



GRAPH MINING IN AMAZON PRODUCT CO-PURCHASING

Muhammad Fauzan Jaisyurrahman - 2206814040

Muhammad Nabiel Subhan - 2206081553

Hanan Adipratama - 2206081824

Kevin Ignatius Wijaya - 2206083470

Iqza Ardiansyah - 2206810042



01

BUSINESS UNDERSTANDING

BUSINESS UNDERSTANDING

01 BUSINESS OBJECTIVE

- Mengidentifikasi grup produk yang paling **sering dibeli bersama** untuk meningkatkan efektivitas rekomendasi produk dalam e-commerce. Dengan begitu, sistem rekomendasi dapat lebih akurat dalam menampilkan produk yang relevan, sehingga **meningkatkan konversi penjualan dan kepuasan pelanggan**.

02 ASSESS SITUATION

- Perusahaan e-commerce **masih mengandalkan analisis tabular** yang terbatas pada atribut produk dan pelanggan, tanpa mempertimbangkan hubungan antar produk. Ini dapat diatasi dengan **analisis berbasis graf**.

BUSINESS UNDERSTANDING

03 DATA MINING GOALS

- Ditemukan pola co-purchasing produk yang paling sering terjadi, yaitu **produk mana yang sering muncul bersama dalam transaksi pelanggan**.

04 PROJECT PLAN

- Kami menggunakan **Frequent Subgraph Mining (FSM)** dengan **Pattern Growth** seperti **GRAMI** untuk mengidentifikasi pola keterhubungan dalam graf co-purchasing. FSM lebih efisien dibandingkan Apriori dan FP-Growth untuk dataset besar, cocok untuk menemukan pola berulang, dan dapat digunakan dalam analisis aturan asosiasi.
- Namun, tantangannya adalah kebutuhan komputasi yang tinggi serta tidak mempertimbangkan bobot hubungan antar produk seperti GNN.

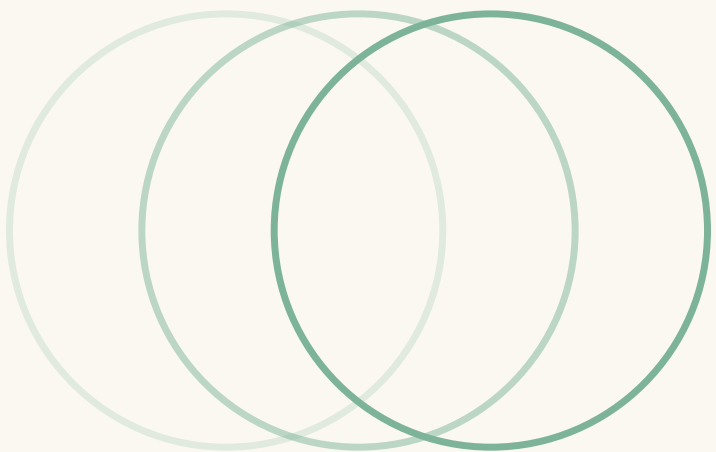


02 DATA UNDERSTANDING

EDA

- 01 MEMAHAMI STRUKTUR DATA GRAF**
- 02 MENGEKSPLORASI JARINGAN KETERHUBUNGAN PRODUK**
- 03 ANALISIS KOMUNITAS DAN STRUKTUR GRAF**

DATA UNDERSTANDING



- Terdapat 2 jenis dataset yang kelompok kami gunakan pada studi kasus kali ini,

COPURCHASE.CSV

- Berisi edge list (source, target) copurcase pada marketplace yang dikaji pada proyek ini.

	Source	Target
0	1	2
1	1	4
2	1	5
3	1	15
4	2	11

Tambang BTC

PRODUCTS.CSV

- Data ini berisi detail mengenai product yang ada pada data copurchase

	id	title	group	salesrank	review_cnt	downloads	rating
0	1	Patterns of Preaching: A Sermon Sampler	Book	396585.0	2	2	5.0
1	2	Candlemas: Feast of Flames	Book	168596.0	12	12	4.5
2	3	World War II Allied Fighter Planes Trading Cards	Book	1270652.0	1	1	5.0
3	4	Life Application Bible Commentary: 1 and 2 Tim...	Book	631289.0	1	1	4.0
4	5	Prayers That Avail Much for Business: Executive	Book	455160.0	0	0	0.0

