Mutual Fund Investment Startup
~Python for Data Analytics~

•••

Fathan Abdallah

List of Contents :

Python Milestone 1

(Data Cleaning and EDA)

Analytical Objective

1. There are 4 types of investing such as saham, pasar uang, pendapatan tetap and campuran so customers can choose among those 4 options
2. Because of it is investing company so we gain profit from customers who doing transactIts investing dataset (mutual fund), so we can analyze according to the dataset and macro/micro economic
3. Profit comes from transactions fee (buy and sell fee) or profit sharing if gaining capital gain

Data Cleaning Users

Copy	• Make a copy as df_users_dc
Checking Data Types	• Melihan ukuran row x kolom dengan .shape() • Melihat informasi keseluruhan dengan .info() • Mencari nilai blank • Convert kolom data (object) menjadi datetime • Mengubah user_id menjadi string karena tidak diperlukan dalam proses operasi
Treat Missing / Irrelevant Values	• Drop user_occupation and user_income_source becaus irrelevant • In referral_code_used replace blank values with unknown
Check for Values & Typo	• Check values each columns

Data Cleaning Users

Data Manipulation	Clustering for each income range into 3 categories to make easier to identify
Checking Duplicates	No duplicates found

Data Cleaning Daily Transaction

Copy	● Make a copy as df_DT_dc
Checking Data Types	● Melihat ukuran row x kolom dengan .shape() ● Melihat informasi keseluruhan dengan .info() ● Mencari nilai blank ● Convert kolom data (object) menjadi datetime ● Mengubah user_id menjadi string karena tidak diperlukan dalam proses operasi
Treat Missing / Irrelevant Values	● There are many missing values in the dataset. But it doesnt mean we can just remove it. 0 or blank values means there is no transaction not there is no data. So, yes I consider it as data ● The information we can take from zero/blank data is the frequency of the transaction, so 0/blank values will be treated in data manipulation part

Data Cleaning Daily Transaction

Data Manipulation

- From daily transaction dataset we can take some information such number of frequencies an user balanced
- In this part I split "Daily transaction" into 2 separated dataset
- Those are frequency dataset and last balanced of user_id

Find the Frequencies

- Make a copy from “df_DT_dc” as “df_DT_freq”
- Find the number of frequencies by using .count() function
- add frequencies between buy and sell, also drop the columns amount because we just got the information we needed

Find User Balance

- Make a copy from “df_DT_dc” as “df_DT_balance”
- Sort by user_id and date to find out the last balance
- Rename column to make it compact
- Drop duplicates and keep the last one (after sorted) to get the last balance in the last date
- **Merge “Frequencies Table” and “Balance Table” as “df_daily_tran”**
- **Merge “df_users” and “df_daily_tran” as “df_merged”**

Exploratory Data Analytics (EDA)

1. Make a copy from df.merged as df_eda
2. Analyze the characteristic of the table with .info() function
3. Descriptive analysis about numeric information

```
pd.set_option('display.float_format', lambda x: '%.3f' % x)

numeric = ['user_age','end_of_month_invested_amount','total_buy_amount',
           'total_sell_amount','Saham_Freq','Pasar_Uang_Freq','Pend_Tetap_Freq',
           'Campuran_Freq','saham_bal','uang_bal','pend_tetapbal','campuran_bal','total_balance']

desc_eda = df_eda[numeric].describe()
desc_eda.loc['kurtosis'] = df_eda[numeric].kurt() #show kurtosis statistic
desc_eda.loc['skewness'] = df_eda[numeric].skew() #show Skewness statistic
desc_eda.loc['variance'] = df_eda[numeric].var() #show variance statistic
desc_eda.round(2)
```

Colab Link

Exploratory Data Analytics (EDA)

4. Descriptive analysis about string variable

```
objects = ['user_id',
 'user_gender',
 'user_income_range',
 'referral_code_used']
df_eda[objects].describe()
```

	user_id	user_gender	user_income_range	referral_code_used
count	8277	8277	8277	8277
unique	8277	2	3	2
top	3816789	Male	silver	unknown
freq	1	5176	6233	5322

Exploratory Data Analytics (EDA)

5. Descriptive analysis about date type variable

```
df_eda['registration_import_datetime'].describe()
```

6. Number of Customer

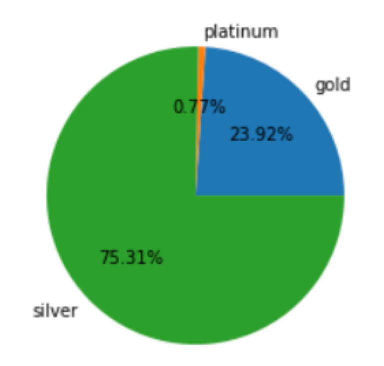
```
[90] #Number of customer
     df_eda['user_id'].count()
```

```
8277
```

