Rakamin Academy

Shipment Arrival Prediction

Homework
Stage 2

Pembagian Tugas

Stage 2:

- Nur Cahyanti (Feature Transformation; Quantile)
- Handika (Feature Transformation; Log, Standardization, Normalization)
- Fajar Nurdiono (Handling Outliers)
- Refanie Fajrina (Tidak perlu dilakukan di data ini: Null values, Duplicate values, Class Imbalance. Dilakukan di data ini: Feature Encoding)
- Utlia Rahma (Feature Selection and Extraction)
- Indra Laksana (4 feature tambahan)

Data Cleansing (Null and duplicate values)

Pertama kita cek apakah ada data yang kosong dan duplicate.

- Tidak ada data null.
- Tidak ada data duplicate.

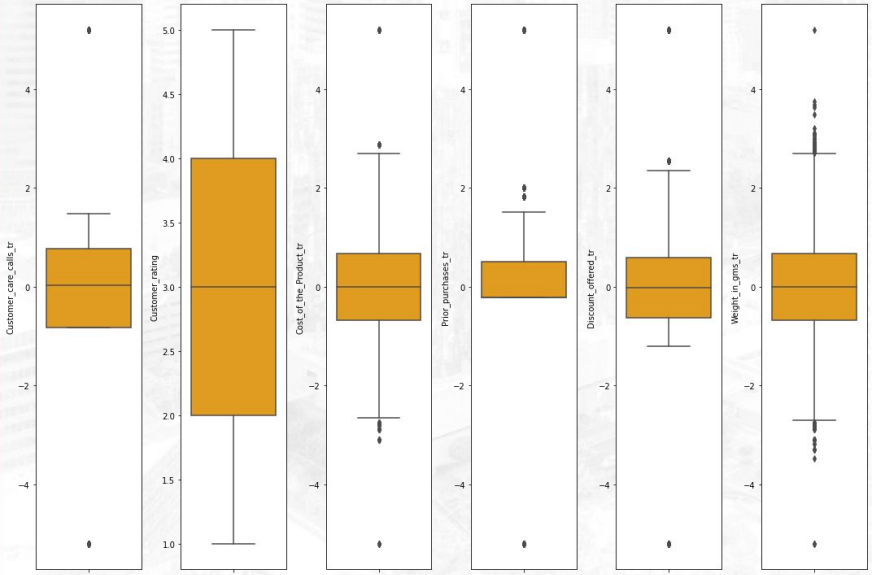
```
[26] df.isna().sum()

ID                     0
Warehouse_block        0
Mode_of_Shipment       0
Customer_care_calls    0
Customer_rating        0
Cost_of_the_Product    0
Prior_purchases        0
Product_importance     0
Gender                 0
Discount_offered       0
Weight_in_gms          0
Reached.on.Time_Y.N    0
dtype: int64
```

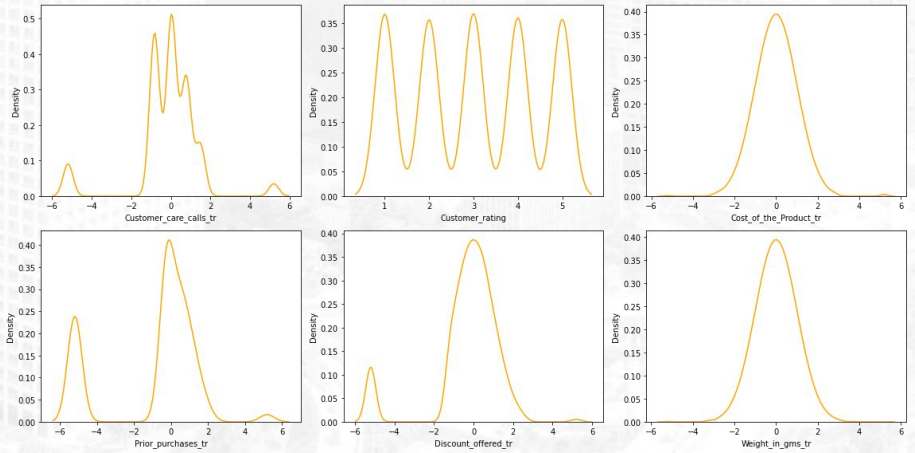
```
[27] df.duplicated().any()

False
```

Feature Transformation (Quantile)

Kami melakukan banyak jenis transformasi fitur untuk data ini, yaitu dengan log transform, quantile transform, normalization, dan standardization.