MEMAHAMI STRUKTUR DATA GRAF

a. Cek ukuran dataset

```
[40] print("Shape of df_copurchase:", df_copurchase.shape)
      print("Shape of df_products:", df_products.shape)
```

```
➡ Shape of df_copurchase: (1234870, 2)
   Shape of df_products: (259167, 7)
```

b. Tampilkan tipe data dan nilai unik

```
▶ print(df_copurchase.info())

print(df_products.info())
for col in df_products.select_dtypes(include=['object', 'category']).columns:
    print(f'\nColumn: {col}')
    print(f'Unique values: {df_products[col].unique()}')
```

```
➡ <class 'pandas.core.frame.DataFrame'>
RangeIndex: 1234870 entries, 0 to 1234869
Data columns (total 2 columns):
 #   Column  Non-Null Count  Dtype
---  ---
 0   Source  1234870 non-null int64
 1   Target  1234870 non-null int64
dtypes: int64(2)
memory usage: 18.8 MB
None

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 259167 entries, 0 to 259166
Data columns (total 7 columns):
 #   Column  Non-Null Count  Dtype
---  ---
 0   id      259167 non-null int64
 1   title   259167 non-null object
 2   group   259167 non-null object
 3   salesrank 259167 non-null float64
 4   review_cnt 259167 non-null int64
 5   downloads 259167 non-null int64
 6   rating   259167 non-null float64
dtypes: float64(2), int64(3), object(2)
memory usage: 13.8+ MB
None

Column: title
Unique values: ['Patterns of Preaching: A Sermon Sampler' 'Candlemas: Feast of Flames'
'World War II Allied Fighter Planes Trading Cards' ... 'Halloween II'
'Book Of Vision Quest'
"Favorite Russian Fairy Tales (Dover Children's Thrift Classics)"]

Column: group
Unique values: ['Book' 'Music' 'DVD' 'Video' 'Toy' 'Video Games' 'Software'
'Baby Product' 'CE']
```


