

DAFTAR PUSTAKA

- [1] “Pusbangjak.” Accessed: Aug. 07, 2024. [Online]. Available: <https://pusbangjak.kemnaker.go.id/publication-details/presentasi-indonesia-employment-report->
- [2] “Trade and employment : country report for Indonesia,” p. 92, 2013, Accessed: Aug. 07, 2024. [Online]. Available: <https://www.ilo.org/publications/trade-and-employment-country-report-indonesia>
- [3] S. E. Hashemi, F. Gholian-Jouybari, and M. Hajiaghaei-Keshteli, “A fuzzy C-means algorithm for optimizing data clustering,” *Expert Syst Appl*, vol. 227, p. 120377, Oct. 2023, doi: 10.1016/J.ESWA.2023.120377.
- [4] S. Waschull, J. A. C. Bokhorst, J. C. Wortmann, and E. Molleman, “The redesign of blue- and white-collar work triggered by digitalization: collar matters,” *Comput Ind Eng*, vol. 165, p. 107910, Mar. 2022, doi: 10.1016/J.CIE.2021.107910.
- [5] I. Karakatsanis *et al.*, “Data mining approach to monitoring the requirements of the job market: A case study,” *Inf Syst*, vol. 65, pp. 1–6, Apr. 2017, doi: 10.1016/J.IS.2016.10.009.
- [6] I. A. Wowczko, “Skills and Vacancy Analysis with Data Mining Techniques,” *Informatics 2015, Vol. 2, Pages 31-49*, vol. 2, no. 4, pp. 31–49, Nov. 2015, doi: 10.3390/INFORMATICS2040031.
- [7] B. Walek and O. Pektor, “Data Mining of Job Requirements in Online Job Advertisements Using Machine Learning and SDCA Logistic Regression,” *Mathematics 2021, Vol. 9, Page 2475*, vol. 9, no. 19, p. 2475, Oct. 2021, doi: 10.3390/MATH9192475.
- [8] W. N. Dalilah *et al.*, “Analysing Online Vacancy and Skills Demand using Text Mining,” *J Phys Conf Ser*, vol. 1496, no. 1, p. 012011, Mar. 2020, doi: 10.1088/1742-6596/1496/1/012011.
- [9] J. Ha, M. Kambe, and J. Pe, “Data Mining: Concepts and Techniques,” *Data Mining: Concepts and Techniques*, pp. 1–703, Jan. 2011, doi: 10.1016/C2009-0-61819-5.
- [10] S. E. Hashemi, F. Gholian-Jouybari, and M. Hajiaghaei-Keshteli, “A fuzzy C-means algorithm for optimizing data clustering,” *Expert Syst Appl*, vol. 227, p. 120377, Oct. 2023, doi: 10.1016/J.ESWA.2023.120377.
- [11] S. Askari, “Fuzzy C-Means clustering algorithm for data with unequal cluster sizes and contaminated with noise and outliers: Review and development,” *Expert Syst Appl*, vol. 165, p. 113856, Mar. 2021, doi: 10.1016/J.ESWA.2020.113856.

LAMPIRAN KODE

```
import dash
from dash import dcc, html
import dash_bootstrap_components as dbc
import plotly.express as px
import plotly.graph_objects as go
import pandas as pd
import numpy as np
from dash.dependencies import Input, Output
from sklearn.preprocessing import StandardScaler
import skfuzzy as fuzz

df1 =
pd.read_csv(r'C:\Users\jkchristian\Downloads\2023\Jobdetails_2023Full.csv')
df2 =
pd.read_csv(r'C:\Users\jkchristian\Downloads\2023\Businessesprofile_2023Full.csv')

merged_df = pd.merge(df1, df2, on='employer_user_id', how='inner')
merged_df.drop_duplicates(inplace=True)
merged_df.fillna(0, inplace=True)

filtered_df =
merged_df[merged_df['jse_cluster_area_x'].str.contains('Greater Yogyakarta', case=False, na=False)]

demand_supply_analysis = filtered_df.groupby(['industry']).agg({
    'job_count': 'sum',
    'apply_count': 'sum'
}).reset_index()

demand_supply_analysis['demand_supply_ratio'] = (
    demand_supply_analysis['job_count'] /
    demand_supply_analysis['apply_count'].replace(0, np.nan)
)
demand_supply_analysis['demand_supply_ratio'].fillna(0,
inplace=True)

demand_supply_analysis['job_vacancy_rate'] = (
    (demand_supply_analysis['job_count'] -
    demand_supply_analysis['apply_count']) /
    demand_supply_analysis['job_count']
)
demand_supply_analysis['job_vacancy_rate'].fillna(0, inplace=True)
```

```

demand_supply_analysis['application_rate'] = (
    demand_supply_analysis['apply_count'] /
    demand_supply_analysis['job_count']
)
demand_supply_analysis['application_rate'].fillna(0, inplace=True)

role_analysis = filtered_df.groupby(['industry',
    'mapped_role']).agg({
    'job_count': 'sum',
    'apply_count': 'sum'
}).reset_index()

features = demand_supply_analysis[['job_count', 'apply_count',
    'demand_supply_ratio', 'job_vacancy_rate',
    'application_rate']].values

scaler = StandardScaler()
normalized_features = scaler.fit_transform(features)

c = 5
m = 2

cntr, u, _, _, jm, _, fpc = fuzz.cluster.cmeans(
    normalized_features.T, c=c, m=m, error=0.005, maxiter=1000,
    init=None, seed=42
)

demand_supply_analysis['cluster'] = np.argmax(u, axis=0)

print(f"Fuzzy Partition Coefficient (FPC): {fpc}")

app = dash.Dash(__name__,
    external_stylesheets=[dbc.themes.BOOTSTRAP])

app.layout = dbc.Container([
    dbc.Row([dbc.Col(html.H1("Supply and Demand Analysis of Semi-
    Skilled Workers in Yogyakarta"), className="mb-4")]),
    dcc.Interval(id="interval-component", interval=1*1000,
    n_intervals=0),
    dbc.Row([dbc.Col(dcc.Graph(id='demand-supply-ratio-chart'),
    width=12)]),
    dbc.Row([dbc.Col(dcc.Graph(id='job-vacancy-rate-chart'),
    width=12)]),
    dbc.Row([dbc.Col(dcc.Graph(id='application-rate-chart'),
    width=12)]),
    dbc.Row([dbc.Col(dcc.Graph(id='cluster-scatter-plot'),
    width=12)]),
    dbc.Row([dbc.Col(dcc.Graph(id='sse-plot'), width=12)]),