7. User_income_range

```
#Count the id as users and group it by the loan status
#Reset the index to make it into a pandas dataframe

income_category = df_eda.groupby('user_income_range')['user_id'].count()
income_category = income_category.reset_index()
income_category
```

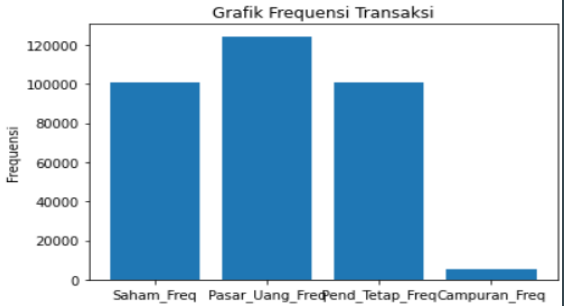
	user_income_range	user_id
0	gold	1980
1	platinum	64
2	silver	6233



Exploratory Data Analytics (EDA)

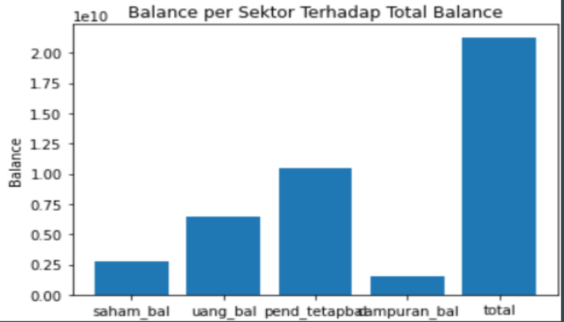
8. Transaction Frequency

```
Saham_Freq = df_eda['Saham_Freq'].sum()
Pasar_Uang_Freq = df_eda['Pasar_Uang_Freq'].sum()
Pend_Tetap_Freq = df_eda['Pend_Tetap_Freq'].sum()
Campuran_Freq = df_eda['Campuran_Freq'].sum()

print(f"saham freq: {Saham_Freq}")
print(f"uang freq: {Pasar_Uang_Freq}")
print(f"pendapatan tetap freq: {Pend_Tetap_Freq}")
print(f"campuran freq: {Campuran_Freq}")
```



9. Sector Balance to Total Balance

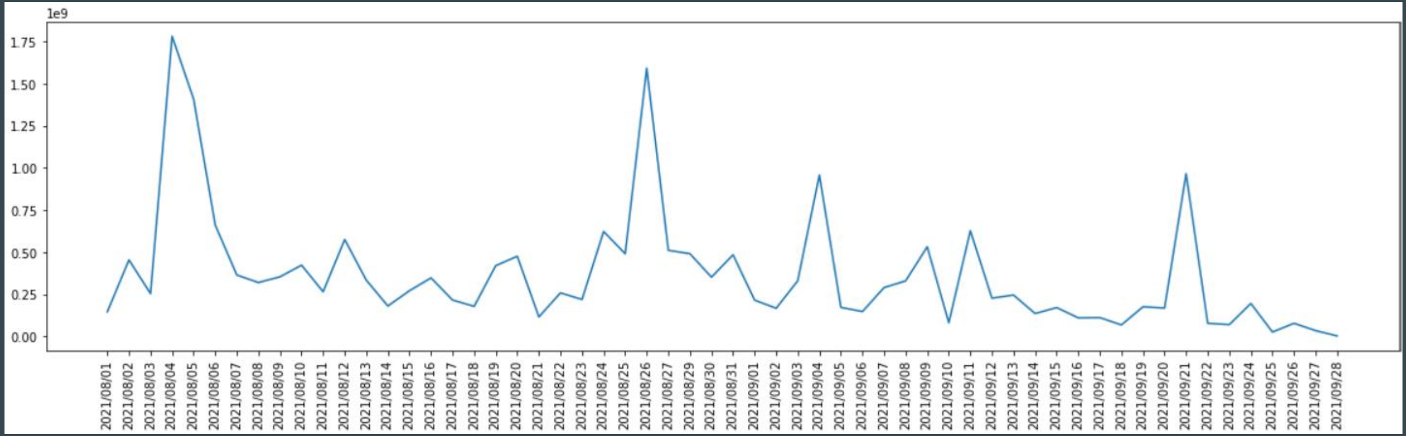
```
saham_bal = df_eda['saham_bal'].sum()
uang_bal = df_eda['uang_bal'].sum()
pend_tetapbal = df_eda['pend_tetapbal'].sum()
campuran_bal = df_eda['campuran_bal'].sum()
total = df_eda['total_balance'].sum()

print(f"saham balance: {saham_bal}")
print(f"uang balance: {uang_bal}")
print(f"pendapatan tetap balance: {pend_tetapbal}")
print(f"campuran balance: {campuran_bal}")
print(f"total: {total}")
```