JARINGAN KETERHUBUNGAN

519677
NODES

1234870
EDGES

```
Number of unique products (nodes): 519677  
Number of co-purchasing relations (edges): 1234870
```

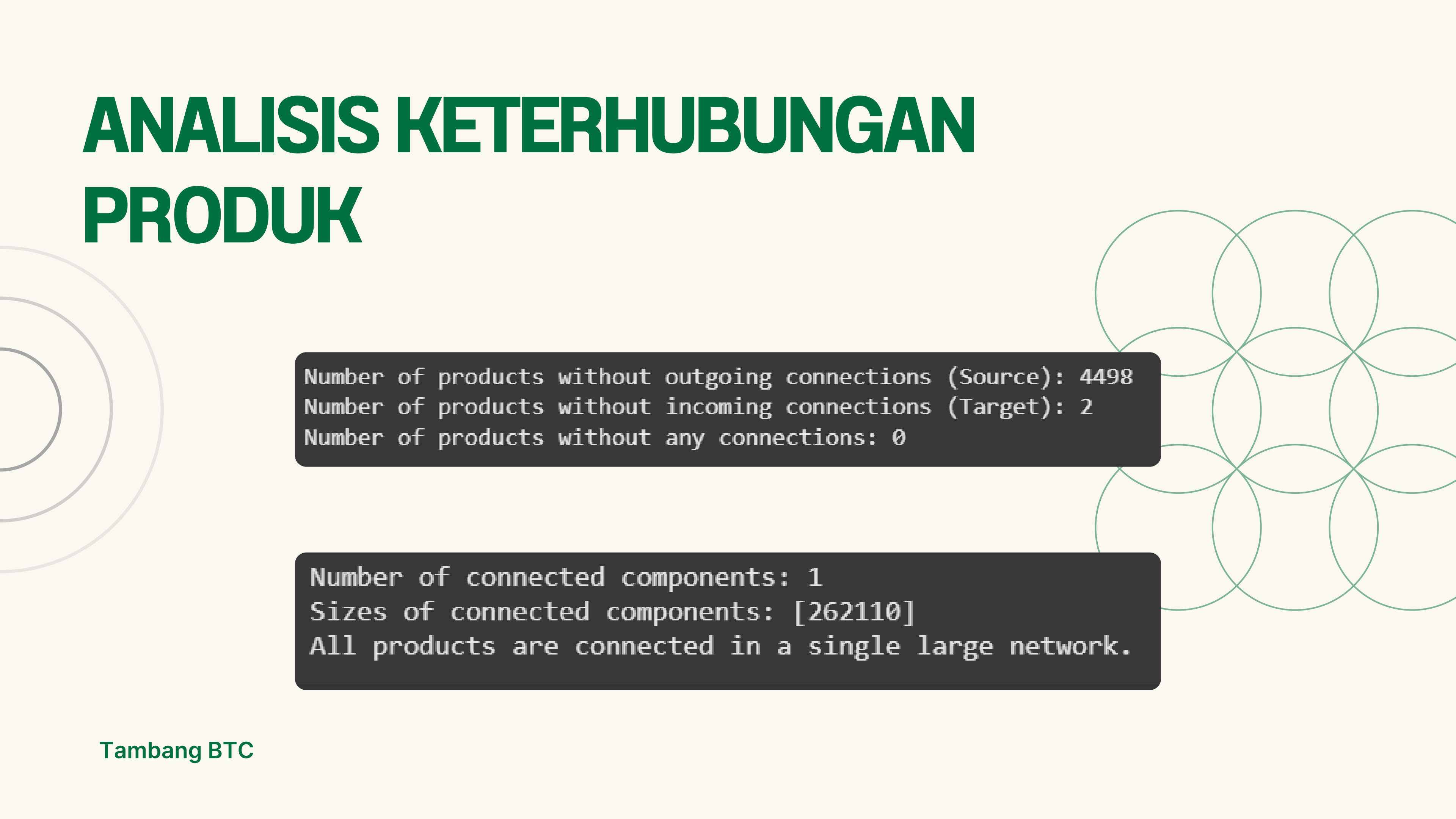
PRODUK DENGAN CO-PURCHASING TERBANYAK

Top 10 Source Products with most co-purchases:		
	count	
Source		
131454	5	
166501	5	
166487	5	
166488	5	
166489	5	
166490	5	
166491	5	
166492	5	
166493	5	
166494	5	

Top 10 Target Products with most co-purchases:		
	count	
Target		
14949	420	
4429	404	
33	361	
10519	334	
12771	330	
8	293	
297	280	
481	275	
5737	272	
9106	227	

Tambang BTC

ANALISIS KETERHUBUNGAN PRODUK

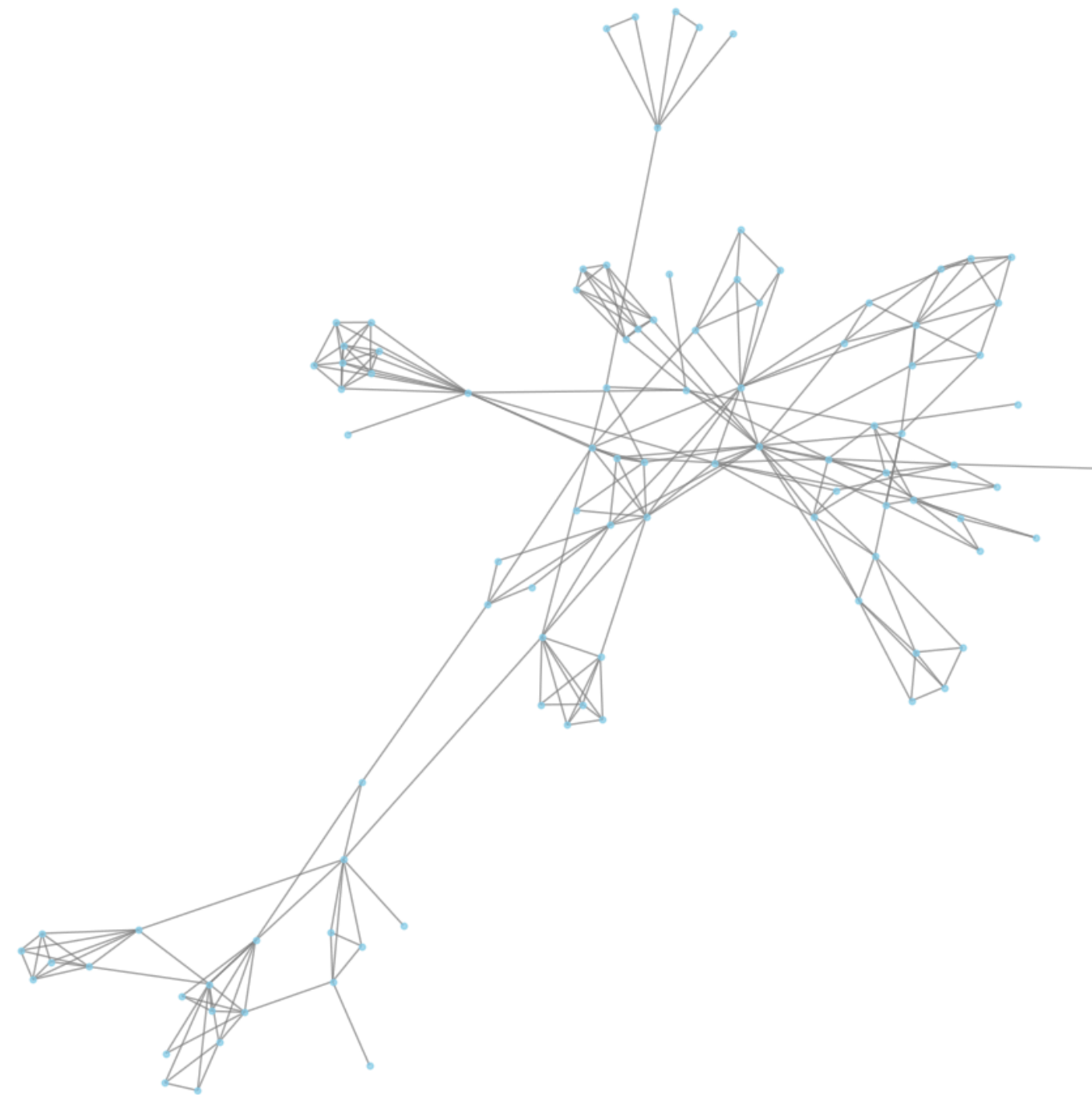


```
Number of products without outgoing connections (Source): 4498  
Number of products without incoming connections (Target): 2  
Number of products without any connections: 0
```