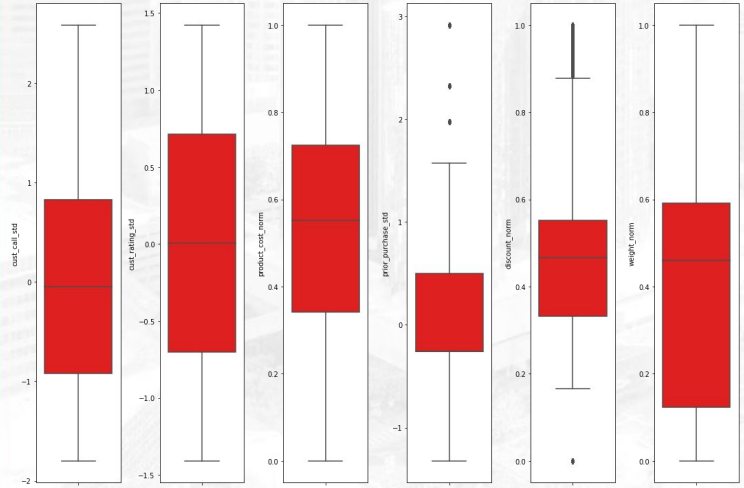
Quantile transformation digunakan pada data **Customer_care_calls_tr**, **Customer_rating**, **Cost_of_the_Product_tr**, **Prior_purchases_tr**, **Discount_offered_tr**, **Weight_in_gms_tr**

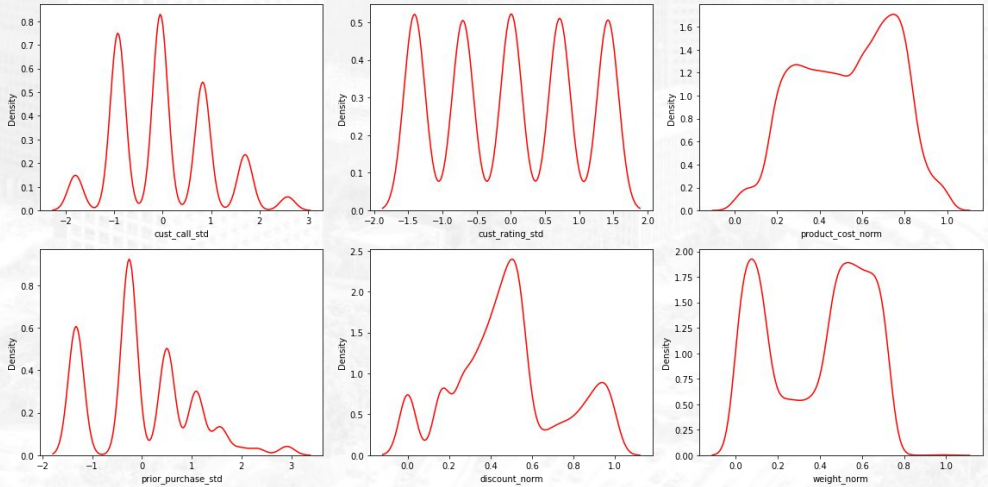




Feature Transformation (Log, Standard, Normal)

- Log Transformasi digunakan ke feature **Discount_offered** dan **Prior_purchases** karena 2 feature tersebut memiliki skew positif yang besar.
- Standarisasi digunakan ke feature **Customer_care_calls**, **Customer_rating**, dan **Prior_purchaces_log**.
- Normalisasi digunakan ke feature **Cost_of_the_Product**, **Discount_offered_log**, dan **Weight_in_gms**.





Data Cleansing (Outlier)

Kami memutuskan untuk membuat outlier dari kolom `**Discount_offered**` saja karena memang nilai outliernya cukup banyak.

