Aparilo*

Fashion
E-commerce

Python Data analytics & visualization
project:

E-Commerce Customers
Segmentation

by Ryandi Putra
LinkedIn: Ryandi Putra | email: dipermanaputra@gmail.com

Hello & welcome

to my data analytics presentation

***Aparilo E-commerce, a personal-made up ecommerce firm.**

Analytics were conducted using 4 years worth of data completed through utilization of Python programming and related statistical algorithm. The code script is accessible through the author's public GitHub repository and Google Colab.

Contents:

Objectives

Introducing Aparilo Ecommerce, a self-built online retail firm poised to revolutionize the fashion industry. Founded in 2021, Aparilo is a fast-growing player in the market, continuously enhancing its strategies to stay ahead. Our primary objective is to expand our customer base by acquiring new users while fostering loyalty and encouraging repeat purchases from existing customers.

To achieve these goals, Aparilo is conducting a thorough analysis of its sales, products, and user data through exploratory data analysis (EDA). In addition, Aparilo is aiming to cluster its existing users/customers and segment them based on purchase behavior to create more precise marketing campaigns that can improve conversion rates and revenue by deploying targeted marketing strategies.

DATA GATHERING

Processing, Cleaning, Transformation

Data dictionaries

The available datasets for this analytics:

Orders dataset

order_id	int: PRIMARY KEY
user_id	int: FOREIGN KEY to users dataset
status	string: Processing, Shipped, Canceled, Complete, Return
gender	string: male/female
created_at	datetime
returned_at	datetime
shipped_at	datetime
delivered_at	datetime
num_of_item	int

Order items dataset

id	int: PRIMARY KEY
order_id	int: FOREIGN KEY
user_id	int: FOREIGN KEY
product_id	int: FOREIGN KEY
inventory_item_id	int
status	object:
created_at	datetime
shipped_at	datetime
delivered_at	datetime
returned_at	datetime
sale_price	float

Products dataset

id	int: PRIMARY KEY
cost	float
category	string
name	string
brand	string
retail_price	float
department	string
sku	string
distribution_center_id	int

Users dataset

id	int: PRIMARY KEY
first_name	string
last_name	string
email	string
age	int
gender	string
state	string
street_address	string
postal_code	int
city	string
country	string
latitude	float
longitude	float
traffic_source	string
created_at	datetime

Preparing the analytics field: bringing in the right libraries

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
from datetime import datetime, timedelta

%matplotlib inline
warnings.filterwarnings("ignore")
pd.set_option('display.float_format', lambda x: '%.2f' % x)
```

There will be more libraries used in this task

Data cleaning

Data cleaning and formatting:

- Timestamp formatting

```
t_orders['created_at'] = pd.to_datetime(t_orders['created_at'], format = '%Y-%m-%d %H:%M:%S')
t_orders['returned_at'] = pd.to_datetime(t_orders['returned_at'], format = '%Y-%m-%d %H:%M:%S')
t_orders['shipped_at'] = pd.to_datetime(t_orders['shipped_at'], format = '%Y-%m-%d %H:%M:%S')
t_orders['delivered_at'] = pd.to_datetime(t_orders['delivered_at'], format = '%Y-%m-%d %H:%M:%S')
```

Dateime formatting will be useful in data aggregation with time condition filtering

- Handling missing values (null)

```
t_products.dropna(subset = ['department','sku','distribution_center_id'], inplace= True)
```

The missing values in dataset summed up to only 1.5% of total data. We can remove it to keep the data integrity

- Filling missing values with its correlated & most relevant and nearest subset

```
t_products['brand'].fillna(t_products['name'].str.split().str[:3].str.join(' '),
inplace= True)

t_products['name'].fillna(t_products['brand'], inplace= True)
```

- Fixing typos

```
def check_values(df):
  for c in df.columns:
    unique_qty = df[c].value_counts()
    print(f'Column: {c}')
    print(unique_qty, '\n')
```