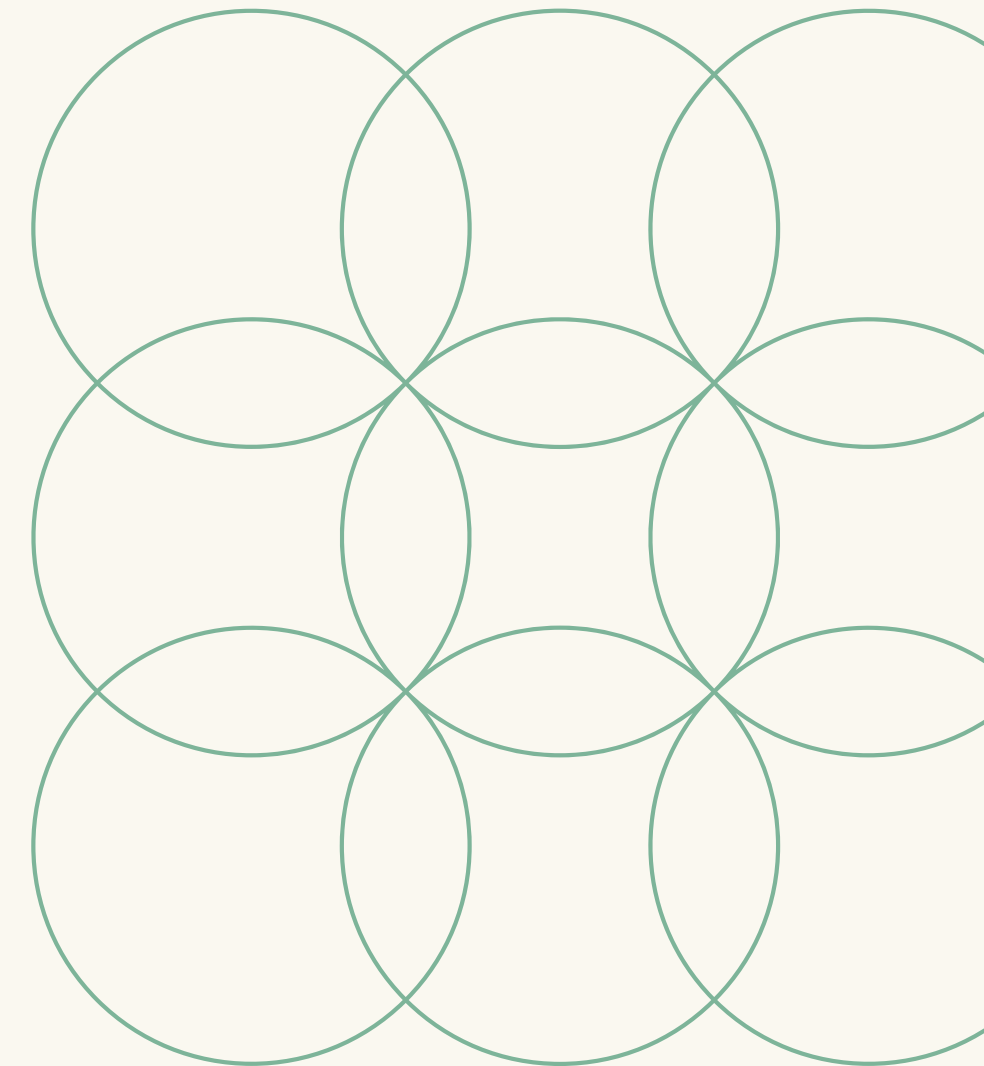
```
Number of connected components: 1  
Sizes of connected components: [262110]  
All products are connected in a single large network.
```


VISUALISASI GRAF KECIL

Visualization of a Small Connected Component



Tambang BTC





03

DATA

PREPARATION

DATA PREPARATION

- 01 CEK KEBERADAAN DATA DUPLIKAT.**
- 02 CEK KEBERADAAN DATA KOSONG/NULL PADA KEDUA FILE.**
- 03 GABUNGGAN COPURCHASE.CSV DENGAN PRODUCTS.CSV**
- 04 NORMALISASI DATA HARGA DAN RATING**
- 05 FEATURE ENGINEERING**

CEK KEBERADAAN DATA DUPLIKAT

```
▶ duplicates = df_copurchase[df_copurchase.duplicated(subset=['Source', 'Target'], keep=False)]

if not duplicates.empty:
    print("Duplicate entries found:")
    print(duplicates)

    # Aggregate duplicate entries
    df_copurchase = df_copurchase.groupby(['Source', 'Target']).size().reset_index(name='Frequency')
    print("\nAggregated DataFrame:")
    print(df_copurchase)
else:
    print("No duplicate entries found.")
```

➞ No duplicate entries found.