```

```

        dbc.Row([dbc.Col(dcc.Graph(id='role-breakdown-chart'),
width=12)]),
        dbc.Row([dbc.Col(dcc.Graph(id='membership-matrix-plot'),
width=12)]),
        dbc.Row([dbc.Col(dcc.Graph(id='cluster-centers-plot'),
width=12)])
    ])

@app.callback(
    [Output('demand-supply-ratio-chart', 'figure'),
    Output('job-vacancy-rate-chart', 'figure'),
    Output('application-rate-chart', 'figure'),
    Output('cluster-scatter-plot', 'figure'),
    Output('sse-plot', 'figure'),
    Output('role-breakdown-chart', 'figure'),
    Output('membership-matrix-plot', 'figure'),
    Output('cluster-centers-plot', 'figure')],
    [Input('interval-component', 'n_intervals')]
)
def update_charts(_):
    demand_supply_fig = go.Figure()

    demand_supply_fig.add_trace(go.Bar(
        x=demand_supply_analysis['industry'],
        y=demand_supply_analysis['job_count'],
        name='Job Count (Demand)',
        marker_color='blue'
    ))

    demand_supply_fig.add_trace(go.Bar(
        x=demand_supply_analysis['industry'],
        y=demand_supply_analysis['apply_count'],
        name='Apply Count (Supply)',
        marker_color='green'
    ))

    demand_supply_fig.add_trace(go.Scatter(
        x=demand_supply_analysis['industry'],
        y=demand_supply_analysis['demand_supply_ratio'],
        name='Demand-Supply Ratio',
        mode='lines+markers',
        yaxis='y2',
        line=dict(color='red', width=3)
    ))

    demand_supply_fig.update_layout(
        title='Job Demand vs. Supply by Industry',
        xaxis_title='Industry',
        yaxis=dict(title='Job Count / Apply Count'),

```

```

        yaxis2=dict(title='Demand-Supply Ratio', overlaying='y',
side='right', showgrid=False),
        barmode='group'
    )

    job_vacancy_fig = px.bar(demand_supply_analysis, x='industry',
y='job_vacancy_rate', title='Job Vacancy Rate by Industry',
labels={'job_vacancy_rate': 'Job Vacancy Rate (%)'},
color_discrete_sequence=['orange'])

    application_rate_fig = px.bar(demand_supply_analysis,
x='industry', y='application_rate', title='Application Rate by
Industry', labels={'application_rate': 'Application Rate (%)'},
color_discrete_sequence=['purple'])

    cluster_fig = px.scatter(demand_supply_analysis,
x='job_count', y='apply_count', color='cluster',
size='demand_supply_ratio', title='Cluster Analysis of Job Demand
vs. Supply', labels={'job_count': 'Job Count (Demand)',
'apply_count': 'Apply Count (Supply)'})

    sse_fig = px.line(x=range(len(jm)), y=jm, title='Sum of
Squared Errors (Fuzzy C-Means) Over Iterations', labels={'x':
'Iteration', 'y': 'SSE (Objective Function)'})

    role_chart = px.bar(role_analysis, x='mapped_role',
y='job_count', color='industry', title='Job Count by Role within
Each Industry', labels={'job_count': 'Job Count', 'mapped_role':
'Role'}, barmode='stack')

    membership_matrix_fig = go.Figure(data=go.Heatmap(
        z=u,
        x=[f"Cluster {i+1}" for i in range(c)],
        y=demand_supply_analysis['industry'],
        colorscale='Viridis'
    ))
    membership_matrix_fig.update_layout(
        title='Membership Matrix',
        xaxis_title='Clusters',
        yaxis_title='Industry'
    )

    cluster_centers_fig = go.Figure()
    for i in range(c):
        cluster_centers_fig.add_trace(go.Scatter(
            x=[f"Feature {j+1}" for j in
range(normalized_features.shape[1])],
            y=cntr[i],
            name=f"Cluster {i+1}"

```

```
    ))
    cluster_centers_fig.update_layout(
        title='Cluster Centers',
        xaxis_title='Features',
        yaxis_title='Normalized Value'
    )

    return demand_supply_fig, job_vacancy_fig,
    application_rate_fig, cluster_fig, sse_fig, role_chart,
    membership_matrix_fig, cluster_centers_fig

if __name__ == '__main__':
    app.run_server(debug=True)
```