```
check_values([any premade dataframe])
```

```
d_users['country']=d_users['country'].replace({'España':'Spain'})
```

Processed datasets overview:

orders, order items, products, and users

order_id	user_id	status	gender	created_at	returned_at	shipped_at	delivered_at	num_of_item
1	1	Returned	F	27/11/2022 16:02	02/12/2022 15:23	27/11/2022 21:43	01/12/2022 18:05	1
2	1	Shipped	F	23/09/2022 16:02	NaT	25/09/2022 23:03	NaT	1
3	1	Complete	F	21/01/2023 16:02	NaT	22/01/2023 08:33	22/01/2023 21:28	1
4	4	Complete	F	03/12/2021 00:41	NaT	03/12/2021 21:22	08/12/2021 17:24	1
5	5	Returned	F	16/04/2021 08:41	22/04/2021 08:02	18/04/2021 07:47	21/04/2021 10:57	3

125082 rows × 9 columns

id	order_id	user_id	product_id	inventory_item_id	status	created_at	shipped_at	delivered_at	returned_at	sale_price
1	1	1	11767	2	Returned	27/11/2022 13:21	27/11/2022 21:43	01/12/2022 18:05	02/12/2022 15:23	26.99
2	2	1	7	4	Shipped	23/09/2022 13:45	25/09/2022 23:03	NaT	NaT	39.5
3	3	1	5669	6	Complete	21/01/2023 14:46	22/01/2023 08:33	22/01/2023 21:28	NaT	28
4	4	4	12947	10	Complete	03/12/2021 00:39	03/12/2021 21:22	08/12/2021 17:24	NaT	19.9

181735 rows × 11 columns

id	cost	category	name	brand	retail_price	department	sku	distribution_center_id	
0	27569	92.65	Swim	2XU Men's Swimmers Compression Long Sleeve Top	2XU	150.41	Men	B23C5765E165D83AA924FA8F13C05F25	1
1	27445	24.72	Swim	TYR Sport Men's Square Leg Short Swim Suit	TYR	38.99	Men	2AB7D3B23574C3DEA2BD278AFD0939AB	1
2	27457	15.9	Swim	TYR Sport Men's Solid Durafast Jammer Swim Suit	TYR	27.6	Men	8F831227B0EB6C6D09A0555531365933	1
3	27466	17.85	Swim	TYR Sport Men's Swim Short/Resistance Short Sw...	TYR	30	Men	67317D6DCC4CB778AEB9219565F5456B	1

29095 rows × 9 columns

id	first_name	last_name	email	age	gender	state	street_address	postal_code	city	country	latitude	longitude	traffic_source	created_at
1	Laura	Barber	laurabarber@gmail.com	48	F	Hebei	49866 William Lodge	74099	Shanghai	China	39.33	115.88	Search	30/11/2021
2	Patrick	Nelson	patricknelson@gmail.com	29	M	Ceará	3452 Hoffman Mountain Suite 873	62900-000	Russas	Brasil	-4.84	-38.15	Organic	12/02/2020
3	Phillip	Parker	phillipparker@gmail.com	27	M	Bourgogne-Franche-Comté	9978 Dodson Drives Apt. 469	21200	Beaune	France	47.01	4.88	Facebook	24/08/2022
4	Bethany	West	bethanywest@gmail.com	32	F	Sachsen	4857 Bryan Ramp Suite 914	1796	Pirna	Germany	50.95	13.96	Facebook	05/03/2019

99037 rows × 15 columns

Data merging

Merging & once again merged data cleaning wil give us wider view of the data and make it usable.

order_id	user_id_x	product_id	status_x	created_at_x	shipped_at_x	delivered_at_x	returned_at_x	sale_ price	gender	...	category	brand	department	distribution _center_id	age	state	city	country	traffic_source	acc_made
92627	73944	13606	Shipped	16/09/2020 15:54	15/09/2020 14:00	NaT	NaT	2.5	F	...	Accessori	Scarf_trad	Women	3	51	Fujian	Hefei	China	Search	06/07/2020
63301	50599	13606	Complete	16/01/2023 10:52	16/01/2023 21:29	21/01/2023 10:49	NaT	2.5	F	...	Accessori	Scarf_trad	Women	3	69	Zhejiang	Shenzhen	China	Search	27/03/2019
51752	41347	28951	Shipped	08/03/2023 14:19	06/03/2023 19:05	NaT	NaT	3	M	...	Accessori	Nice Shad	Men	6	23	New Jers	Lawrence	United States	Search	27/02/2023
12085	9744	28951	Complete	15/10/2021 00:16	17/10/2021 16:29	20/10/2021 06:45	NaT	3	M	...	Accessori	Nice Shad	Men	6	43	Ceará	Guaraciab	Brasil	Search	31/01/2020

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 179862 entries, 0 to 179861
Data columns (total 21 columns):
 #   Column                  Non-Null Count   Dtype
---  ------                  --------------   -----
 0   order_id                179862 non-null  int64
 1   user_id_x               179862 non-null  int64
 2   product_id              179862 non-null  int64
 3   status_x                179862 non-null  object
 4   created_at_x            179862 non-null  datetime64[ns]
 5   shipped_at_x            117079 non-null  datetime64[ns]
 6   delivered_at_x          63381 non-null   datetime64[ns]
 7   returned_at_x           18143 non-null   datetime64[ns]
 8   sale_price              179862 non-null  float64
 9   gender                  179862 non-null  object
 10  cost                    179862 non-null  float64
 11  category                179862 non-null  object
 12  brand                   179862 non-null  object
 13  department              179862 non-null  object
 14  distribution_center_id  179862 non-null  int64
 15  age                     179862 non-null  int64
 16  state                   179862 non-null  object
 17  city                    179862 non-null  object
 18  country                 179862 non-null  object
 19  traffic_source          179862 non-null  object
 20  acc_made                179862 non-null  datetime64[ns]
dtypes: datetime64[ns](5), float64(2), int64(5), object(9)
memory usage: 28.8+ MB
```

```
d_orderitem = pd.merge(d_oitems, d_orders, on = 'order_id', how = 'left')
```

Dropping irrelevant columns

```
d_op = pd.merge(d_orderitem, clean_products, left_on= 'product_id', right_on= 'id', how= 'left')
```

Dropping irrelevant columns

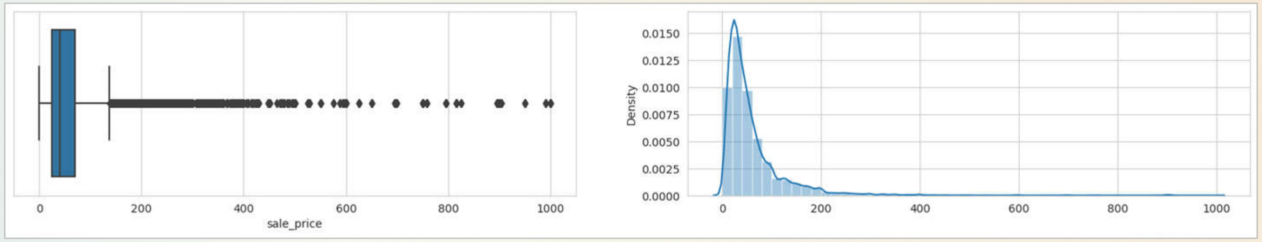
```
df_a = pd.merge(d_op, clean_users, left_on= 'user_id_x', right_on='id', how= 'left')
```

Final cleaning & formatting until we achieve level state on each subset, except what remain as is to retain most of the data (timestamps on order stages).

Merged data shape: 179862 entries, 21 columns

Outliers removal

Outliers can take in various forms, including mistyped data points or values that significantly deviate from the range of other data points. As a business, it is crucial to identify and address outliers to ensure data integrity, maintain model assumptions, and enhance future model performance.