Tambang BTC

GABUNGGKAN COPURCHASE.CSV DENGAN PRODUCTS.CSV

	id	title	group	salesrank	review_cnt	downloads	rating	Source	Target
0	1.0	Patterns of Preaching: A Sermon Sampler	Book	396585.0	2.0	2.0	5.0	1	2
1	1.0	Patterns of Preaching: A Sermon Sampler	Book	396585.0	2.0	2.0	5.0	1	4
2	1.0	Patterns of Preaching: A Sermon Sampler	Book	396585.0	2.0	2.0	5.0	1	5
3	1.0	Patterns of Preaching: A Sermon Sampler	Book	396585.0	2.0	2.0	5.0	1	15
4	2.0	Candlemas: Feast of Flames	Book	168596.0	12.0	12.0	4.5	2	11

Tambang BTC

NORMALISASI DATA HARGA DAN RATING

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()

df_merged[['salesrank', 'rating']] = scaler.fit_transform(df_merged[['salesrank', 'rating']])

df_merged.head()
```

	id	title	group	salesrank	review_cnt	downloads	rating	Source	Target
0	1.0	Patterns of Preaching: A Sermon Sampler	Book	0.104549	2.0	2.0	1.0	1	2
1	1.0	Patterns of Preaching: A Sermon Sampler	Book	0.104549	2.0	2.0	1.0	1	4
2	1.0	Patterns of Preaching: A Sermon Sampler	Book	0.104549	2.0	2.0	1.0	1	5
3	1.0	Patterns of Preaching: A Sermon Sampler	Book	0.104549	2.0	2.0	1.0	1	15
4	2.0	Candlemas: Feast of Flames	Book	0.044446	12.0	12.0	0.9	2	11

Tambang BTC

CEK KEBERADAAN DATA KOSONG/NULL PADA KEDUA FILE

```
[ ] df_copurchase.isnull().sum()
```

	0
Source	0
Target	0

dtype: int64

Tambang BTC

