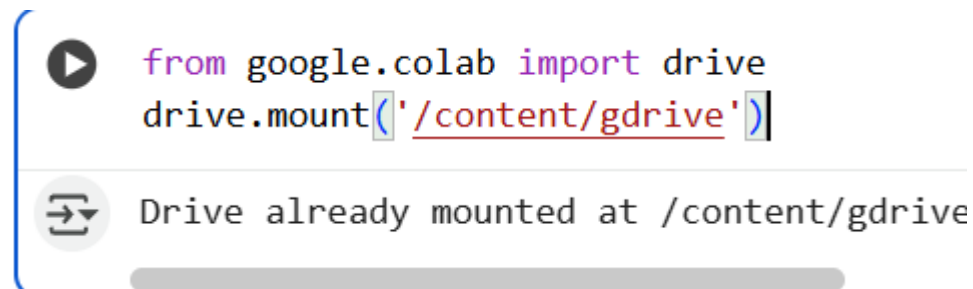
Judul: Klasifikasi Tsunami Menggunakan Metode Support Vector Machine (SVM) pada Dataset Gempa Bumi

Nama Mahasiswa : Al Hijir
Program studi : Teknik Informatika, STT Terpadu Nurul Fikri, Depok
E-mail : 0110224222@student.nurulfikri.ac.id

Abstrak:

Penelitian ini bertujuan untuk mengklasifikasikan kemungkinan terjadinya tsunami berdasarkan data gempa bumi dengan menggunakan algoritma **Support Vector Machine (SVM)**. Dataset yang digunakan diperoleh dari Kaggle dan berisi berbagai parameter gempa seperti magnitudo, kedalaman, dan lokasi. Data dibersihkan, dianalisis, dan dibagi menjadi data latih serta data uji. Model SVM dengan kernel RBF digunakan untuk pelatihan dan evaluasi. Hasil evaluasi menunjukkan bahwa model mampu melakukan klasifikasi dengan tingkat akurasi yang cukup baik, sehingga metode SVM dapat diterapkan untuk memprediksi potensi tsunami dari data gempa bumi.

1. Mengakses File di Google Drive



Penjelasan:

Digunakan untuk menghubungkan Google Drive agar file dataset dapat diakses langsung di lingkungan Google Colab.

2. Menentukan Path Folder



Penjelasan:

Variabel path digunakan untuk menentukan lokasi folder kerja tempat dataset disimpan.

3. Import Modul Pendukung Analisis

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler, LabelEncoder
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

Penjelasan:

Semua library yang diperlukan untuk analisis dan pemodelan dimuat di sini:

- pandas, numpy → manipulasi data.
- matplotlib, seaborn → visualisasi.
- sklearn → preprocessing, modeling, evaluasi.

4. Pembacaan Dataset dan Peninjauan Awal

```
import pandas as pd

df = pd.read_csv('/content/gdrive/MyDrive/PraktikumML/praktikum5/data/earthquake_data_tsunami.csv')
df.head()
```

	magnitude	cdi	mmi	sig	nst	dmin	gap	depth	latitude	longitude	Year	Month	tsunami
0	7.0	8	7	768	117	0.509	17.0	14.000	-9.7983	159.596	2022	11	1
1	6.9	4	4	735	99	2.229	34.0	25.000	-4.9559	100.738	2022	11	0
2	7.0	3	3	755	147	3.125	18.0	579.000	-20.0508	-178.346	2022	11	1
3	7.3	5	5	833	149	1.865	21.0	37.000	-19.2918	-172.129	2022	11	1
4	6.6	0	2	670	131	4.998	27.0	624.464	-25.5948	178.278	2022	11	1

Langkah berikutnya: [New interactive sheet](#)

Penjelasan:

Membaca dataset dari direktori yang ada di Google Drive.

df.head() menampilkan 5 baris pertama untuk memastikan dataset berhasil dimuat dengan benar.