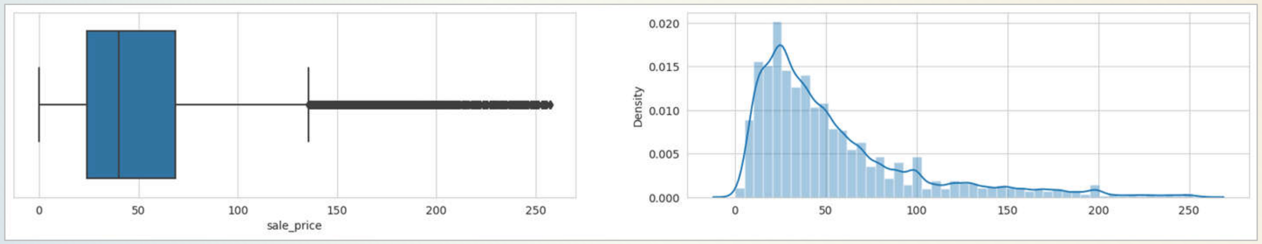
Data figure with outliers:

monetary value variable

We pick outliers out using Z-score method. For Z-score > 3 to preserve data as much as possible

```
from scipy import stats
z_s = stats.zscore(dfa['sale_price'])
```

```
dfc=dfa[(z_s<3)]
```

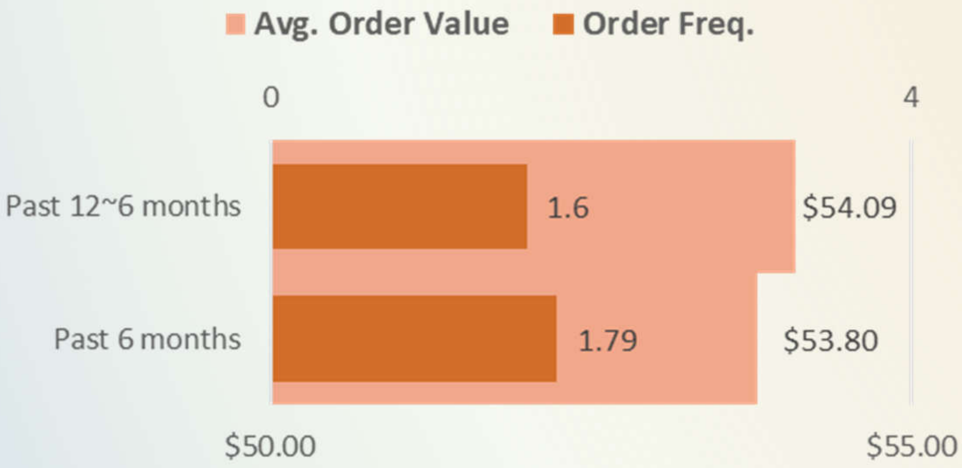


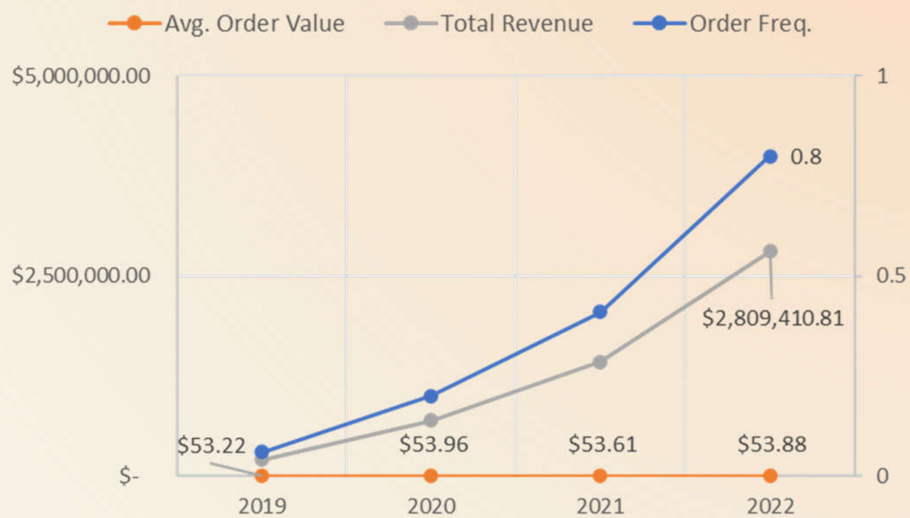
Data figure outliers removed

EDA

Exploratory Data Analysis

Main customer retention metrics between 2 periods and 2019 – 2022 yearly





In Figure 1, observation on 6-month comparison shows the order frequency in the most recent 6 months has increased by 11%. However, during the same period, there has been a slight decrease in the average order value by $0.29.

Fig. 2: Strong positive trends in total revenue and order frequency from 2019 to 2022, but average order value remains stagnant, indicating growth in revenue and order frequency without significant changes in average order value.

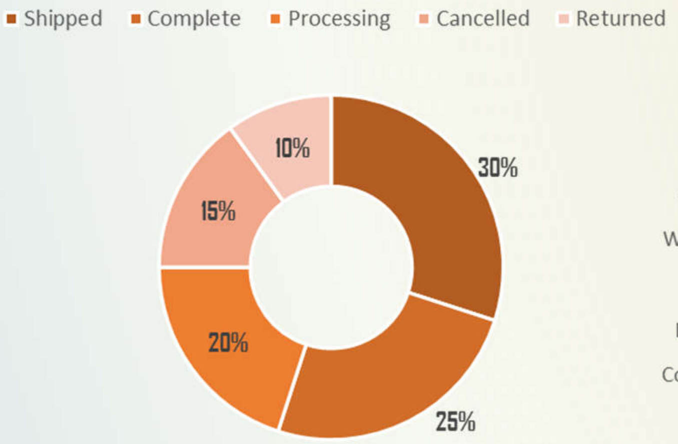
2019-2022 Revenue grouped by market country