Exploratory Data Analytics (EDA)

8. Trend

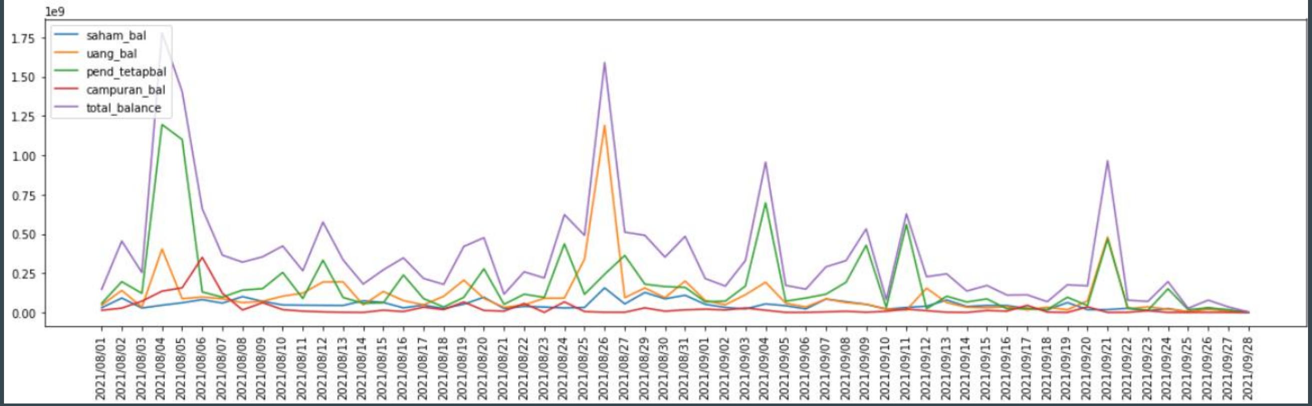
```
plt.figure(figsize =(20, 5))

plt.xticks(rotation = 90)
plt.plot(transaction_trend['trans_date'], transaction_trend['total_balance'])

plt.show()
```



Exploratory Data Analytics (EDA)

9. Trend Each Product

```
#Create line chart
plt.figure(figsize =(20, 5))
axis = transaction_trend2.columns.tolist()
#axis.head()

for x in axis[1:]:    #
  plt.plot(transaction_trend2['trans_date'], transaction_trend2[x])

plt.xticks(rotation = 90)
plt.legend(transaction_trend2.iloc[:,1:],loc = 2)

plt.show()
```



Python Milestone 2

(Segmentation and Clustering Analysis)