```
[ ] df_products.isnull().sum()
```

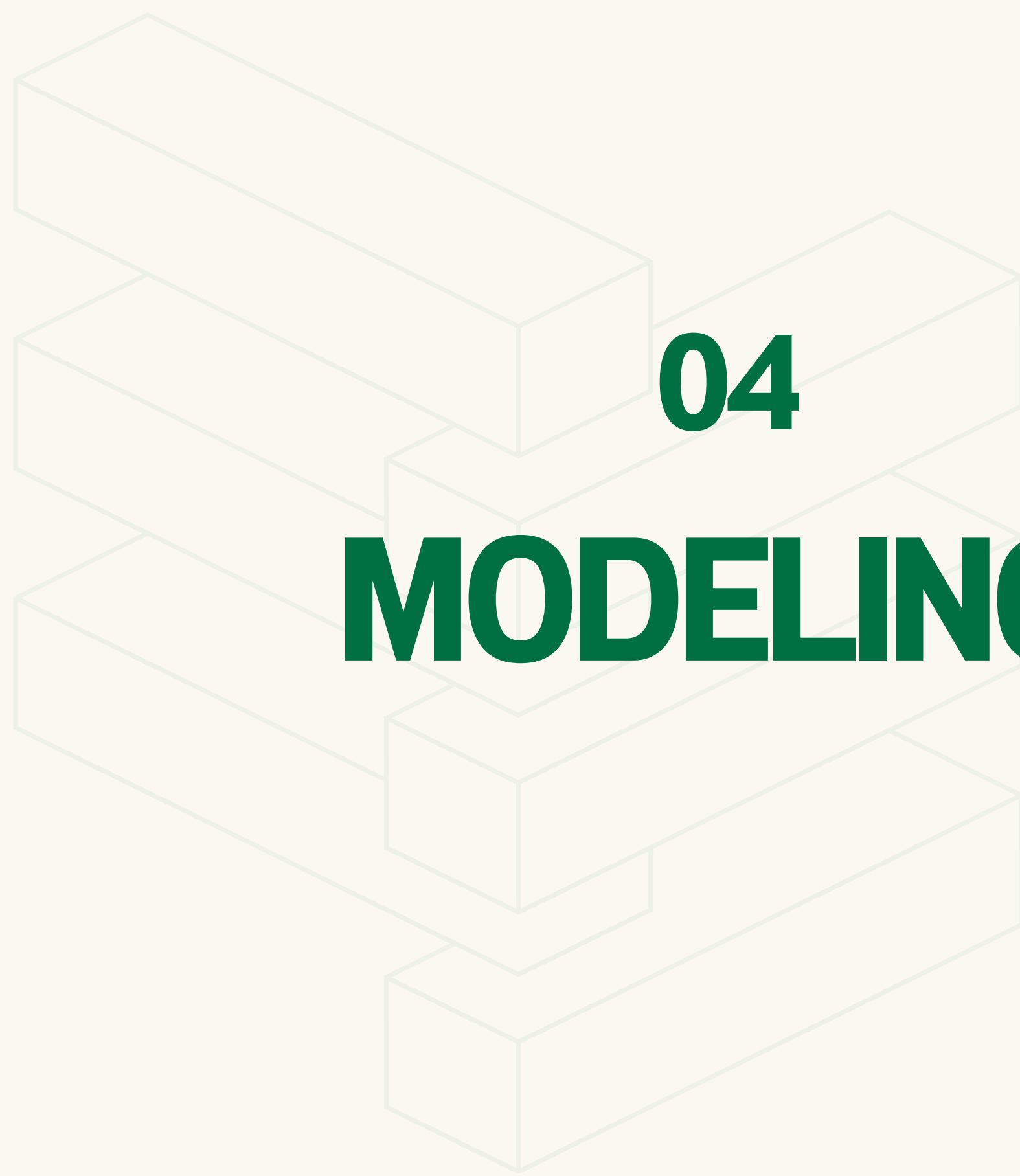
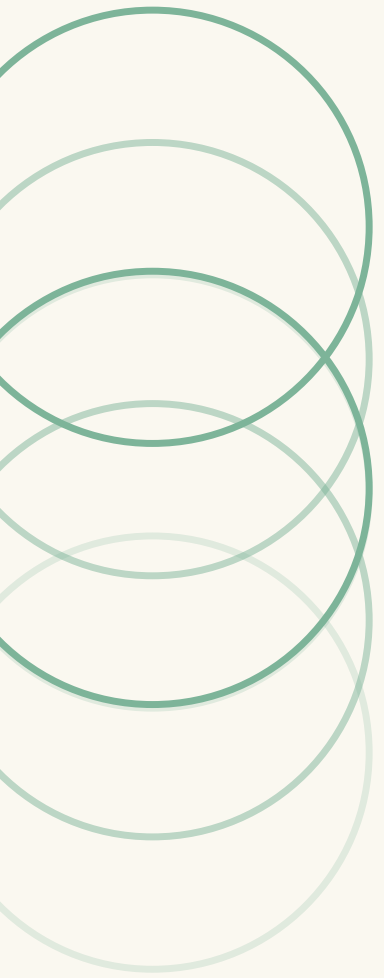
	0
id	0
title	0
group	0
salesrank	0
review_cnt	0
downloads	0
rating	0

FEATURE ENGINEERING

```
source_counts = copurchase_df['Source'].value_counts().to_dict()
target_counts = copurchase_df['Target'].value_counts().to_dict()

# Add count information to products_df
if 'id' in products_df.columns:
    key_col = 'id'
elif 'product_id' in products_df.columns:
    key_col = 'product_id'