Country	Rev($) 2019	Rev($) 2020	Rev($) 2021	Rev($) 2022
China	$ 70,363.25	$ 232,737.46	$ 487,254.32	$ 967,150.71
United States	$ 43,625.42	$ 160,154.03	$ 321,728.38	$ 631,857.98
Brasil	$ 29,196.96	$ 96,887.29	$ 196,751.40	$ 388,466.62
South Korea	$ 13,686.18	$ 41,068.72	$ 81,679.74	$ 148,757.25
United Kingdom	$ 9,960.60	$ 29,688.82	$ 64,773.99	$ 135,385.89
France	$ 11,803.99	$ 30,223.67	$ 64,256.33	$ 132,641.26
Germany	$ 7,757.78	$ 28,918.72	$ 61,895.80	$ 116,301.95
Spain	$ 7,441.33	$ 27,734.04	$ 55,182.12	$ 113,225.55
Japan	$ 4,728.68	$ 17,198.83	$ 40,339.61	$ 69,031.68
Australia	$ 3,971.61	$ 17,023.20	$ 29,576.24	$ 62,650.64
Belgium	$ 2,230.57	$ 6,666.30	$ 19,417.11	$ 36,407.03
Poland	$ 500.55	$ 1,301.46	$ 2,452.79	$ 7,170.82
Colombia	$ 59.99	$ -	$ 32.95	$ 311.30
Austria	$ -	$ -	$ -	$ 52.13

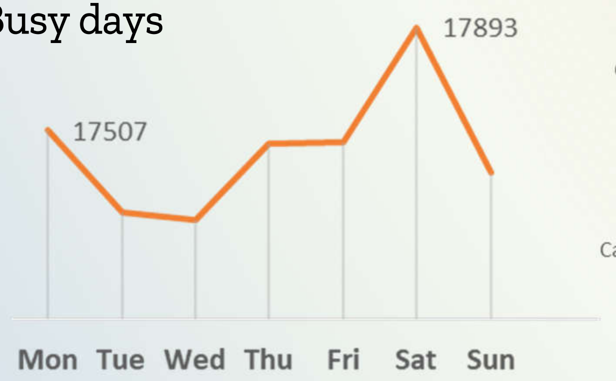
Fig 3: The strong positive trend is reflected in the revenue gained from each market country. Excitingly, we successfully penetrated the new European market in Austria in 2022, contributing to our overall growth.

Order activity, status & product brand performance

Firm's overall order status

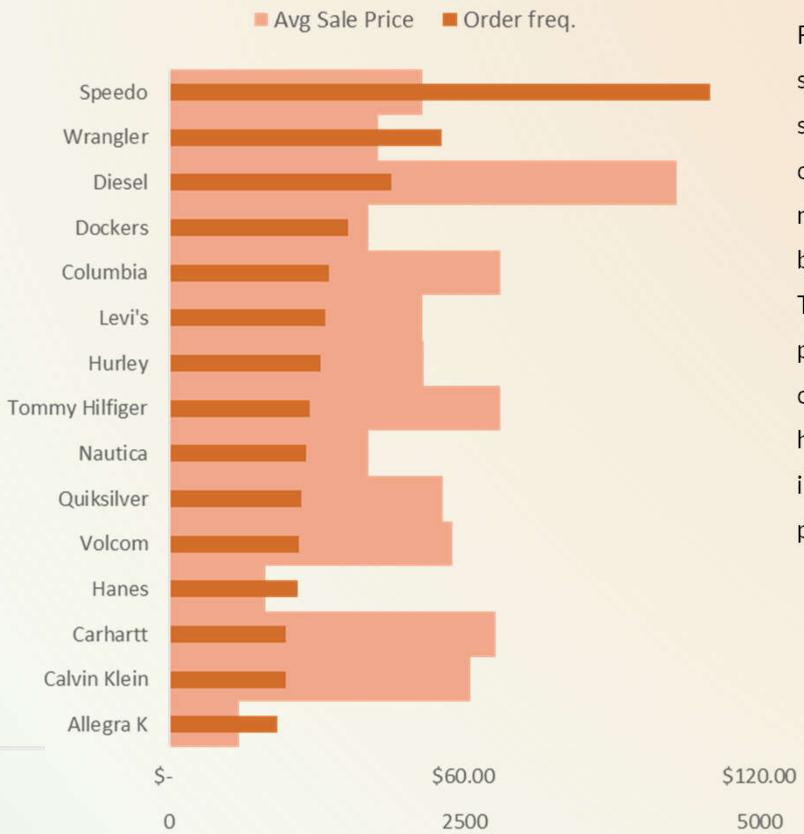


Busy days



Top selling brands

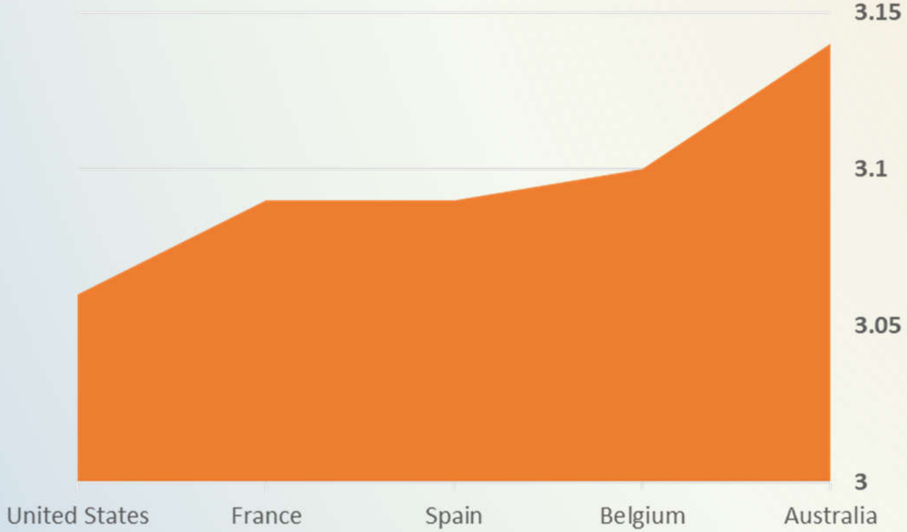
(average sale price relative to order frequency)