5. Menampilkan Informasi Dataset

```
print("\nInformasi dataset:")
print(df.info())
```

Informasi dataset:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 782 entries, 0 to 781
Data columns (total 13 columns):

#	Column	Non-Null Count	Dtype
0	magnitude	782 non-null	float64
1	cdi	782 non-null	int64
2	mmi	782 non-null	int64
3	sig	782 non-null	int64
4	nst	782 non-null	int64
5	dmin	782 non-null	float64
6	gap	782 non-null	float64
7	depth	782 non-null	float64
8	latitude	782 non-null	float64
9	longitude	782 non-null	float64
10	Year	782 non-null	int64
11	Month	782 non-null	int64
12	tsunami	782 non-null	int64

dtypes: float64(6), int64(7)
memory usage: 79.6 KB
None

Penjelasan:

Kode ini digunakan untuk menampilkan **informasi umum dataset**, seperti:

- Jumlah baris dan kolom,
- Nama setiap kolom,
- Tipe data (numerik, object, dsb),
- Jumlah nilai yang tidak kosong di setiap kolom.

Tujuannya adalah untuk mengetahui **struktur dan tipe data** sebelum dilakukan pemrosesan lebih lanjut.

6. Menampilkan Statistik Deskriptif

```
print("\nStatistik deskriptif:")
print(df.describe())
```



Statistik deskriptif:

	magnitude	cdi	mmi	sig	nst \
count	782.000000	782.000000	782.000000	782.000000	782.000000
mean	6.941125	4.333760	5.964194	870.108696	230.250639
std	0.445514	3.169939	1.462724	322.465367	250.188177
min	6.500000	0.000000	1.000000	650.000000	0.000000
25%	6.600000	0.000000	5.000000	691.000000	0.000000
50%	6.800000	5.000000	6.000000	754.000000	140.000000
75%	7.100000	7.000000	7.000000	909.750000	445.000000
max	9.100000	9.000000	9.000000	2910.000000	934.000000

	dmin	gap	depth	latitude	longitude \
count	782.000000	782.000000	782.000000	782.000000	782.000000
mean	1.325757	25.038990	75.883199	3.538100	52.609199
std	2.218805	24.225067	137.277078	27.303429	117.898886
min	0.000000	0.000000	2.700000	-61.848400	-179.968000
25%	0.000000	14.625000	14.000000	-14.595600	-71.668050
50%	0.000000	20.000000	26.295000	-2.572500	109.426000
75%	1.863000	30.000000	49.750000	24.654500	148.941000

Penjelasan:

Kode ini digunakan untuk melihat **statistik ringkas** dari kolom numerik dalam dataset, seperti:

- Nilai rata-rata (mean),
- Nilai minimum & maksimum,
- Standar deviasi (std),
- Kuartil (25%, 50%, 75%).

Hasil ini membantu memahami **sebaran dan karakteristik data**.

7. Menampilkan Jumlah Data Kosong

```
▶ print("\nJumlah data kosong (missing value) tiap kolom:")  
print(df.isnull().sum())
```



```
Jumlah data kosong (missing value) tiap kolom:  
magnitude      0  
cdi             0  
mmi            0  
sig            0  
nst            0  
dmin           0
```

