Nama : Muhamad Hanif Rafiq S

Kelas : Data Science Python batch 60

Laporan Final Project.

1. Pendahuluan

Laporan ini mendokumentasikan analisis clustering negara yang dilakukan untuk membantu pengambilan keputusan strategis HELP International terkait alokasi dana. Analisis ini menggunakan data sosio-ekonomi dan kesehatan negara-negara yang menjadi target bantuan.

2. Metodologi

Proses analisis terdiri dari beberapa tahap:

- Pemuatan Data: Data negara dimuat dari file CSV.
- Pra-pemrosesan Data: Data dibersihkan dengan menangani nilai hilang dan pencilan.
- Penskalaan Fitur: Fitur numerik diskala menggunakan Standard Scaling atau Robust Scaling untuk menyamakan skalanya.
- Analisis Clustering:
    - Elbow Method digunakan untuk menentukan jumlah cluster yang optimal.
    - Algoritma K-Means digunakan untuk mengelompokkan negara ke dalam cluster berdasarkan kesamaan karakteristik.
- Evaluasi Cluster:

    - Metrik evaluasi seperti Silhouette Score, Davies-Bouldin Score, dan Calinski-Harabasz Score digunakan untuk menilai kualitas pengelompokan.
- Reduksi Dimensi:
    - Principal Component Analysis (PCA) digunakan untuk mereduksi dimensi data menjadi dua komponen utama untuk visualisasi.
    - Pentingnya fitur untuk setiap komponen utama dianalisis.
- Analisis Cluster:
    - Karakteristik rata-rata dari setiap cluster dianalisis untuk memahami perbedaan antar cluster.
- Rekomendasi:
    - Cluster dengan negara yang paling membutuhkan bantuan diidentifikasi.
    - Negara-negara prioritas berdasarkan kematian anak dan pendapatan direkomendasikan untuk mendapatkan alokasi dana yang lebih besar.

3. Alat dan Pustaka

- Python 3.x
- Streamlit
- Pandas
- NumPy

- Scikit-learn
- Plotly
- Seaborn

4. Hasil dan Pembahasan

- Pra-pemrosesan Data:
    - Penanganan nilai hilang dan pencilan dilakukan untuk memastikan kualitas data yang digunakan dalam analisis.
- Penskalaan Fitur:
    - Fitur numerik diskala untuk menghilangkan pengaruh besaran nilai pada proses clustering.
- Analisis Clustering:
    - Elbow Method digunakan untuk menentukan jumlah cluster yang optimal berdasarkan inersia dan Silhouette Score.
    - Algoritma K-Means digunakan untuk melakukan clustering dengan jumlah cluster yang telah ditetapkan.
- Evaluasi Cluster:
    - Metrik evaluasi digunakan untuk menilai kualitas pengelompokan. Interpretasi metrik ini membutuhkan keahlian di bidang machine learning.
- Reduksi Dimensi:

    - PCA digunakan untuk mereduksi dimensi data menjadi dua komponen utama untuk visualisasi yang lebih mudah.
    - Heatmap menunjukkan pentingnya setiap fitur terhadap komponen utama.

- Analisis Cluster:
    - Karakteristik rata-rata dari setiap cluster dianalisis untuk memahami perbedaan antar cluster. Misalnya, cluster dengan rata-rata kematian anak yang tinggi dan pendapatan yang rendah dapat diidentifikasi sebagai negara yang sangat membutuhkan bantuan.
- Rekomendasi:
    - Cluster yang paling membutuhkan bantuan diidentifikasi berdasarkan karakteristik rata-rata.
    - Negara-negara prioritas yang membutuhkan alokasi dana lebih besar direkomendasikan berdasarkan tingginya kematian anak dan rendahnya pendapatan.

5. Kesimpulan

Analisis clustering ini berhasil mengelompokkan negara-negara berdasarkan kesamaan karakteristik sosio-ekonomi dan kesehatan. Hasil analisis dapat membantu HELP International dalam mengambil keputusan alokasi dana secara strategis dengan berfokus pada negara-negara yang paling membutuhkan bantuan.