Terdapat 903 baris yang dibuang setelah dilakukan filter IQR.

```
[44] print(f' jumlah baris sebelum memfilter outlier: {len(df)}')

     filtered_entries = np.array([True] * len(df)) # coba tambah line ini

     Q1 = df['Discount_offered_tr'].quantile(0.25)
     Q3 = df['Discount_offered_tr'].quantile(0.75)
     IQR = Q3 - Q1
     low_limit = Q1 - (1.5 * IQR)
     high_limit = Q3 + (1.5 * IQR)
     filtered_entries = ((df['Discount_offered_tr'] >= low_limit) & (df['Discount_offered_tr'] <= high_limit))
     df = df[filtered_entries]

     print(f' jumlah baris sesudah memfilter outlier: {len(df)}')

      jumlah baris sebelum memfilter outlier: 10999
      jumlah baris sesudah memfilter outlier: 10096
```

Class Imbalance

MENCARI TARGET FEATURE YANG DATANYA TIDAK BERIMBANG

Rasio nilai 0 dan 1 pada kolom target adalah 40:60 dan distribusinya tidak sangat timpang, maka dari itu tidak ada class imbalance pada data ini.

```
[50] late = df.groupby(['Reached.on.Time_Y.N']).agg({'ID':'nunique'}).reset_index()
     late.columns = ['Late','Total']
     late['Percentage'] = late.apply(lambda x: round(x['Total']/10999*100,2),axis=1)
     late
```

	Late	Total	Percentage
0	0	4009	36.45
1	1	6087	55.34

Feature Encoding

1. Untuk kolom Product Importance dan Gender kami menggunakan proses perubahan data manual

```
[51] # label encoding mapping cats yang punya 2 distinct value / ordinal
     # Product_importance dan Gender
     # jenis_kelamin & pendidikan
     mapping_gender = {
         'F' : 0,
         'M' : 1
     }

     mapping_product_importance = {
         'low' : 0,
         'medium' : 1,
         'high' : 2 # dapat dihilangkan karena kalau low = 0, medium = 0 sudah otomatis sama dengan high
     }

     df['Gender'] = df['Gender'].map(mapping_gender)
     df['Product_importance'] = df['Product_importance'].map(mapping_product_importance)
```

2. Untuk feature Warehouse Block dan Mode of Shipment, kami menggunakan One Hots Encoding, untuk melihat tingkatan antar data lebih representatif :

```
[48] Int64Index: 10096 entries, 0 to 10998
     Data columns (total 29 columns):
      #   Column                  Non-Null Count  Dtype
     ---  ------                  --------------  -----
      0   ID                      10096 non-null  int64
      1   Customer_care_calls     10096 non-null  int64
      2   Customer_rating         10096 non-null  int64
      3   Cost_of_the_Product     10096 non-null  int64
      4   Prior_purchases         10096 non-null  int64
      5   Product_importance      10096 non-null  int64
      6   Gender                  10096 non-null  int64
      7   Discount_offered        10096 non-null  int64
      8   Weight_in_gms           10096 non-null  int64
      9   Reached.on.Time_Y.N     10096 non-null  int64
      10  Cost_of_the_Product_tr  10096 non-null  float64
      11  Discount_offered_tr     10096 non-null  float64
      12  Customer_care_calls_tr  10096 non-null  float64
      13  Weight_in_gms_tr        10096 non-null  float64
      14  Prior_purchases_tr      10096 non-null  float64
      15  Discount_offered_log    10096 non-null  float64
      16  Prior_purchases_log     10096 non-null  float64
      17  cust_call_std           10096 non-null  float64
      18  cust_rating_std         10096 non-null  float64
      19  prior_purchase_std      10096 non-null  float64
      20  product_cost_norm       10096 non-null  float64
      21  discount_norm           10096 non-null  float64
      22  weight_norm             10096 non-null  float64
      23  Warehouse_block_B       10096 non-null  uint8
      24  Warehouse_block_C       10096 non-null  uint8
      25  Warehouse_block_D       10096 non-null  uint8
      26  Warehouse_block_F       10096 non-null  uint8
      27  Mode_of_Shipment_Road   10096 non-null  uint8
      28  Mode_of_Shipment_Ship   10096 non-null  uint8
     dtypes: float64(13), int64(10), uint8(6)
     memory usage: 1.9 MB
```

Feature Selection

Feature yang tidak diseleksi, yakni **`ID`**, **`Customer_care_calls`**, **`Customer_rating`**

Hal ini disebabkan karena ketiga feature tersebut tidak berkorelasi terkait tepat waktu atau tidaknya pengiriman.