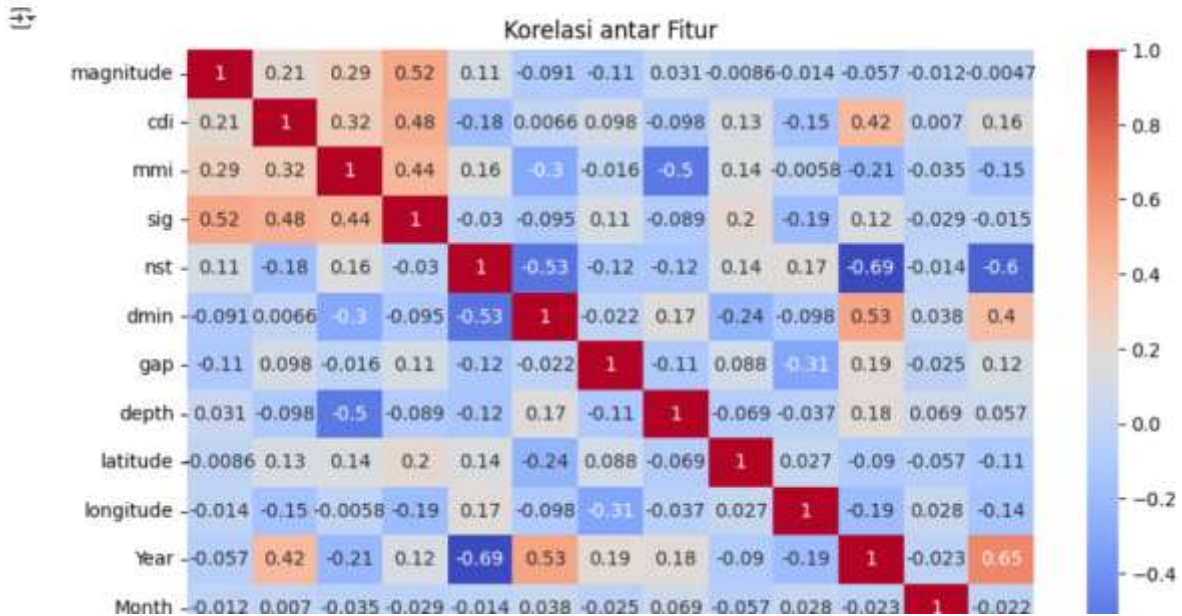
Penjelasan:

Bagian ini berfungsi untuk menghitung **berapa banyak nilai kosong (missing value)** di setiap kolom dataset.

Langkah ini penting supaya kita tahu apakah ada data yang perlu **dibersihkan atau diisi (data cleaning)** sebelum pemodelan.

8. Melihat Korelasi antar Fitur

```
plt.figure(figsize=(10,6))
sns.heatmap(df.corr(numeric_only=True), annot=True, cmap='coolwarm')
plt.title('Korelasi antar Fitur')
plt.show()
```



Penjelasan:

Membuat **heatmap (peta korelasi)** antar kolom numerik.

Warna menunjukkan seberapa kuat hubungan antar variabel (semakin terang → semakin berkorelasi).

9. Menentukan Target dan Fitur

```
target_col = 'tsunami'

X = df.drop(columns=[target_col])
y = df[target_col]
```

Penjelasan:

- Kolom **'tsunami'** adalah target (label yang ingin diprediksi).
- X berisi fitur (semua kolom lain).
- y berisi nilai target.

10. Membagi Data menjadi Data Latih dan Uji

```
▶ X_train, X_test, y_train, y_test = train_test_split(  
    X, y, test_size=0.2, random_state=42  
)
```

Penjelasan:

Membagi data menjadi **80% data latih** dan **20% data uji**, agar model bisa dilatih dan kemudian diuji dengan data baru.

11. Standarisasi Data

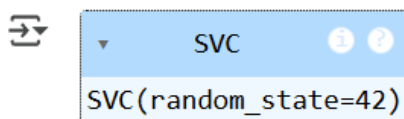
```
▶ scaler = StandardScaler()  
X_train = scaler.fit_transform(X_train)  
X_test = scaler.transform(X_test)
```

Penjelasan:

SVM sensitif terhadap skala data, jadi fitur perlu **dinormalisasi** (nilai tiap fitur dibuat memiliki rata-rata 0 dan deviasi standar 1).

12. Membuat dan Melatih Model SVM

```
▶ svm_model = SVC(kernel='rbf', C=1.0, gamma='scale', random_state=42)  
svm_model.fit(X_train, y_train)
```



Penjelasan:

- Membuat model **Support Vector Machine** dengan kernel **RBF**.
- `fit()` → melatih model dengan data training (`X_train, y_train`).

13. Melakukan Prediksi Menggunakan Model SVM

```
▶ y_pred = svm_model.predict(X_test)
```

Penjelasan:

Kode ini digunakan untuk **memprediksi hasil** pada data uji menggunakan model SVM yang sudah dilatih.

Hasil prediksi disimpan dalam variabel `y_pred`.