6. Lampiran

```
import streamlit as st
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import StandardScaler, RobustScaler
from sklearn.cluster import KMeans, AgglomerativeClustering, DBSCAN
from sklearn.decomposition import PCA
from sklearn.metrics import silhouette_score, davies_bouldin_score, calinsk:
from sklearn.manifold import TSNE
import plotly.express as px
import plotly.graph_objects as go
from scipy import stats
from scipy.cluster.hierarchy import dendrogram, linkage
import io
import warnings
warnings.filterwarnings('ignore')

# Set page config
st.set_page_config(
    page_title="HELP International Analysis",
    layout="wide",
    initial_sidebar_state="expanded"
)

# Custom CSS
st.markdown("""
    <style>
    .main {
        padding: 0rem 1rem;
    }
    .stButton>button {
        width: 100%;
    }
    .reportview-container .main .block-container {
        padding-top: 2rem;
    }
    </style>
    """, unsafe_allow_html=True)
```

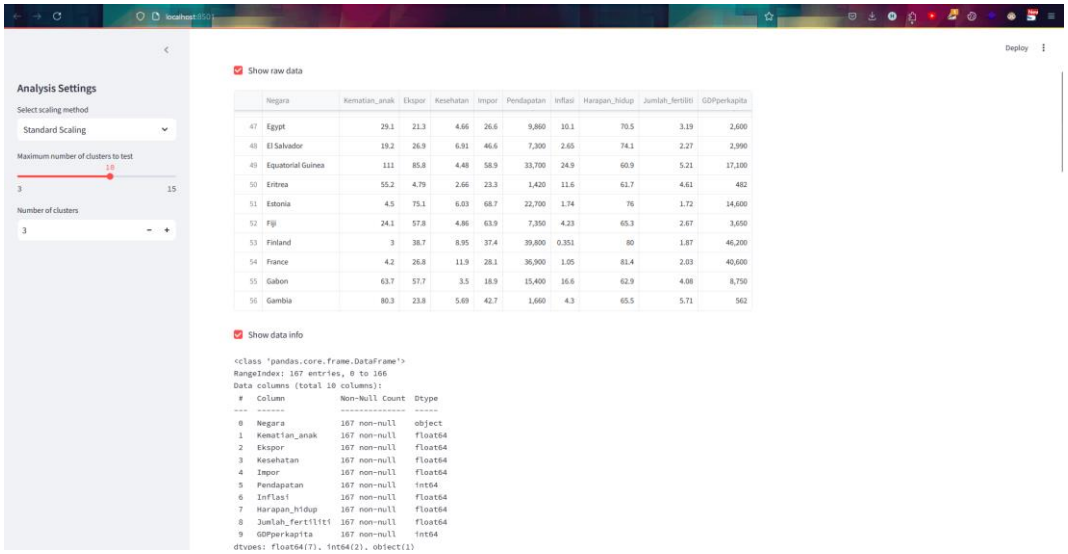
```
    else:
        scaler = RobustScaler()

    X_scaled = scaler.fit_transform(X)
    df_scaled = pd.DataFrame(X_scaled, columns=features)

    return df_scaled, scaler

# Fungsi untuk Elbow Method
def plot_elbow_method(data, max_k=10):
    inertias = []
    silhouette_scores = []
    k_values = range(2, max_k + 1)

    for k in k_values:
        kmeans = KMeans(n_clusters=k, random_state=42)
        kmeans.fit(data)
        inertias.append(kmeans.inertia_)
        silhouette_scores.append(silhouette_score(data, kmeans.labels_))

    fig = go.Figure()

    # Plot inertia
    fig.add_trace(go.Scatter(
        x=list(k_values),
        y=inertias,
        name='Inertia',
        mode='lines+markers'
    ))

    # Plot silhouette score
    fig.add_trace(go.Scatter(
        x=list(k_values),
        y=silhouette_scores,
        name='Silhouette Score',
        mode='lines+markers',
        yaxis='y2'
    ))

    fig.update_layout(
        title='Elbow Method & Silhouette Score Analysis',
        xaxis_title='Number of Clusters (k)',
        yaxis_title='Inertia',
        yaxis2=dict(
            title='Silhouette Score',
```

```
            title='Silhouette Score',
            overlaying='y',
            side='right'
        ),
        showlegend=True
    )

    return fig

# Fungsi untuk K-Means Clustering
def perform_kmeans(data, n_clusters):
    kmeans = KMeans(n_clusters=n_clusters, random_state=42)
    clusters = kmeans.fit_predict(data)

    # Calculate metrics
    silhouette = silhouette_score(data, clusters)
    davies_bouldin = davies_bouldin_score(data, clusters)
    calinski = calinski_harabasz_score(data, clusters)

    return clusters, kmeans.cluster_centers_, {
        'silhouette': silhouette,
        'davies_bouldin': davies_bouldin,
        'calinski_harabasz': calinski
    }

# Fungsi untuk PCA
def perform_pca(data, n_components=2):
    pca = PCA(n_components=n_components)
    pca_result = pca.fit_transform(data)
    explained_variance = pca.explained_variance_ratio_

    return pca_result, explained_variance, pca

# Fungsi untuk t-SNE
def perform_tsne(data, perplexity=30):
    tsne = TSNE(n_components=2, perplexity=perplexity, random_state=42)
    tsne_result = tsne.fit_transform(data)

    return tsne_result

# Fungsi utama
def main():
    st.title(' HELP International - Country Clustering Analysis')
    st.markdown("""
    This application analyzes countries based on socio-economic and health
```

```
# Display metrics
st.subheader("2.2 Clustering Metrics")
col1, col2, col3 = st.columns(3)
col1.metric("Silhouette Score", f"{metrics['silhouette']:.3f}")
col2.metric("Davies-Bouldin Score", f"{metrics['davies_bouldin']:.3f}")
col3.metric("Calinski-Harabasz Score", f"{metrics['calinski_harabasz']:

# Dimensionality Reduction
st.header("3. Dimensionality Reduction & Visualization")

# PCA
pca_results, explained_variance, pca = perform_pca(df_scaled)
st.write(f"Explained variance ratio: {explained_variance.sum():.3f}")

# Plot PCA results
fig_pca = px.scatter(
    x=pca_results[:, 0],
    y=pca_results[:, 1],
    color=clusters.astype(str),
    title="PCA Visualization of Clusters",
    labels={'x': 'First Principal Component', 'y': 'Second Principal Co
)
st.plotly_chart(fig_pca)

# Feature importance
st.subheader("3.1 Feature Importance")
feature_importance = pd.DataFrame(
    pca.components_.T,
    columns=[f'PC{i+1}' for i in range(pca.n_components_)],
    index=df_scaled.columns
)

fig_importance = px.imshow(
    feature_importance,
    title="PCA Components Heatmap",
    labels=dict(x="Principal Components", y="Features", color="Coefficie
)
st.plotly_chart(fig_importance)

# Cluster Analysis
st.header("4. Cluster Analysis")

# Cluster characteristics
st.subheader("4.1 Cluster Characteristics")
```

```python
# Fungsi untuk load data
@st.cache_data
def load_data():
    df = pd.read_csv('country-data.csv')
    return df

# Fungsi preprocessing
def preprocess_data(df):
    # Copy dataframe
    df_clean = df.copy()

    # Handle missing values
    numeric_cols = df_clean.select_dtypes(include=['float64', 'int64']).colu
    for col in numeric_cols:
        # Replace missing values with median
        df_clean[col].fillna(df_clean[col].median(), inplace=True)

    # Handle outliers using IQR method
    def handle_outliers(data, column):
        Q1 = data[column].quantile(0.25)
        Q3 = data[column].quantile(0.75)
        IQR = Q3 - Q1
        lower_bound = Q1 - 1.5 * IQR
        upper_bound = Q3 + 1.5 * IQR
        data[column] = np.where(data[column] > upper_bound, upper_bound,
                                np.where(data[column] < lower_bound, lower_b
        return data

    for col in numeric_cols:
        if col != 'country':  # Skip non-numeric columns
            df_clean = handle_outliers(df_clean, col)

    return df_clean

# Fungsi untuk scaling data
def scale_features(df, scaler_type='standard'):
    # Separate features and country names
    features = df.select_dtypes(include=['float64', 'int64']).columns
    X = df[features]

    if scaler_type == 'standard':
        scaler = StandardScaler()
    else:
        scaler = RobustScaler()
```

```
        ))

    fig_radar.update_layout(
        polar=dict(radialaxis=dict(visible=True, range=[-2, 2])),
        showlegend=True,
        title="Cluster Characteristics Comparison"
    )
    st.plotly_chart(fig_radar)

    # Recommendations
    st.header("5. Recommendations")

    # Identify countries most in need
    priority_cluster = st.selectbox(
        "Select cluster to analyze",
        range(n_clusters)
    )

    if 'Kematian_anak' in df_clean.columns and 'Pendapatan' in df_clean.colu
        priority_countries = df_clean[df_clean['Cluster'] == priority_cluste
            ['Kematian_anak', 'Pendapatan'],
            ascending=[False, True]
        ).head(10)

        st.write("Top 10 Priority Countries:")
        st.write(priority_countries)
    else:
        st.error("Required columns 'Kematian_anak' and 'Pendapatan' are miss

    # Export results
    if st.button("Export Results"):
        output = io.BytesIO()
        with pd.ExcelWriter(output, engine='xlsxwriter') as writer:
            df_clean.to_excel(writer, sheet_name='Complete Data')
            priority_countries.to_excel(writer, sheet_name='Priority Countri

        st.download_button(
            label="Download Excel file",
            data=output.getvalue(),
            file_name="clustering_results.xlsx",
            mime="application/vnd.openxmlformats-officedocument.spreadsheetm
        )

if __name__ == "__main__":
    main()
```