products_df['outgoing_count'] = products_df[key_col].astype(str).map(source_counts).fillna(0)
products_df['incoming_count'] = products_df[key_col].astype(str).map(target_counts).fillna(0)
products_df['total_connections'] = products_df['outgoing_count'] + products_df['incoming_count']
```



04

MODELING

Tambang BTC

SAMPLING STRATEGY

01 CLUSTERING COEFFICIENT CALCULATION

- Untuk keperluan *clustering coefficient*, dengan network yang memiliki lebih dari 10.000 nodes akan disampling 1.000 secara acak. Estimasi cukup akurat secara statistik dan dapat mengurangi waktu komputasi.

02 CENTRALITY MEASURES

- Untuk *centrality measures*, kita menggunakan *degree centrality* sebagai pengganti dari metrik lain yang secara komputasi lebih mahal. *Degree centrality* juga efektif dalam mengidentifikasi node penting dalam jaringan yang besar.

03 VISUALIZATION

- Untuk keperluan visualisasi, kami akan membatasi untuk *100 top nodes* dengan *degree centrality* tertinggi daripada mengambil sample secara acak. Dengan begitu, visualisasi tetap bisa fokus terhadap bagian yang relevan.

04 RECOMMENDATION EVALUATION

- Untuk evaluasi rekomendasi, kami akan *sampling* acak 50 produk karena cukup efisien dan cukup representatif untuk menilai performa sistem rekomendasi.

MODEL USED (COMMUNITY DETECTION)

LOUVAIN

- Algoritma Louvain adalah metode community detection berbasis optimisasi modularity yang mengelompokkan node dalam graf ke dalam komunitas dengan tujuan memaksimalkan modularity. Algoritma louvain kami pilih, karena implementasinya yang mudah, cepat, dan scalable untuk tujuan kami mencari FSM dari graf terarah.

METRIC EVALUATION

01 PRECISION

- What it measures: Percentage of recommended products that are actually relevant
- Why it matters: Ensures recommendations are accurate and relevant to users

02 RECALL

- What it measures: Percentage of all relevant products that were successfully recommended
- Why it matters: Ensures the system isn't missing important connections

03 F1 SCORE

- What it measures: Harmonic mean of precision and recall
- Why it matters: Balances the trade-off between precision and recall in a single metric

04 HOW IT WORKS

- Kode mengambil sampel random dari produk-produk (node dalam graf) untuk dievaluasi.
- Sampel kemudian di evaluasi diperoleh dari koneksi yang sudah ada dalam graf (tetangga dari produk yang dievaluasi).



05

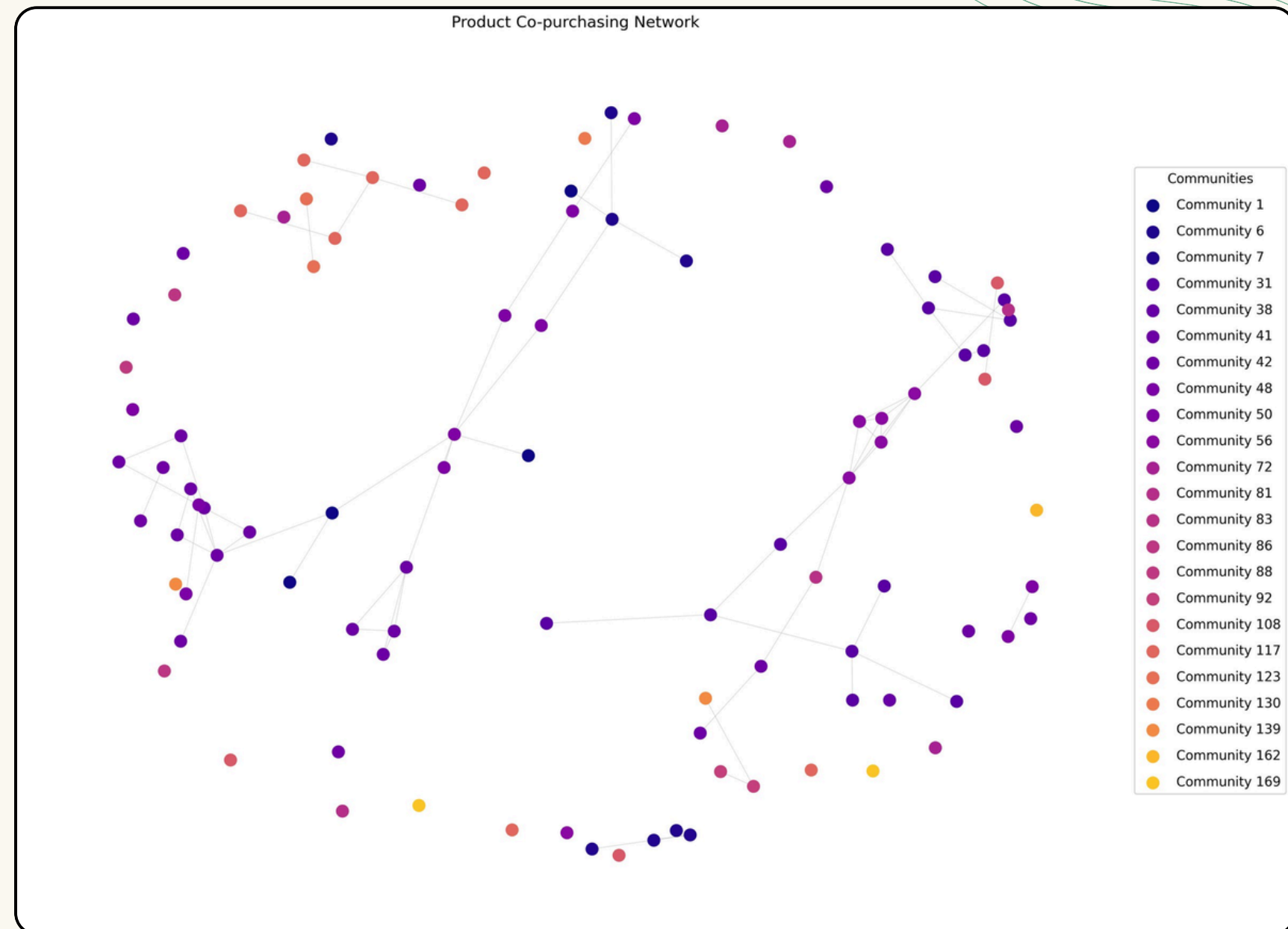
EVALUATION

Tambang BTC

VISUALIZATION (COMMUNITY DETECTION)

01 LOUVAIN

- Algoritma Louvain berhasil mengidentifikasi 211 komunitas produk, dengan 5 komunitas terbesar berisi 9.600-15.280 produk.



METRIC EVALUATION

01 DASAR

- Rekomendasi dasar dalam sistem co-purchase menggunakan fungsi `recommend_products_enhanced` tanpa parameter komunitas, yang hanya mengandalkan struktur koneksi langsung dalam graf