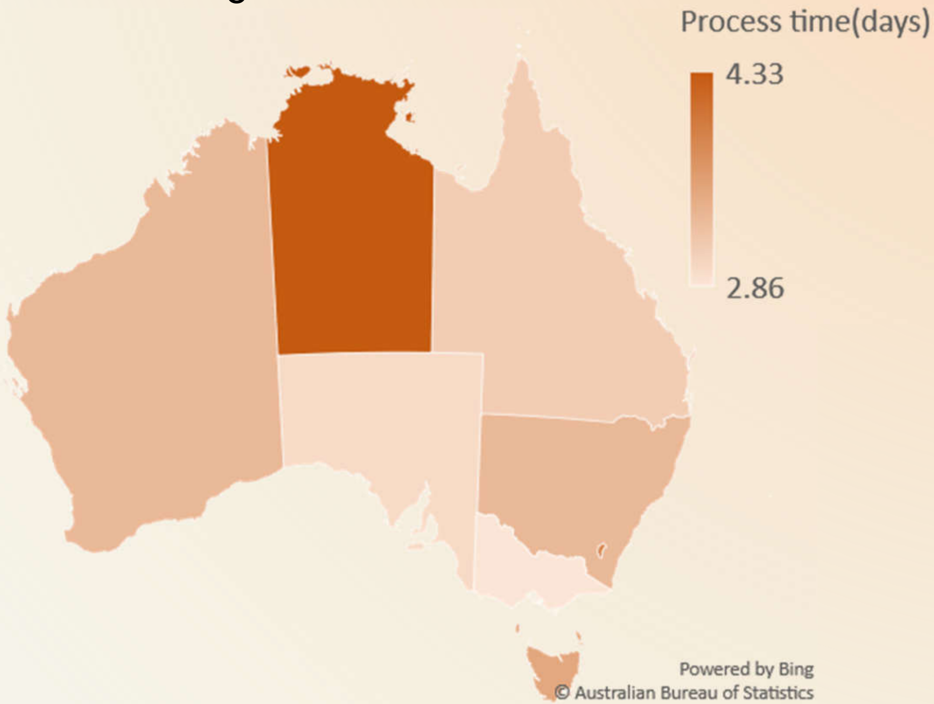
From the upper-left corner clockwise, Figure 4 showcases a promising performance with a significant number of shipped and completed orders, indicating a healthy trend. Moving to the next figure, we can observe the top selling brands in relation to their average sale price. This valuable insight enables us to fine-tune our pricing strategy or launch targeted marketing campaigns for low selling brands that have a high average sale price. By leveraging this information, we can optimize sales and profitability.

5 Countries with lowest performance shipping time

In Figure 7, starting from the lower-left corner and moving clockwise, shown the average process time from order creation until delivery. Australia stands out with the longest process time. This finding prompts us to investigate the processing time for each users' state in that country. From that, we can make informed decisions regarding the location and technology of our distribution centers. This information enables us to optimize our operations and improve customer satisfaction.

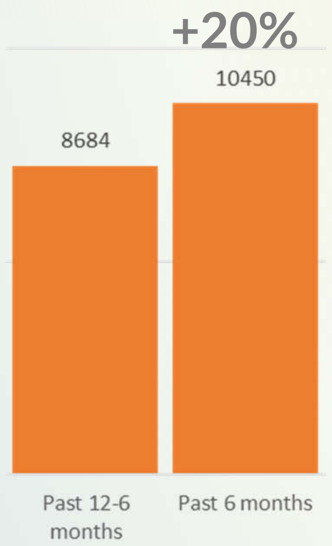


Zooming in to Australia

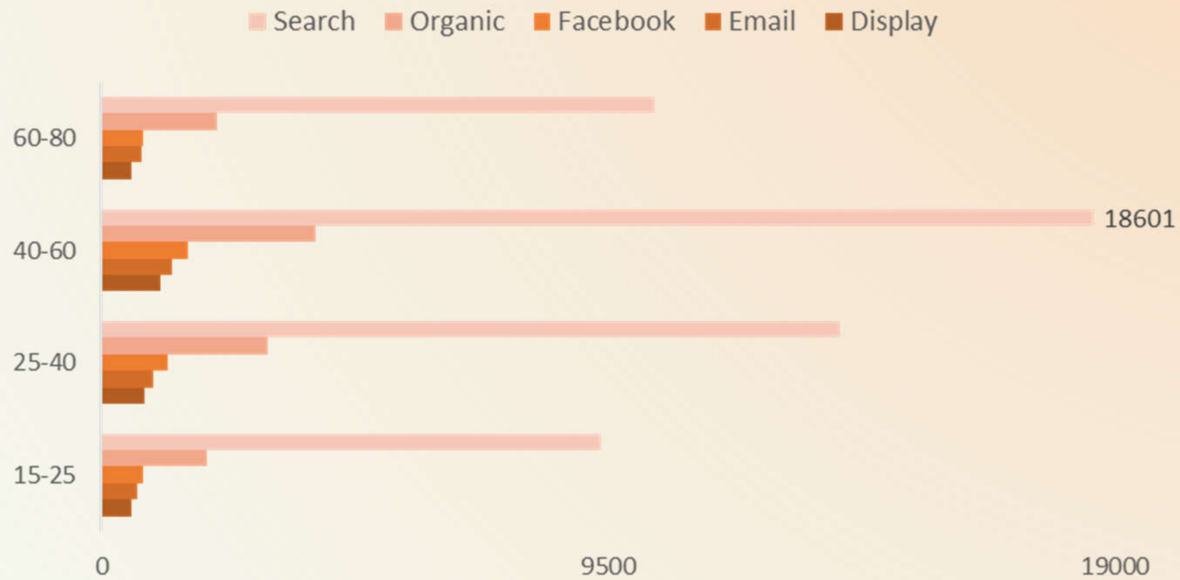


Users acquisition review

New accounts made in 6-month comparison



Users age group & traffic source pivot



There is a 20% increase in new user registrations in the past 6 months compared to previous 6 months period as shown in Figure 9.