•••

Related Variable

In this case I want to make cluster according Total_Freq and total_balance

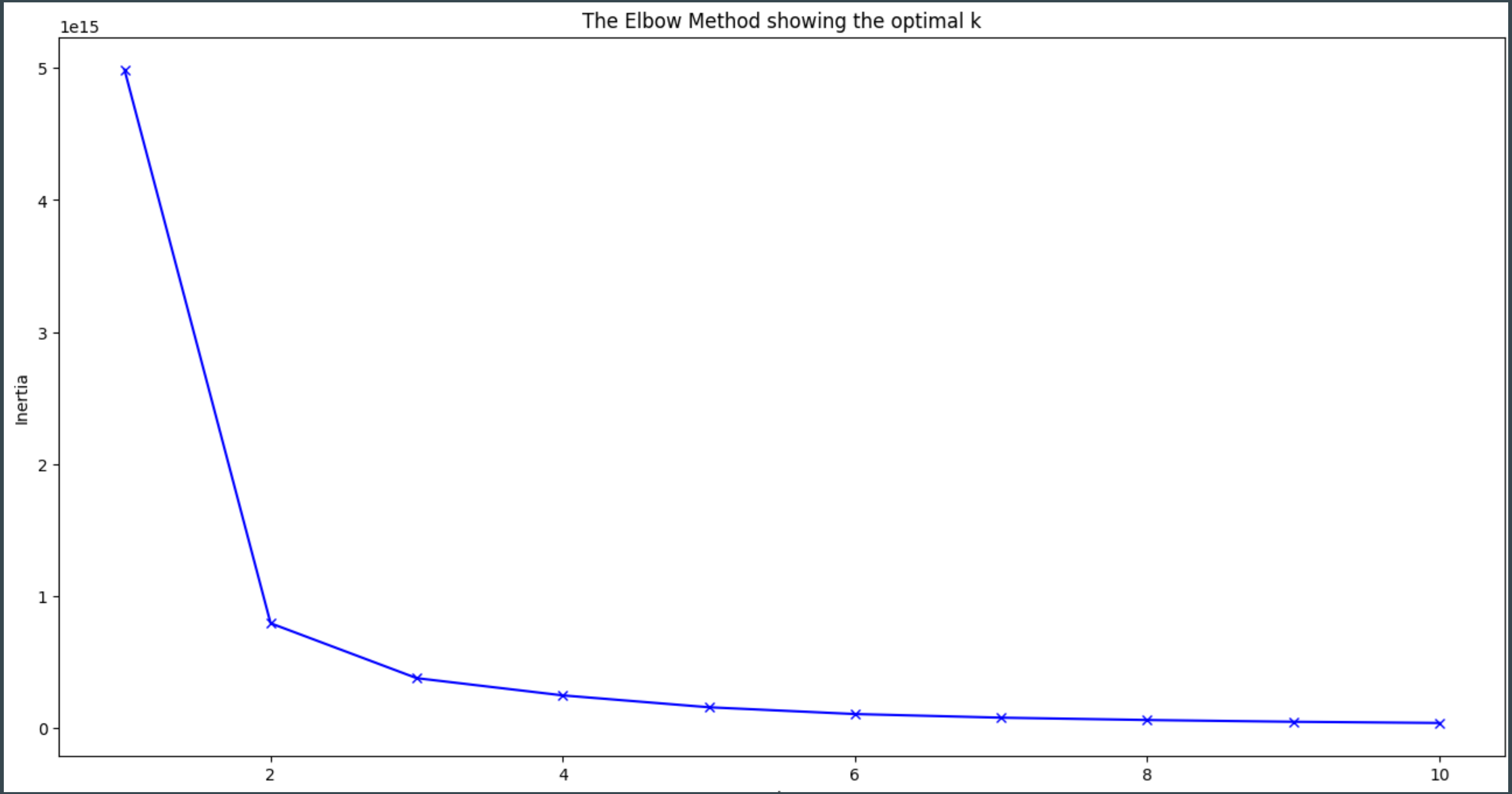
1. Total_Freq is choosen because as fund managers, we gain profit from buy/sell transaction fee. So more often costumers doing transactions, we gain more profit
2. total_balance reflect the buying power of each customers.

Some points we have to notice are it is normal when customers dont have either balance or frequencies in their account. Because investors divided into some classes

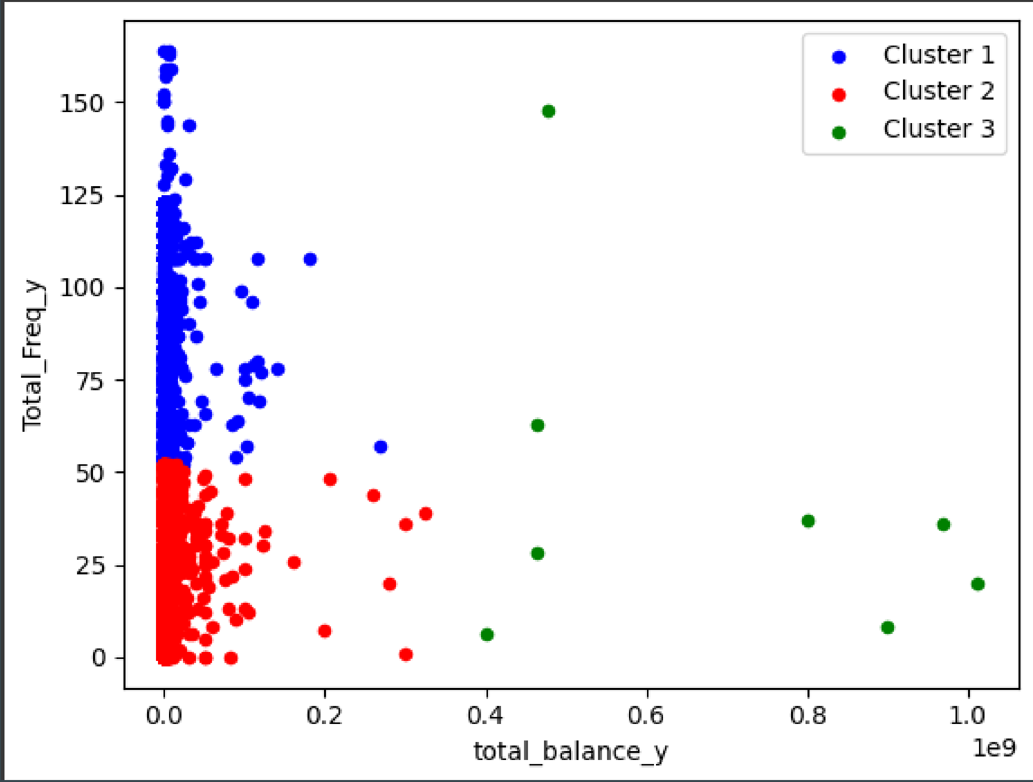
1. Short term investors = Usually high frequencies, prefer cash in the end of month/quarter (0 balance at the end of the month)
2. Long term investors = Low frequencies, making deep analysis of the market before enter positions. As soon as they enter the market, they keep their positions for years at least 1 year