Feature 2 dan 3 dibuang karena dapat menyebabkan data leakage, yang berarti bahwa data dari kedua feature ini sebenarnya belum ada di saat model digunakan untuk memprediksi pengiriman akan terlambat atau tidak.

```
[51] #menyeleksi kolom feature ID

     df_selection = df.drop(columns = ['ID','Customer_care_calls','Customer_rating'])
     df_selection.head()
```

```
[52] #memgekstraksi antara feature Cost dengan Weight (cost per weight)
     #mengekstraksi antara feature Cost_Product dengan Diskon yang ditawarkan. jadi terlihat cost sebelum dan sesudah diskon

     df_selection['Cost_Per_Weight'] = (df_selection['Cost_of_the_Product'] / df_selection['Weight_in_gms'])
     df_selection['Cost_After_Disc'] = (df_selection['Cost_of_the_Product'] - df_selection['Discount_offered'])
```

```
[53] #mengekstrasi Level Weight
     def segment(x):
         if x['Weight_in_gms'] < 3634:
             segment = 'Low'
         elif x['Weight_in_gms'] <= 5846:  #and x['Weight_in_gms'] >= 5846
             segment = 'Mid'
         elif x['Weight_in_gms'] >= 5846:
             segment = 'High'
         return segment

     df_selection['Weight_level'] = df_selection.apply(lambda x: segment(x), axis=1)
     df_selection.head()
```

Feature Extraction

- Kami menambah fitur `**Cost_Per_Weight**` karena memang di dunia shipping ada yang namanya rate harga untuk setiap unit berat. Karena unit berat yang kami pakai adalah gram, maka satuan dari fitur ini adalah $ per gram.
- Kami menambah fitur `**Cost_After_Disc**` untuk melihat harga produk setelah diberi diskon.
- Kami menambah fitur `**Weight_level**` untuk membagi nilai kolom `Weight_in_gms` menjadi 3 kategori berat dari Low, Middle, High.

```
[52] #memgekstraksi antara feature Cost dengan Weight (cost per weight)
     #mengekstraksi antara feature Cost_Product dengan Diskon yang ditawarkan. jadi terlihat cost sebelum dan sesudah diskon

     df_selection['Cost_Per_Weight'] = (df_selection['Cost_of_the_Product'] / df_selection['Weight_in_gms'])
     df_selection['Cost_After_Disc'] = (df_selection['Cost_of_the_Product'] - df_selection['Discount_offered'])
```

```
[53] #mengekstrasi Level Weight
     def segment(x):
         if x['Weight_in_gms'] < 3634:
             segment = 'Low'
         elif x['Weight_in_gms'] <= 5846:  #and x['Weight_in_gms'] >= 5846
             segment = 'Mid'
         elif x['Weight_in_gms'] >= 5846:
             segment = 'High'
         return segment

     df_selection['Weight_level'] = df_selection.apply(lambda x: segment(x), axis=1)
     df_selection.head()
```

4 Feature Tambahan

1. Waktu Pengiriman
Mengetahui kapan waktu yang sibuk dan longgar di setiap warehouse, sehingga dapat digunakan sebagai alat antisipasi potensi keterlambatan pengiriman.
2. Estimasi Waktu Barang Tiba
Mengetahui durasi pengiriman barang dari warehouse menuju tempat pengiriman, sehingga dapat membantu model menganalisis tingkat efektifitas dan efisiensi suatu pengiriman.
3. Region (Domestic / International)
Dokumen dan persyaratan pengiriman internasional memerlukan waktu yang lebih lama, sehingga dapat meningkatkan permodelan yang lebih akurat apabila pengiriman dapat dikategorikan berdasarkan region.
4. Jarak
Semakin jauh jarak yang ditempuh dalam pengiriman, menyebabkan lamanya waktu pengiriman, dengan adanya fitur jarak diharapkan model dapat memprediksi keterlambatan dari suatu pengiriman.