Figure 10 provides insights into the traffic sources of our users, grouped by age. This information is crucial for our firm to effectively leverage each marketing campaign platform and determine the target audience for each source.

CUSTOMERS SEGMENTATION

Purchase Behavior Clustering

Clustering methodology

Now, we move on to phase where we segment customers based on their purchasing behavior in the market.

Considering the vast amount of data we have, we will leverage the power of machine learning algorithms by employing the K-Means algorithm in Python to cluster customers into meaningful segments based on their behavior.

Our plan of attack consists of the following steps:

1. Pick what kind of dataset that represents customers' behavior.
2. Clean, transform, and normalize the dataset using the right statistical methods.
3. Test the number of cluster (k) of Kmeans algorithm and produce the model.
4. Train the model with our dataset
5. Review buyers clustering result

Pick what kind of dataset that represents buyers' behavior

Filtering cleaned merged dataset where entries exclude 'Cancelled' and 'Returned' orders:

```
df_buy = dfclean[~dfclean['status'].isin(['Cancelled','Returned'])]
df_buy.drop(columns='returned', axis = 1, inplace=True)
```

Pivoting filtered dataset to a data that represent customers' behavior, in this case, based on what is provided, we pick RFM (Recency, Frequency, Monetary):

- Recency tells how many days since the last time X customer made purchase(s).
- Frequency tells how many purchase X customer made.
- Monetary (value) is the amount of currency spent from all purchase of customer X.

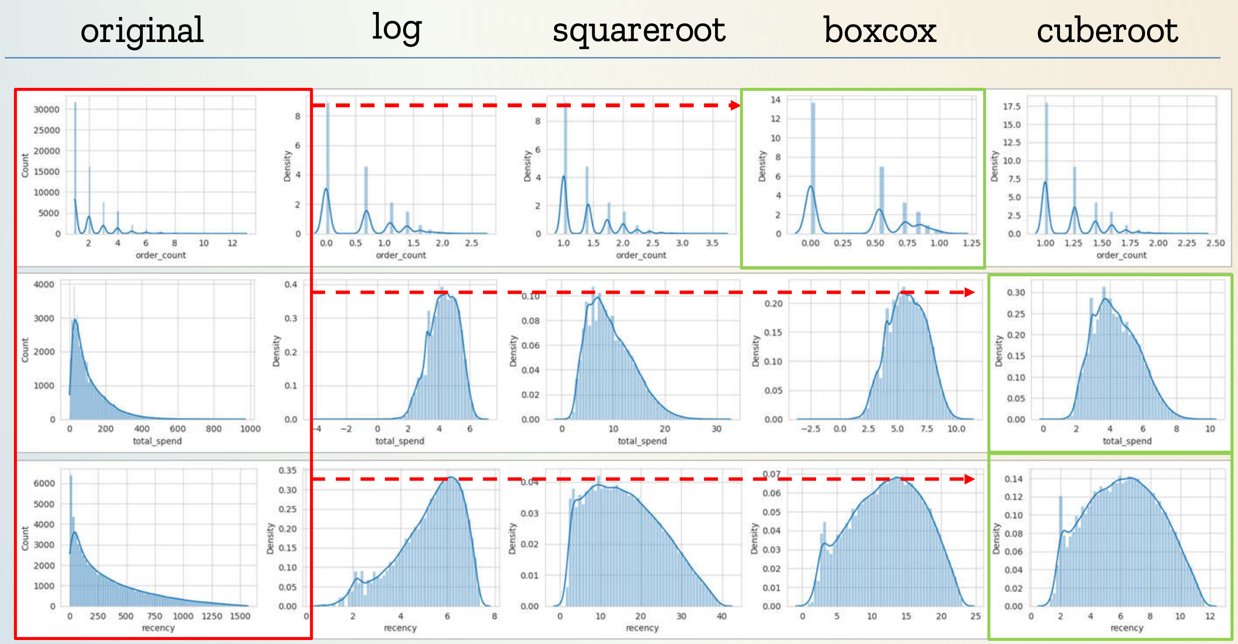
Therefore:

```
rfm = df_buy.groupby('user_id').agg(
    order_count=('order_id','count'),
    total_spend=('sale_price','sum'),
    recency=('created', lambda x: (datetime(2023,5,1)-x.max()).days))
```

user_id	order_count	total_spend	recency
1	2	67.50	99
4	1	19.90	513
6	4	233.44	116

65014 rows × 3 columns

Transformation methods evaluation




```
df= rfm
for c in df.columns:
    fig, ax = plt.subplots(1, 5, figsize=(20,3))
    sns.histplot(df[c], kde=True, ax=ax[0])
    sns.distplot(np.log(df[c]),ax=ax[1])
    sns.distplot(np.sqrt(df[c]),ax=ax[2])
    sns.distplot(stats.boxcox(df[c])[0],ax=ax[3])
    sns.distplot(np.cbrt(df[c]),ax=ax[4])
    plt.tight_layout()

    print(c,'\n','original: ', df[c].skew().round(2))
    print('log: ', np.log(df[c]).skew().round(2))
    print('squareroot: ', np.sqrt(df[c]).skew().round(2))
    print('boxcox: ',
pd.Series(stats.boxcox(df[c])[0]).skew().round(2))
      print('cuberoot', np.cbrt(df[c]).skew().round(2), '\n')
```

```
order_count
 original:  1.73
log:  0.64
squareroot:  1.09
boxcox:  0.25
cuberoot 0.92
```

```
total_spend
 original:  1.75
log:  -0.29
squareroot:  0.71
boxcox:  -0.02
cuberoot 0.39
```

```
recency
 original:  1.18
log:  -0.76
squareroot:  0.35
boxcox:  -0.11
cuberoot 0.03
```

The primary condition that must be met before undertaking clustering analysis is that the data should be assumed to follow a normal distribution with zero or nearly zero skewness. In the figure provided above, we have thoroughly assessed all commonly used transformation methods and carefully selected the distribution plot that exhibits the most favorable shape, devoid of any values below zero.