Elbow Method

Colab Link



Clustering



Colab Link

Segmentation

x	0	1	2
Gender	Male	Male	Female
Age	26	27	39
Income Range	Silver	Silver	Gold
Total Frequency	20	83	43
Total Balance	1.716.676,29	2.349.707,50	685.778.338,25
Total Transaction	916.760,04	2.348.775,22	455.228.150,50
Saham Balance	444.725,37	587.351,74	6.500.000
Pasar Uang Balance	726.924,01	643.763,58	435.568.988,75
Pendapatan Tetap Balance	1.377.780,66	1.069.620,35	527.564.393
Campuran Balance	9.171.888,25	2.890.038,29	12.666.666,60

Learner investors, Contribute 0,1% of total transaction

Experience investors, high frequency of transaction. Contribute 0,5% of total transaction

Big money, long term investors, give 99% contribution of total transaction

Cluster 0 (Learner Investors)

x	0
Gender	Male
Age	26
Income Range	Silver
Total Frequency	20
Total Balance	1.716.676,29
Total Transaction	916.760,04
Saham Balance	444.725,37
Pasar Uang Balance	726.924,01
Pendapatan Tetap Balance	1.377.780,66
Campuran Balance	9.171.888,25

Recommendations :

1. Give more promotions to gain more transactions or top up on their accounts
2. Give education about investing through short videos or articles
3. Give promotions with referral because this cluster is huge in numbers

Cluster 1 (Experience Investors)

x	1
Gender	Male
Age	27
Income Range	Silver
Total Frequency	83
Total Balance	2.349.707,50
Total Transaction	2.348.775,22
Saham Balance	587.351,74
Pasar Uang Balance	643.763,58
Pendapatan Tetap Balance	1.069.620,35
Campuran Balance	2.890.038,29

Recommendations :

1. Give advance education about retirement income from investing
2. Give promotion or coupon from referral because this cluster is huge in numbers

Cluster 2 (Big Money)

x	2
Gender	Female
Age	39
Income Range	Gold
Total Frequency	43
Total Balance	685.778.338,25
Total Transaction	455.228.150,50
Saham Balance	6.500.000
Pasar Uang Balance	435.568.988,75
Pendapatan Tetap Balance	527.564.393
Campuran Balance	12.666.666,60

Recommendations :

1. Give transaction fee discount
2. Low profit sharing to fund managers

Python Milestone 3

(Python Logistic Regression Model)

Understanding Business Problem

1. Mutual Investment Company has some investment products : stocks, bond, money market, and mixed investment mutual funds
2. Company has some issues such as churn rate, and limitation of campaign budget (30%) and it give impact to the company profit
3. According to previous data, cost per campaign is Rp1.000 and the transaction fee (buy and sell) is 0,15%

Steps

Data Gathering → Code Preparation → Raw Data Prep → Cleaning Data → EDA → Logistic Regression

