Rakamin Academy X KALBE Nutritionals

Stock Prediction & Customer Segmentation

Kalbe Nutritionals Data Scientist Project Based Internship Program

Presented by
Ridhwan Muttaqien

Ridhwan Muttaqien

About Me

Data Analyst / Data Scientist who has completed Data Science Bootcamp with a background as a former Safety officer. Educational background is Telecommunication Engineering.

Rakamin Academy

Experiences

Data Science Trainee

Joining Full time Data Science Bootcamp
May - August 2023

Safety Officer

PT. Swahusada Guna Instrumentasi
Jan - April 2023
August 2014 - September 2019

Stock Prediction

Objective :

To estimate the quantity of products sold so that the inventory team can create daily inventory sufficient

Data Loading

Transaction Dataframe

```
df_trx = pd.read_csv('Transaction.csv', delimiter=';')
df_trx
```

[20] ✓ 0.0s

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID
0	TR11369	328	01/01/2022	P3	7500	4	30000	12
1	TR16356	165	01/01/2022	P9	10000	7	70000	1
2	TR1984	183	01/01/2022	P1	8800	4	35200	4
3	TR35256	160	01/01/2022	P1	8800	7	61600	4
4	TR41231	386	01/01/2022	P9	10000	1	10000	4
...	...	...	...	...	...	...	...	...
5015	TR54423	243	31/12/2022	P10	15000	5	75000	3
5016	TR5604	271	31/12/2022	P2	3200	4	12800	9
5017	TR81224	52	31/12/2022	P7	9400	6	56400	9
5018	TR85016	18	31/12/2022	P8	16000	3	48000	13
5019	TR85684	55	31/12/2022	P8	16000	1	16000	6

5020 rows × 8 columns

- Data yang digunakan adalah tabel transaksi
- Transaksi terjadi pada tahun 2022 (1 Januari - 31 Desember)
- Terdapat 5020 transaksi dari 447 Customer

Data Preprocessing

```
df_trx.info()
```
✓ 0.0s

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5020 entries, 0 to 5019
Data columns (total 8 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   TransactionID  5020 non-null   object
 1   CustomerID     5020 non-null   int64 
 2   Date           5020 non-null   object
 3   ProductID      5020 non-null   object
 4   Price          5020 non-null   int64 
 5   Qty            5020 non-null   int64 
 6   TotalAmount    5020 non-null   int64 
 7   StoreID        5020 non-null   int64 
dtypes: int64(5), object(3)
memory usage: 313.9+ KB
```

```
- CustomerID and StoreID are in wrong data type. They're should be object instead of int64, because they're unique number for every customer and store
- Date is in wrong data type. It's should be in datetime instead of object
```

```
# changing data type

df_trx['Date'] = pd.to_datetime(df_trx['Date'])
df_trx['CustomerID'] = df_trx['CustomerID'].astype('object')
df_trx['StoreID'] = df_trx['StoreID'].astype('object')
```
✓ 0.0s

- Terdapat tipe data yang tidak sesuai yaitu StoreID, Date, dan CustomerID
- Mengubah tipe data CustomerID dan StoreID menjadi object
- Mengubah data tanggal menjadi datetime

Data Preprocessing

```
# checking missing value

df_trx.isnull().sum()
```

✓ 0.0s

```
TransactionID    0
CustomerID       0
Date             0
ProductID        0
Price            0
Qty              0
TotalAmount      0
StoreID          0
dtype: int64


No missing value in transaction dataframe
```

```
# checking duplicated data

df_trx.duplicated().sum()
```

✓ 0.0s

```
0


No duplicated data in transaction dataframe
```

- Tidak terdapat missing value pada dataset
- Tidak terdapat data terduplikasi pada dataset

Data Preprocessing

Rakamin Academy

```
# grouping transaction dataframe based on sum of quantity sold product

df_ts = df_trx.groupby('Date', as_index=False)['Qty'].sum()
df_ts
```

✓ 0.0s

	Date	Qty
0	2022-01-01	49
1	2022-01-02	50
2	2022-01-03	76
3	2022-01-04	98
4	2022-01-05	67
...	...	...
360	2022-12-27	70
361	2022-12-28	68
362	2022-12-29	42
363	2022-12-30	44
364	2022-12-31	37

365 rows × 2 columns