02 BERBASIS KOMUNITAS

- Rekomendasi berbasis komunitas menggunakan fungsi yang sama namun dengan parameter tambahan `partition` dan `community_weight=0.5`, yang memberikan bobot tambahan untuk produk dalam komunitas yang sama berdasarkan hasil algoritma deteksi komunitas.

03 LANJUT

- Rekomendasi lanjutan menggunakan fungsi `recommend_products_advanced` yang mempertimbangkan lebih banyak faktor seperti popularitas produk, rating, perhitungan koneksi yang lebih kompleks, dan panjang jalur terpendek antar node.

METRIC SCORE

01 PRECISION

- Rekomendasi Dasar: 0.9680
- Rekomendasi Berbasis Komunitas: 0.968
- Rekomendasi Lanjut: 0.860

03 F1 SCORE

- Rekomendasi Dasar: 0.780
- Rekomendasi Berbasis Komunitas: 0.780
- Rekomendasi Lanjut: 0.704

02 RECALL

- Rekomendasi Dasar: 0.864
- Rekomendasi Berbasis Komunitas: 0.864
- Rekomendasi Lanjut: 0.774

Rekomendasi dasar dan berbasis komunitas menunjukkan performa yang identik dan lebih baik dari pendekatan lanjutan, dengan precision yang sangat tinggi.

Rekomendasi lanjut justru menunjukkan performa yang lebih rendah mengindikasikan bahwa penambahan terlalu banyak faktor mungkin memperkenalkan noise dan pendekatan yang lebih sederhana kadang lebih efektif untuk rekomendasi produk dalam konteks e-commerce.

Tambang BTC

HASIL REKOMENDASI (COMMUNITY DETECTION)

01 LOUVAIN

- Selain menampilkan graf, kami juga mencari tahu, top 10 product apa saja yang memiliki nilai degree centrality tertinggi untuk dievaluasi product mana saja yang memiliki recommendation score yang tertinggi dengan product yang berkaitan

[RISTEK.LINK/REKOMENDASIPRODUCTCP2](https://ristek.link/rekomendasi-product-cp2)

Tambang BTC

RESULT

01 LIMITATION

- Pada implementasi kami saat ini, kami terbatas oleh kekuatan komputasi dan waktu sehingga kami sering melakukan *sampling* yang menyebabkan hasil mungkin tidak terlalu representatif.

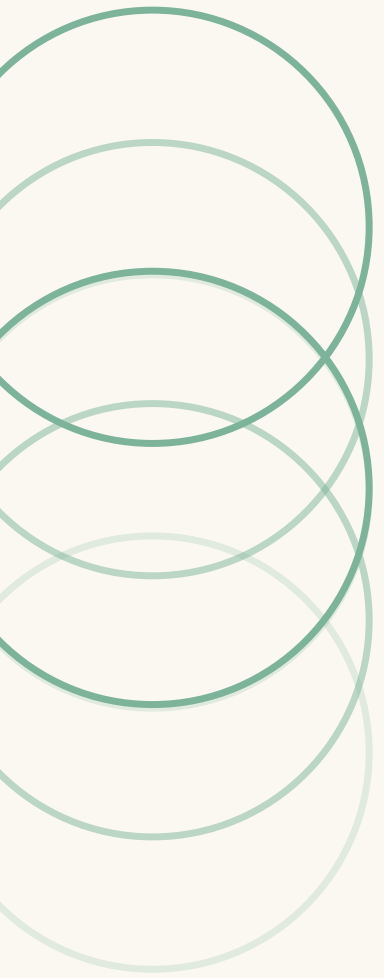
02 SUGGESTION

- Peningkatan ukuran *sample* atau tidak melakukan *sampling* sama sekali dan penggunaan *Leiden Algorithm* untuk meningkatkan hasil Louvain.

RESULT

03 DEPLOYMENT PLAN

- **Integrasi Data** – Menghubungkan data co-purchase dari e-commerce ke database graf (Neo4j) untuk pemrosesan real-time.
- **Pengembangan Microservice** – Membangun layanan rekomendasi dengan algoritma Louvain untuk deteksi komunitas, diperbarui setiap minggu.
- **Implementasi Bertahap**
 - Mulai dari halaman detail produk dengan 5 rekomendasi teratas.
 - Ekspansi ke email notifikasi dan rekomendasi personalisasi.
- **Monitoring & Evaluasi** – Metrik teknis: precision, recall, coverage.
- **Skalabilitas** – Load balancer & caching untuk menangani lonjakan traffic saat high-demand (misal, musim belanja akhir tahun).



TERIMA KASIH

Tambang BTC