Code Preparation

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn import cluster
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import classification_report, confusion_matrix
```

[Colab Link]

Raw Data Preparation

user_id	registration_import_datetime	user_gender	user_age	user_income_range	referral_code_used	Total_Transaction	Saham_Freq	Pasar_Uang_Freq	Pend_Tetap_Freq
3651808	2021-09-12 13:05:04	Male	52	silver	no referral	0	11	11	11
3934978	2021-08-30 14:26:12	Male	21	silver	no referral	238604	20	20	20
3944498	2021-08-31 20:37:09	Male	17	silver	no referral	160000	19	19	19
3980156	2021-09-05 07:08:38	Male	24	silver	used referral	10000	16	16	16

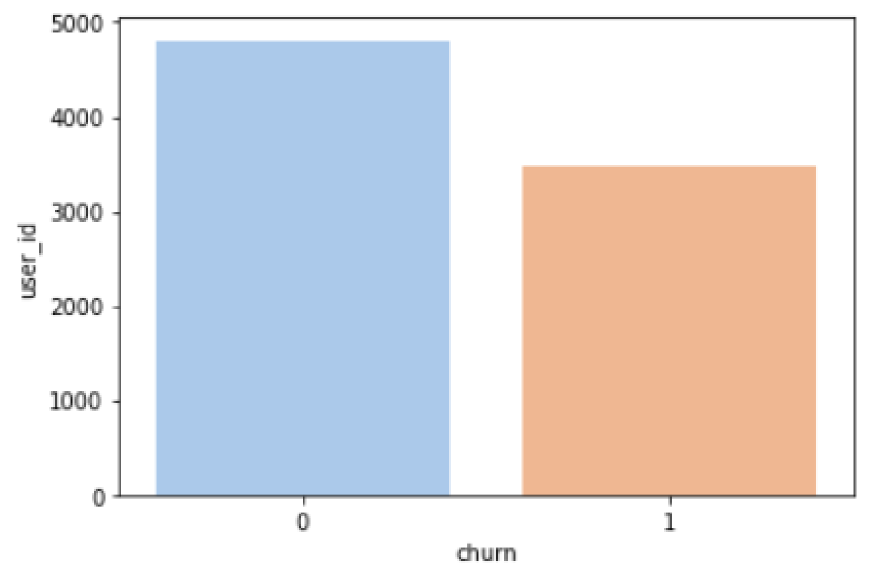
Campuran_Freq	Total_Freq	date	saham_bal	uang_bal	pend_tetapbal	campuran_bal	total_balance	churn
0	33	2021-09-30	30000.000	10000.000	60000.000	0.000	100000	1