```
# making date column as index for time series dataframe

df_ts.set_index('Date', inplace=True)
df_ts
```

✓ 0.0s

	Qty
Date	
2022-01-01	49
2022-01-02	50
2022-01-03	76
2022-01-04	98
2022-01-05	67
...	...
2022-12-27	70
2022-12-28	68
2022-12-29	42
2022-12-30	44
2022-12-31	37

365 rows × 1 columns

Data Splitting

```
# checking index for splitting data

train_size = 0.8*len(df_ts)
train_size
```

✓ 0.0s

```
292.0
```

```
Data is splitted into train and test set with composition 80% train set and 20% test set
```

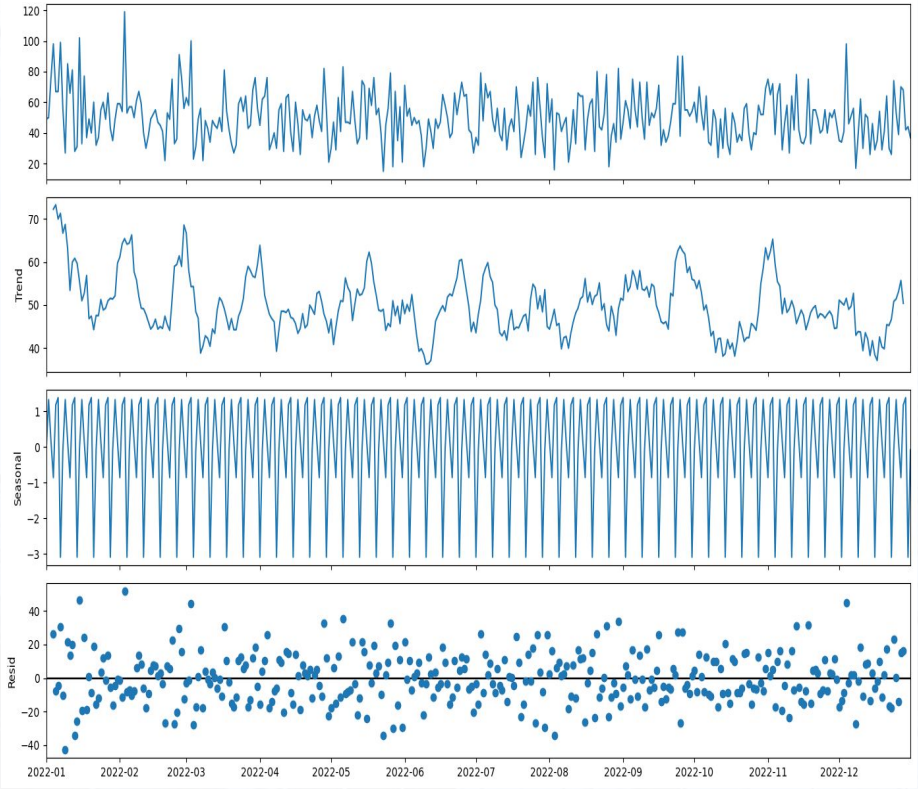
```
# splitting data

train = df_ts[:292]
test = df_ts[292:]

print('Train size : ', train.shape)
print('Test size : ', test.shape)
```

✓ 0.0s

```
Train size :  (292, 1)
Test size :  (73, 1)
```

- Membagi data menjadi train set dan test set dengan pembagian 80:20 (dalam persen)

Time Series Decomposition



Find Value for p,d,and q for ARIMA Model

Rakamin Academy

Checking Stationarity

```
# creating a function to check stationarity

def check_stationarity(series):
    # Copied from https://machinelearningmastery.com/time-series-data-stationary-python/

    result = adfuller(series.values)

    print('ADF Statistic: %f' % result[0])
    print('p-value: %f' % result[1])
    print('Critical Values:')
    for key, value in result[4].items():
        print('\t%s: %.3f' % (key, value))