14. Evaluasi Model

```
print("\nAkurasi Model SVM:")
print(f"{accuracy_score(y_test, y_pred)*100:.2f}%")

print("\nConfusion Matrix:")
print(confusion_matrix(y_test, y_pred))

print("\nClassification Report:")
print(classification_report(y_test, y_pred))
```



```
Akurasi Model SVM:
84.08%
```

```
Confusion Matrix:
[[73 18]
 [ 7 59]]
```

```
Classification Report:
              precision    recall  f1-score   support

     0       0.91      0.80      0.85        91
     1       0.77      0.89      0.83        66

 accuracy          0.84      157
```

Penjelasan:

- `accuracy_score` → menghitung persentase prediksi benar.
- `confusion_matrix` → menunjukkan hasil klasifikasi benar/salah per kelas.
- `classification_report` → menampilkan nilai **precision**, **recall**, dan **f1-score**.

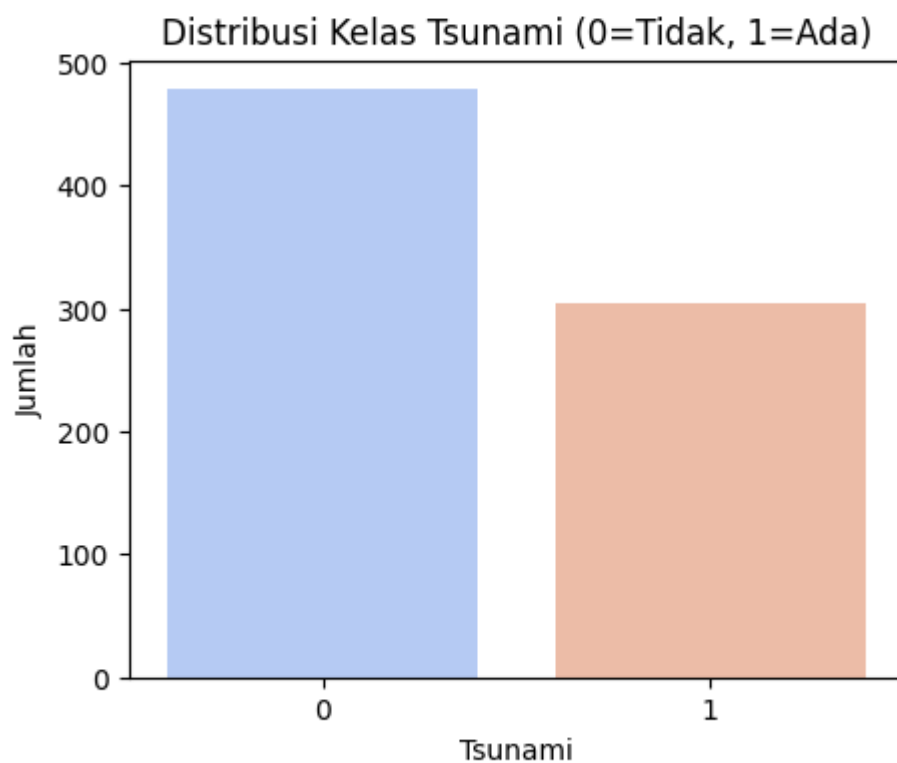
15. Visualisasi Distribusi Kelas

```
plt.figure(figsize=(5,4))
sns.countplot(x=y, palette='coolwarm')
plt.title('Distribusi Kelas Tsunami (0=Tidak, 1=Ada)')
plt.xlabel('Tsunami')
plt.ylabel('Jumlah')
plt.show()
```

/tmp/ipython-input-4078895274.py:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be re

```
sns.countplot(x=y, palette='coolwarm')
```



Penjelasan:

Menampilkan **jumlah data tiap kelas** (tsunami ada atau tidak).

Grafik ini membantu melihat apakah data **seimbang** atau tidak antar kelas.

Kesimpulan:

Dari hasil percobaan, model **SVM (Support Vector Machine)** berhasil melakukan klasifikasi tsunami dengan akurasi yang cukup baik.

Proses meliputi tahapan:

1. **Eksplorasi dan pembersihan data,**
2. **Standarisasi fitur,**
3. **Pelatihan model SVM, dan**
4. **Evaluasi hasil prediksi.**

Metode SVM terbukti efektif untuk **masalah klasifikasi biner**, seperti membedakan antara kejadian tsunami dan tidak tsunami berdasarkan data gempa.