Datapoints transforming & normalizing

Transforming data distribution to around 0 skewness

```
dist_rfm['order_count'] =
stats.boxcox(dist_rfm['order_count'])[0]
dist_rfm['recency'] = np.cbrt(dist_rfm['recency'])
dist_rfm['total_spend'] = np.cbrt(dist_rfm['total_spend'])
```

```
dist_rfm.describe()
```

	order_count	total_spend	recency
count	65014.00	65014.00	65014.00
mean	0.35	4.40	6.20
std	0.36	1.34	2.39
min	0.00	0.27	1.26
25%	0.00	3.36	4.31
50%	0.53	4.27	6.21
75%	0.74	5.35	8.05
max	1.10	9.91	11.62

Is that enough? Well, not quite. Moving forward, our next step involves standardizing the values of different variables within the RFM dataset to a common scale. We will accomplish this using the Sklearn StandardScaler method. By employing this method, we can ensure that all variables have a mean of 0 and a standard deviation of 1. In simpler terms, our objective is to achieve an "apple-to-apple" comparison by making each data series equivalent in scale.

Normalize the data to fit into common scale

```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
normal_rfm[['order_count','total_spend','recency']] =
scaler.fit_transform(normal_rfm[['order_count','total_spend','recency']])
```

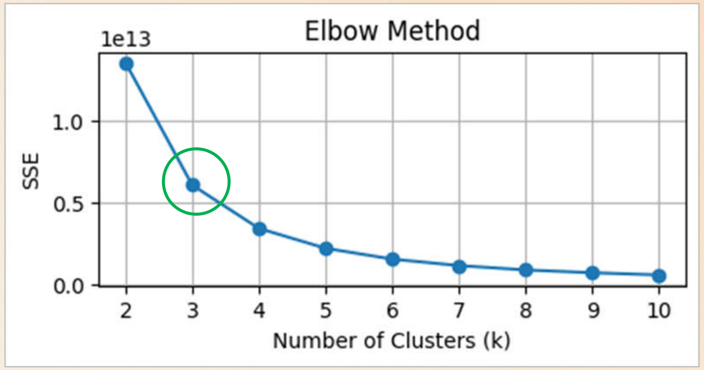
```
normal_rfm.describe()
```

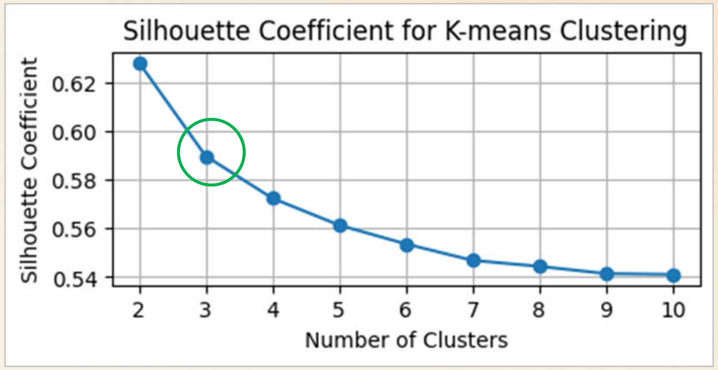
	order_count	total_spend	recency
count	65014.00	65014.00	65014.00
mean	-0.00	-0.00	0.00
std	1.00	1.00	1.00
min	-0.97	-3.07	-2.06
25%	-0.97	-0.77	-0.79
50%	0.51	-0.09	0.00
75%	1.08	0.71	0.77
max	2.10	4.10	2.26

K value of cluster evaluation

```
from sklearn.cluster import KMeans
sse = []
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, n_init=7, random_state=42)
    kmeans.fit(normal_rfm)
    sse.append(kmeans.inertia_)
```

```
from sklearn.metrics import silhouette_score
silhouette_coefficients = []
for k in range(2, 11):
    kmeans = KMeans(n_clusters=k, n_init=7, random_state = 42)
    kmeans.fit(normal_rfm)
    score = silhouette_score(normal_rfm, kmeans.labels_)
    silhouette_coefficients.append(score)
```





Elbow method & Silhouette coefficient. Rather than picking one from another, these methods are better used in complementary manner.

- Elbow method use SSE (sum of squared error) as indicator to define optimal k value. The best k value is where the SSE begin to make steady & linear pattern in its plot.
- Silhouette coefficient values the quality of:
  1. How close the data point is to other points in the cluster
  2. How far away the data point is from points in other clusters

The higher the better

Algorithm training

So we picked our k value. What's next? Train the K-means algorithm to our 0 skewness distributed, normalized RFM dataset. We use **.fit()** method in Python to train the model to the dataset. Then we melt the data for plotting purpose.