    if (result[1] <= 0.05) & (result[4]['5%'] > result[0]):
        print("\u001b[32mStationary\u001b[0m")
    else:
        print("\x1b[31mNon-stationary\x1b[0m")
```

✓ 0.0s

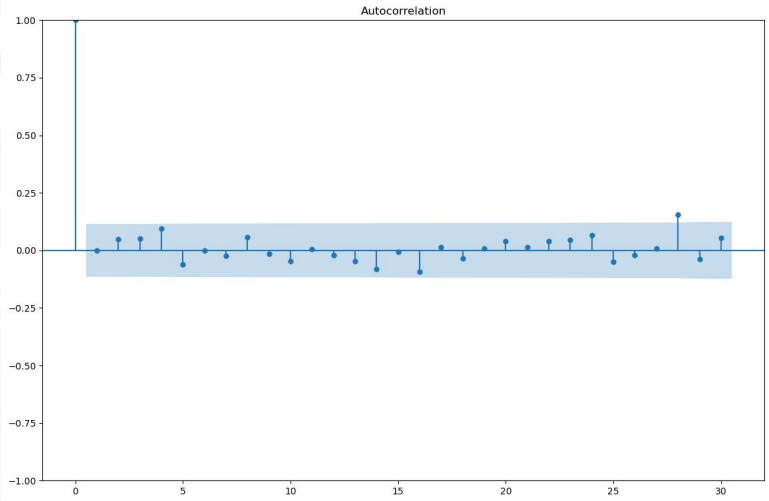
```
# checking stationarity from train set

check_stationarity(train['Qty'])
```

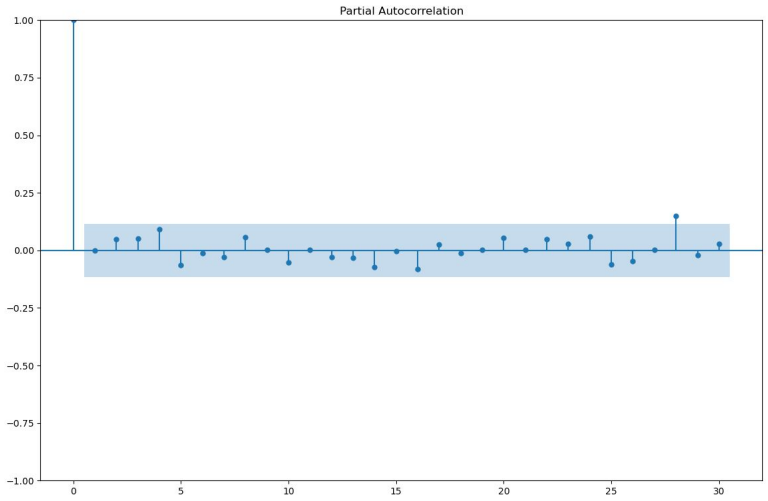
✓ 0.0s

```
ADF Statistic: -16.986059
p-value: 0.000000
Critical Values:
        1%: -3.453
        5%: -2.872
        10%: -2.572
Stationary
```

- Uji stasioneritas menunjukkan data yang kita gunakan sudah stasioner jadi tidak perlu dilakukan differencing
- Karena data sudah stasioner, maka nilai d untuk ARIMA model adalah 0

Find Value for p,d,and q for ARIMA Model





- PACF dan ACF cut off pada lag ke-28, maka nilai p dan q pada ARIMA Model adalah 28

Modelling & Evaluation

Rakamin Academy

Modelling & Predicting

```
# training the model based on order that we get from previous cell

order = (28, 0, 28)
model = ARIMA(train, order=order)
model_fit = model.fit()
```

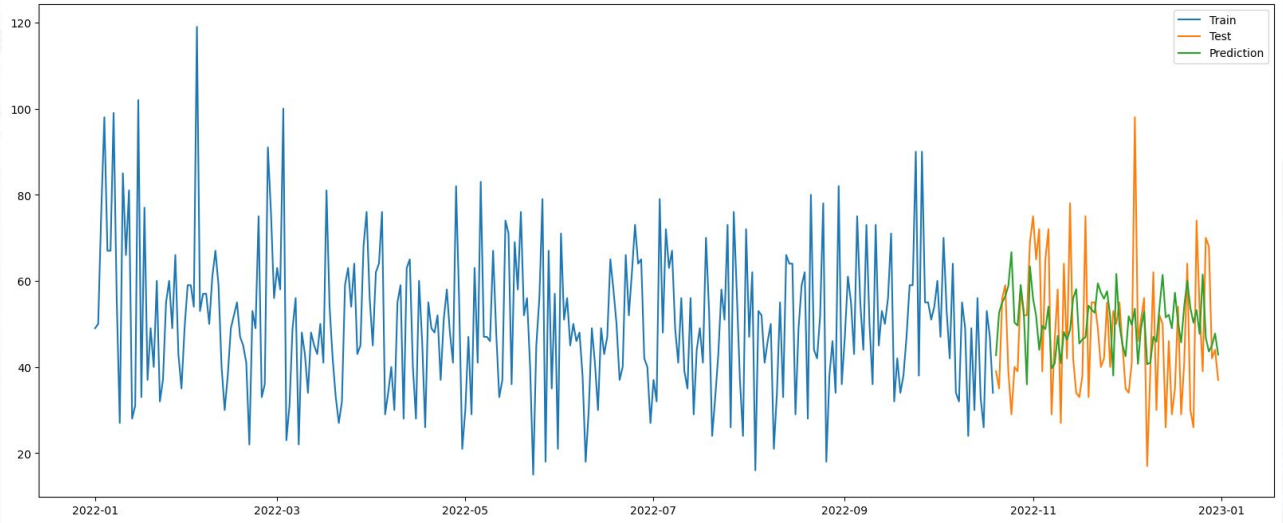
✓ 13.1s