	Negara	Kematian_anak	Ekspor	Kesehatan	Impor	Pendapatan	Inflasi	Harapan_hidup	Jumlah_fertiliti	GDPperkapita
47	Egypt	29.1	21.3	4.66	26.6	9,860	10.1	70.5	3.19	2,600
48	El Salvador	19.2	26.9	6.91	46.6	7,300	2.65	74.1	2.27	2,990
49	Equatorial Guinea	111	85.8	4.48	58.9	33,700	24.9	60.9	5.21	17,100
50	Eritrea	55.2	4.79	2.66	23.3	1,420	11.6	61.7	4.61	482
51	Estonia	4.5	75.1	6.03	68.7	22,700	1.74	76	1.72	14,600
52	Fiji	24.1	57.8	4.86	63.9	7,350	4.23	65.3	2.67	3,650
53	Finland	3	38.7	8.95	37.4	39,800	0.351	80	1.87	46,200
54	France	4.2	26.8	11.9	28.1	36,900	1.05	81.4	2.03	40,600
55	Gabon	63.7	57.7	3.5	18.9	15,400	16.6	62.9	4.08	8,750
56	Gambia	80.3	23.8	5.69	42.7	1,660	4.3	65.5	5.71	562

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 167 entries, 0 to 166
Data columns (total 10 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Negara           167 non-null    object
 1   Kematian_anak    167 non-null    float64
 2   Ekspor           167 non-null    float64
 3   Kesehatan        167 non-null    float64
 4   Impor            167 non-null    float64
 5   Pendapatan       167 non-null    int64
 6   Inflasi          167 non-null    float64
 7   Harapan_hidup    167 non-null    float64
 8   Jumlah_fertiliti 167 non-null    float64
 9   GDPperkapita     167 non-null    int64
dtypes: float64(7), int64(2), object(1)
```