In this case, we will use Seaborn snake plot.

```
model = KMeans(n_clusters=3, n_init=7, random_state=42)
model.fit(normal_rfm)
model.labels_
```

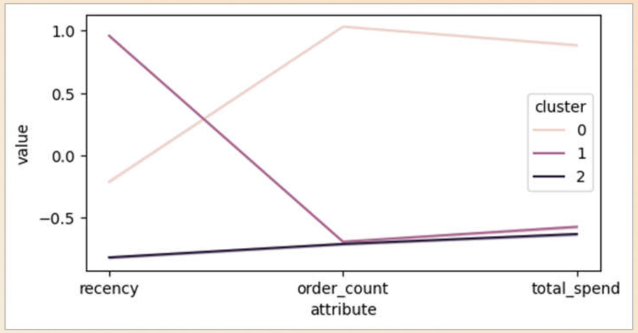
```
normal_rfm['cluster'] = model.labels_
```

```
meltrfm = pd.melt(normal_rfm,
              id_vars = ['user_id', 'cluster'],
              value_vars = ['recency','order_count','total_spend'],
              var_name = 'attribute',
              value_name = 'value')
```

```
plt.figure(figsize = (6,3))
plt.grid(False)
sns.lineplot(x='attribute', y='value', hue='cluster', data=meltrfm)
```

	user_id	cluster	attribute	value
0	1	2	recency	-0.66
1	4	1	recency	0.76
2	6	0	recency	-0.55
3	7	0	recency	-0.75
4	8	2	recency	-0.09
...	...	...	...	...
195037	99994	1	total_spend	-0.38
195038	99996	2	total_spend	-1.25
195039	99998	1	total_spend	-1.26
195040	99999	2	total_spend	-0.38
195041	100000	0	total_spend	2.18

195042 rows × 4 columns



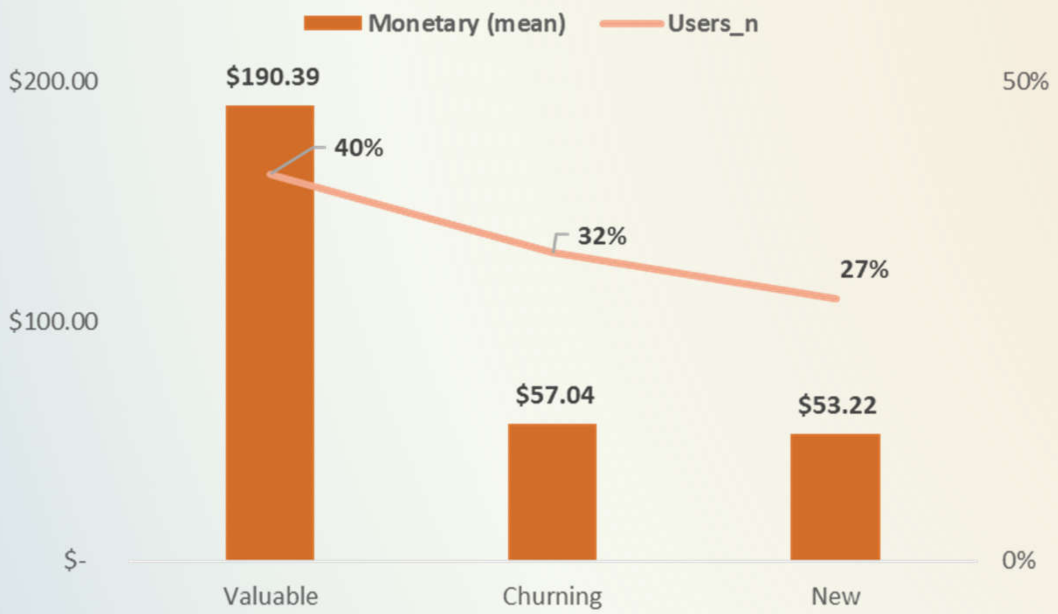
cluster	Recency	Frequency_order	Monetary_value	Users_n
0	-0.21	1.03	0.88	26256
1	0.96	-0.69	-0.57	20933
2	-0.82	-0.71	-0.63	17825

Conclusion:

•cluster 0 = **low** Recency, **highest** Frequency, **highest** Spending: **valuable users**

•cluster 1 = **highest** Recency, **low** Frequency, **low** Spending: **churning users**

•cluster 2 = **lowest** Recency, **lowest** Frequency, **lowest** Spending: **new users**



Customer segmentation result readings:

1. Valuable users make up 40% of Aparilo's total users: they contributes the least amount of both in order count and order monetary value.
2. Churning users make up 32% of Aparilo's total users: they contributes both in order count and order monetary in almost the same level with new users.
3. New users make up 27% of Aparilo's total users: they contributes the least amount of both in order count and order monetary value.

What can we do based on all this insights?

Conclusion

In 2022, Aparilo had an increase in total revenue & average order frequency compared to previous years. However, in YTD alone, in spite of increasing average order frequency, average order value suffered a minor decline compared to previous 6 months period, this reflected in stagnant average order over value in each year.

China, US, and Brazil are leading as our main market that represent its region, Asia, North America, and South America. This is a cue for Aparilo to penetrate harder into European market. However, in 2022 Aparilo successfully gained market share in new country Austria.

In distribution & shipping domain, Australia is country with the poorest shipping velocity, averaging in almost 3.15 days.

There is a 20% increase in new user registrations in the past 6 months compared to previous 6 months period and web search contributes the most Aparilo users of all time.

Recommendations:

1. Make a follow up pricing strategy on certain product categories & brands based on their sales count performance. We want to keep the surprise element as low as possible.
2. Review all distribution center points then make both system-wise & geographical adjustments to achieve better/swiftier processing time for customer satisfactory.
3. From users segmentation, firm can make informed marketing decisions & action items accordingly such as, for:

a. Segment: Valuable Customers

Marketing Campaign Style: Appreciation & Loyalty Building

Goal: Maintain and enhance the relationship with high-value customers.

b. Segment: Churning Customers

Marketing Campaign Style: Retention & Re-engagement

Goal: Win back and re-engage at-risk customers.

c. Segment: New Customers

Marketing Campaign Style: Onboarding & Introduction

Goal: Nurture and convert new customers into repeat buyers.

Thank you for being a part of this presentation

Ryandi Putra, 2023