```
# predicting

predictions = model_fit.forecast(steps=len(test))
```

✓ 0.0s

Modelling & Evaluation

Rakamin Academy



Model Evaluation

```
# calculating evaluation metrics for regression

from sklearn.metrics import mean_squared_error, mean_absolute_error
from math import sqrt
rmse = sqrt(mean_squared_error(test, predictions))
mae = mean_absolute_error(test, predictions)

print("mae :",mae)
print("rmse :",rmse)
```

0.2s

```
mae : 13.000288880955754
rmse : 15.984890791244794
```

Customer Segmentation

Objective :

This customer segment will later be used by marketing team to provide personalized promotion and sales treatment

Joining Dataset

```
df_clus = df_trx.merge(df_cust, on='CustomerID', how='left') # joining transaction and customer dataframe
df_clus
```

[38] ✓ 0.0s

	TransactionID	CustomerID	Date	ProductID	Price	Qty	TotalAmount	StoreID	Age	Gender	Marital Status	Income
0	TR11369	328	2022-01-01	P3	7500	4	30000	12	36.0	0.0	Married	10.53
1	TR16356	165	2022-01-01	P9	10000	7	70000	1	44.0	1.0	Married	14.58
2	TR1984	183	2022-01-01	P1	8800	4	35200	4	27.0	1.0	Single	0.18
3	TR35256	160	2022-01-01	P1	8800	7	61600	4	48.0	1.0	Married	12.57
4	TR41231	386	2022-01-01	P9	10000	1	10000	4	33.0	0.0	Married	6.95
...	...	...	...	...	...	...	...	...	...	...	...	...
5015	TR54423	243	2022-12-31	P10	15000	5	75000	3	38.0	0.0	Married	3.34
5016	TR5604	271	2022-12-31	P2	3200	4	12800	9	29.0	0.0	Married	4.74
5017	TR81224	52	2022-12-31	P7	9400	6	56400	9	37.0	0.0	Married	3.73
5018	TR85016	18	2022-12-31	P8	16000	3	48000	13	47.0	0.0	Married	13.60
5019	TR85684	55	2022-12-31	P8	16000	1	16000	6	34.0	1.0	Married	8.44

5020 rows × 12 columns

```
df_clus.isnull().sum() # checking missing value
```

[39] ✓ 0.0s

```
TransactionID     0
CustomerID        0
Date              0
ProductID         0
Price             0
Qty               0
TotalAmount       0
StoreID           0
Age               9
Gender            9
Marital Status    9
Income            9
dtype: int64
```

- Menggabungkan data transaksi dan data customer
- Terdapat 9 missing value. Missing value akan ditangani dengan menghapus semua baris yang terdapat missing value

Grouping dataset based on customerID

```
aggregation = {
    'TransactionID': 'count',
    'Qty': 'sum',
    'TotalAmount': 'sum',
    'Age': 'first',
    'Income': 'first'}

cluster = df_clus.groupby('CustomerID', as_index=False).agg(aggregation) #making new dataframe by grouping dataframe based on list of aggregate
cluster
```

✓ 0.0s

	CustomerID	TransactionID	Qty	TotalAmount	Age	Income
0	1	17	60	623300	55.0	5.12
1	2	13	57	392300	60.0	6.23
2	3	15	56	446200	32.0	9.17
3	4	10	46	302500	31.0	4.87
4	5	7	27	268600	58.0	3.57
...	...	...	...	...	...	...
441	443	16	59	485100	33.0	9.28
442	444	18	62	577700	53.0	15.31
443	445	18	68	587200	51.0	14.48
444	446	11	42	423300	57.0	7.81
445	447	13	42	439300	54.0	20.37

446 rows × 6 columns