0	60	2021-09-30	800.000	87804.000	110000.000	0.000	198604	1
0	57	2021-09-30	0.000	0.000	40000.000	0.000	40000	0
0	48	2021-09-30	0.000	0.000	0.000	0.000	0	0

Category of each variable

```
numerical_column = ['user_age','Total_Transaction','Saham_Freq','Pasar_Uang_Freq','Pend_Tetap_Freq','Campuran_Freq',
                    'Total_Freq','saham_bal','uang_bal','pend_tetapbal','campuran_bal','total_balance']
binary_categorical_column = ['user_gender','referral_code_used']
polytomous_categorical_column = ['user_income_range']
all_categorical_column = binary_categorical_column+polytomous_categorical_column
```

EDA

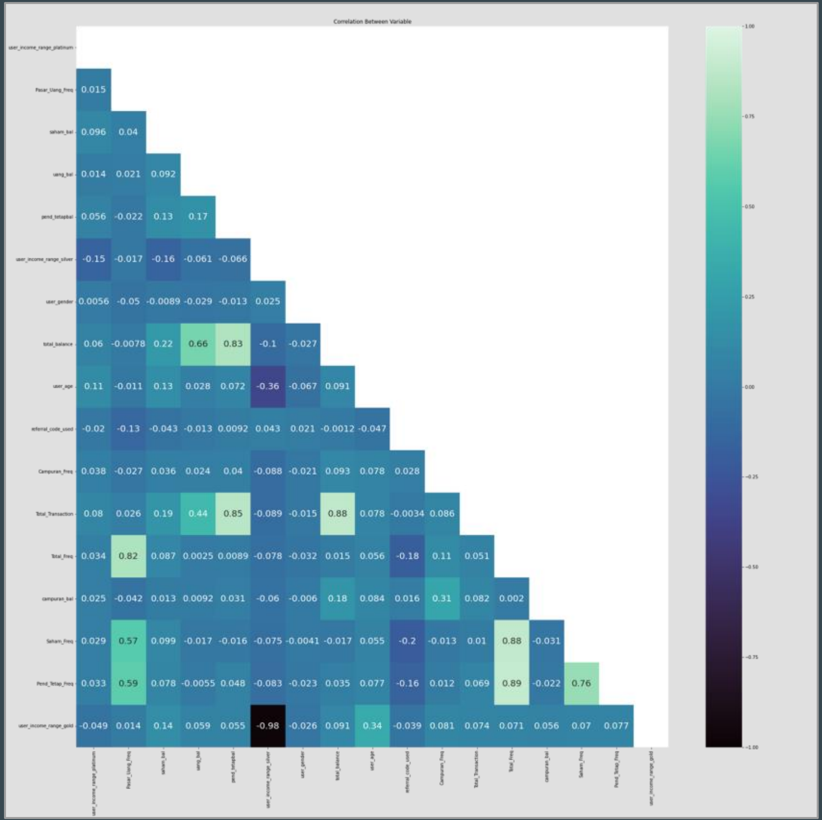
```
        user_id
churn
0          4809
1          3468
<AxesSubplot:xlabel='churn', ylabel='user_id'>
```



churn rate

3468/(4809+3468) = 0,418

Uji Korelasi dan Contoh



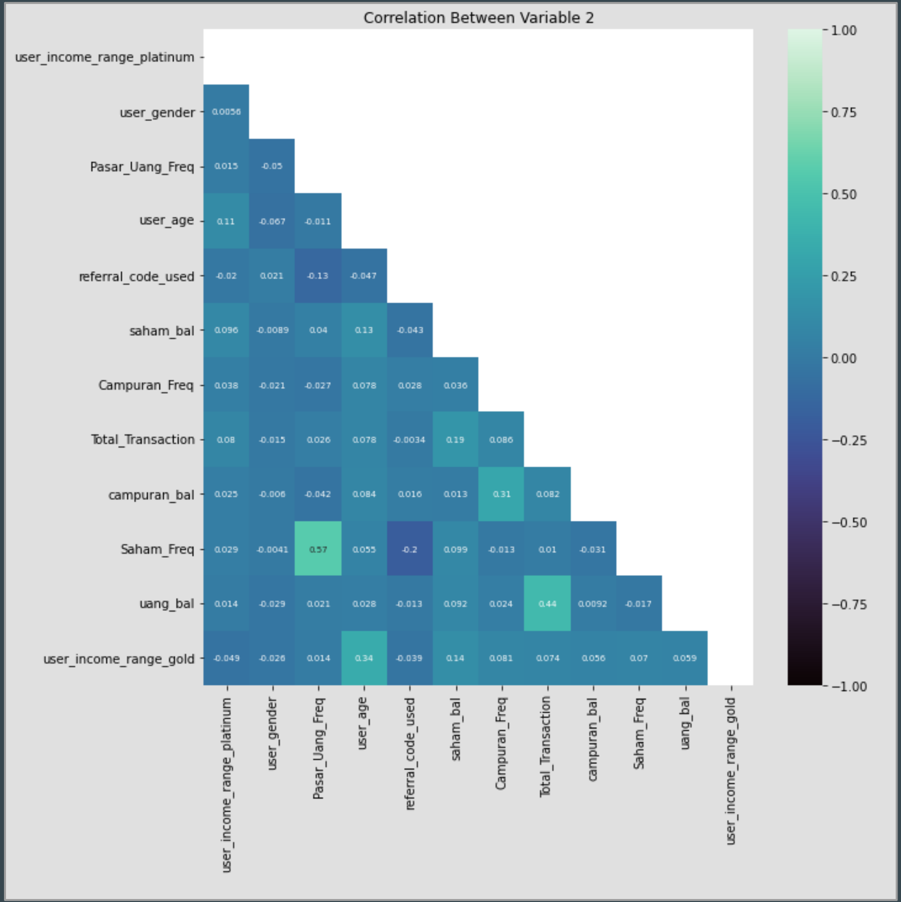
Drop High Correlation Var