```
cluster.drop('CustomerID', axis=1, inplace=True) # deleting customerID column
```

✓ 0.0s

- Mengelompokkan dataset berdasarkan customer ID
- Menghapus kolom customer ID karena tidak terpakai untuk proses clustering

Handling Outliers & Feature Scaling

```
# winsorizing outliers for right skewed distribution

wins_skw = Winsorizer(capping_method='iqr', fold=3, tail='both', variables=['Income'])
cluster = wins_skw.fit_transform(cluster)
```

✓ 0.0s

```
# winsorizing outliers for normal distribution

wins_nrm = Winsorizer(capping_method='gaussian', fold=3, tail='both', variables=['TransactionID','TotalAmount'])
cluster = wins_nrm.fit_transform(cluster)
```

✓ 0.0s

Feature Scaling

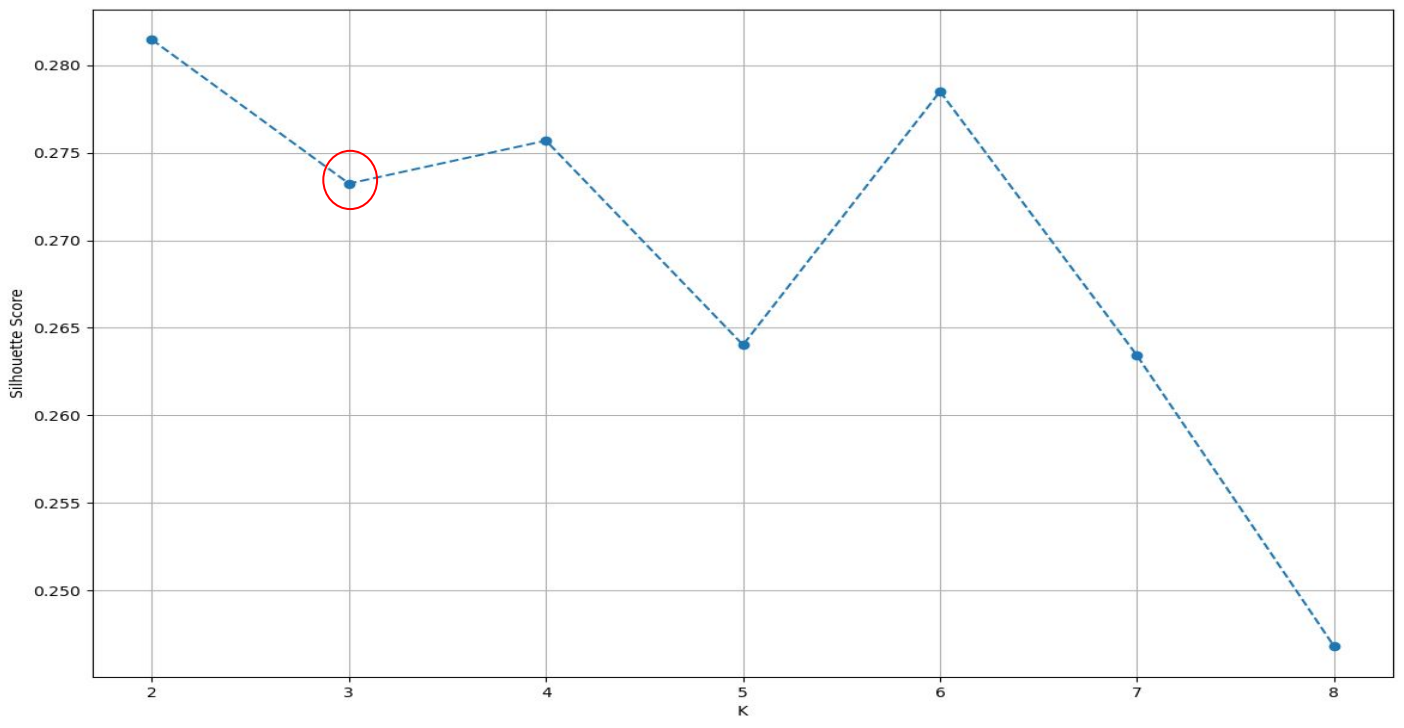
```
scaler = MinMaxScaler()
cluster = pd.DataFrame(scaler.fit_transform(cluster))
cluster.columns = scaler.get_feature_names_out()
cluster.head()
```

✓ 0.0s

	TransactionID	Qty	TotalAmount	Age	Income
0	0.779243	0.724638	0.824406	0.685185	0.166328
1	0.556602	0.681159	0.465901	0.777778	0.202388
2	0.667922	0.666667	0.549552	0.259259	0.297897
3	0.389621	0.521739	0.326534	0.240741	0.158207
4	0.222641	0.246377	0.273923	0.740741	0.115975

- Melakukan capping pada outliers
- Melakukan scaling dengan MinMax scaler

Silhouette Method



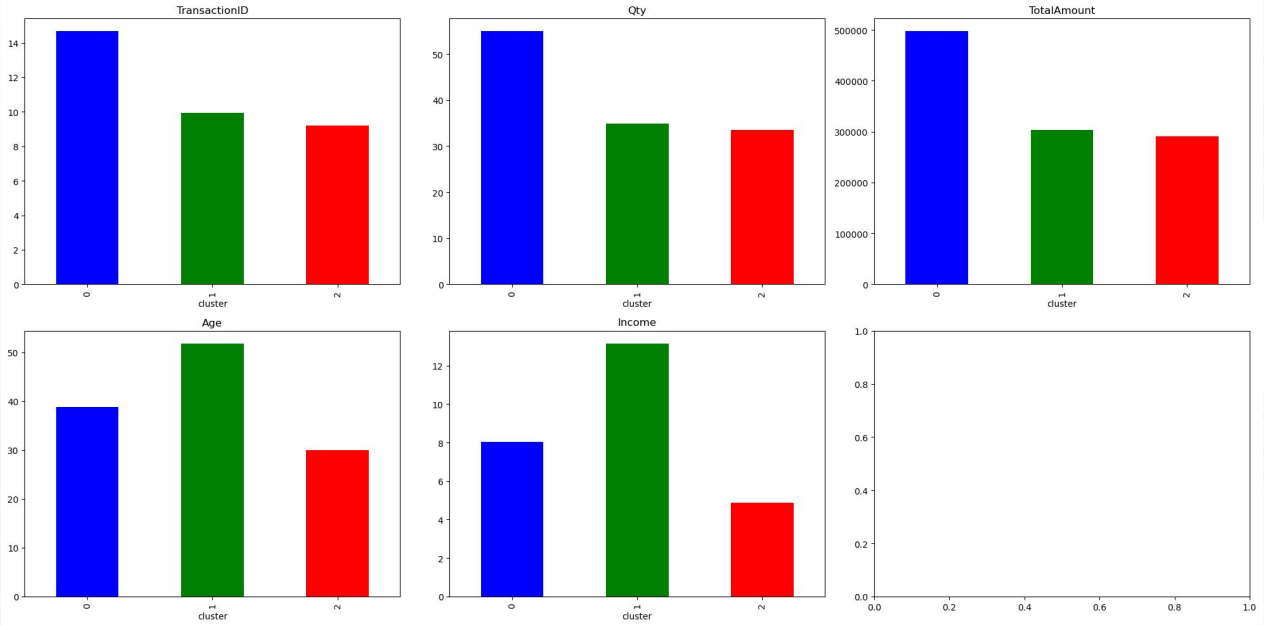
Handling Outliers & Feature Scaling

Rakamin Academy