```
# Create correlation matrix
corr_matrix = df_churn_new.corr().abs()

# Select upper triangle of correlation matrix
upper = corr_matrix.where(np.triu(np.ones(corr_matrix.shape), k=1).astype(np.bool))

# Find features with correlation greater than 0.7, and add into to_drop list
to_drop = [column for column in upper.columns if any(upper[column] > 0.7)]
to_drop
```

```
['Pend_Tetap_Freq',
 'Total_Freq',
 'pend_tetapbal',
 'total_balance',
 'user_income_range_silver']
```

Dropped columns

New Correlation After Drop Column



Fit Logistic Regression - Accuracy Rate

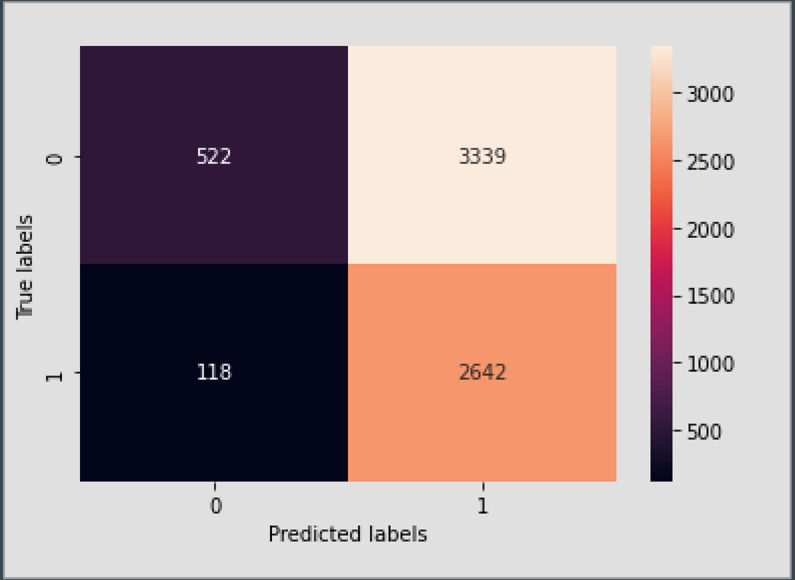
```
model = LogisticRegression(class_weight='balanced',max_iter=500)
model.fit(x_training, y_training)
```

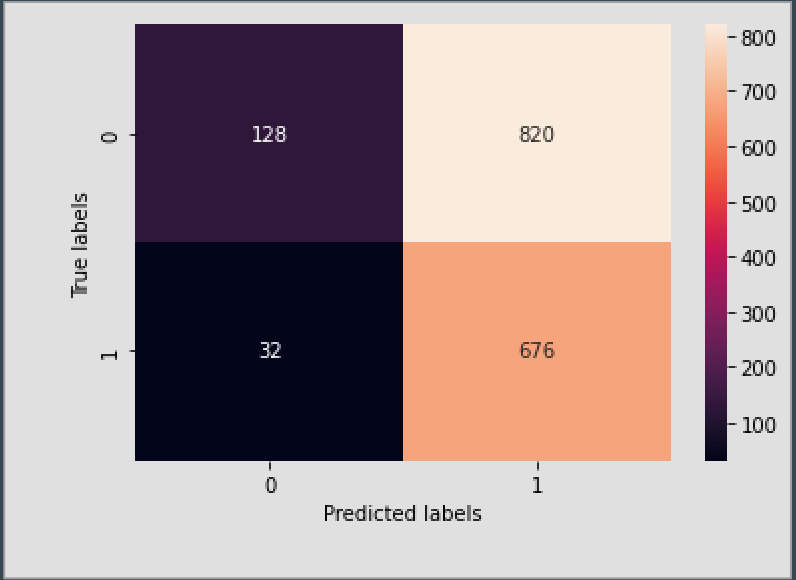
```
LogisticRegression
LogisticRegression(class_weight='balanced', max_iter=500)
```

```
# Accuracy dr prediksi model dengan data training
model.score(x_training, y_training)
```

```
0.4778734330161607
```

Confusion Matrix & Performance Test





	precision	recall	f1-score	support
Not Churn	0.82	0.14	0.23	3861
Churn	0.44	0.96	0.60	2760
accuracy			0.48	6621
macro avg	0.63	0.55	0.42	6621
weighted avg	0.66	0.48	0.39	6621

	precision	recall	f1-score	support
Not Churn	0.80	0.14	0.23	948
Churn	0.45	0.95	0.61	708
accuracy			0.49	1656
macro avg	0.63	0.54	0.42	1656
weighted avg	0.65	0.49	0.39	1656

Benefit Cost Analysis (Transaction Fee)

	user_id	Total_Transaction	fee_trans
0	3816789	100000	150.000
1	3802293	8500000	12750.000
2	3049927	149000	223.500
3	3836491	0	0.000
4	3783302	2889569	4334.354

fee_trans = 0,15% x Total_Transaction

= Rp 22.001.245,962

```
user_id              380229337833023722610378720238370993740290376 8...
Total_Transaction                                           14667497308
fee_trans                                                  22001245.962
churn                                                              1287
dtype: object
```

```
# Get 70% threshold of profit
prft70_threshold = df_tot_trans['fee_trans'].quantile(0.7)

# Get top 70% customers
prft70_customer = df_tot_trans[df_tot_trans['fee_trans'] > prft70_threshold]

# Merge with actual churn outcome
prft70_customer = prft70_customer.merge(y, left_index = True, right_index = True)
prft70_customer.sum()
```

fee_trans
Quantile 70%

Benefit Cost Analysis (Churn Customers)

	predicted_score	churn	Flag
0	0.532	0	Medium Churn
1	0.983	0	Mostly Churn
5	0.527	1	Medium Churn
8	0.519	1	Medium Churn
10	0.670	1	Mostly Churn

```
pct70_customer['churn'].value_counts()
```

```
1    1603
0     880
Name: churn, dtype: int64
```

```
def threshold_category(pct70_customer):

    if (pct70_customer['predicted_score'] <= 0.3 ):
        return 0
    elif (pct70_customer['predicted_score'] > 0.3) and (pct70_customer['predicted_score'] <= 0.6):
        return 'Medium Churn'
    else:
        return 'Mostly Churn'

pct70_customer['Flag'] = pct70_customer.apply(threshold_category, axis = 1)
pct70_customer
```

Potential Benefit per Campaign

If case transaction fee = 0,15%

Cost campaign = Rp 1000

Churn users = 1603

Potential Benefit = 22.001.245,962 - (Rp1000 x 1603)

= Rp 20.398.245,962

THANKS FOR YOUR ATTENTION