```
df_eda.groupby('cluster')[['TransactionID','Qty','TotalAmount','Age','Income']].mean() # grouping dataframe based on cluster
```

✓ 0.0s

cluster	TransactionID	Qty	TotalAmount	Age	Income
0	14.671233	54.938356	497462.328767	38.828767	8.026301
1	9.937931	34.813793	303807.586207	51.813793	13.148759
2	9.212903	33.529032	290753.548387	29.974194	4.874129



Recommendations

Characteristic of cluster & Recommendation

- Cluster 0 : middle income customer with average of their age is 38, high transaction, quantity, and automatically high spending
- Cluster 1 : high income customer with average of their age is 51, more transaction than cluster 2 but relatively same quantity and spending
- Cluster 2 : low incom customer with average of their age is 29, low transaction, quantity, and spending

Recommendations :

- Cluster 0 :
    1. Personalized Recommendations: Use their transaction history to provide personalized product recommendations.
    2. High-Value Discounts: Offer discounts for high-value purchases to encourage them to spend even more
- Cluster 1 :
    1. VIP Services: Provide them with VIP treatment, such as dedicated customer service, express shipping, or extended warranties
    2. Promote them with healthy products
- Cluster 2 :
    1. Affordable Options: Focus on promoting budget-friendly products
    2. Discounts and Bargains: Offer discounts, flash sales, or clearance sales to attract cost-conscious shoppers.
    3. Bundled Savings: Create bundles of products that offer savings compared to buying items individually.

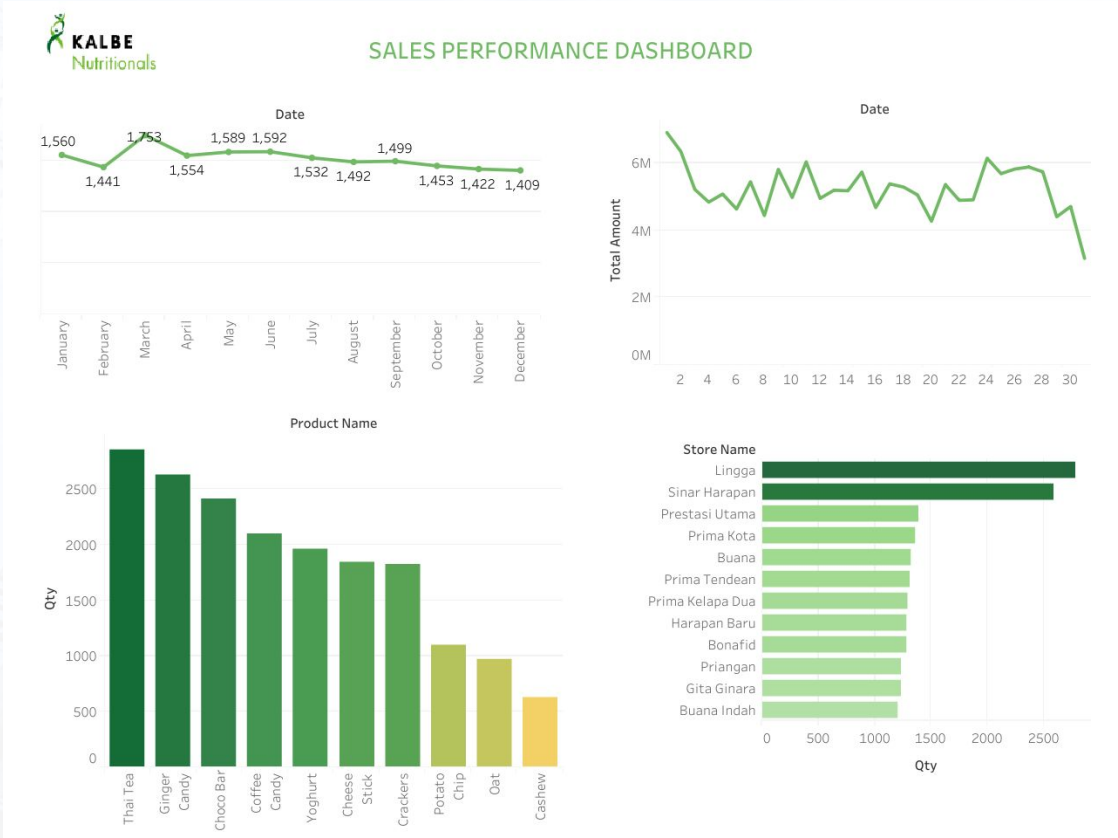
GitHub :

https://github.com/ridhwanmuttaqien/Kalbe_Nutritionals_Data_Scientist_Project_Based_Internship_Program

Rakamin Academy

Video Presentation :

https://drive.google.com/file/d/1vGVhCPsUraJ2akm6x3ZSCct0tmF7g-K8/view?usp=sharing



Link Tableau Public :
https://public.tableau.com/authoring/KalbeVIX_16959949535470/Dashboard1#1

Thank You