

Homework

Final Project - Stage 2



1. Data Pre-Processing

A. Data Splitting

```

1 # split train and test set
2
3 # memisahkan antara training dan test set
4 from sklearn.model_selection import train_test_split
5 data_train, data_test = train_test_split(data, test_size=0.2, random_state=42)
6
7 # menampilkan shape dari train dan test set
8 print(f'data_train: {data_train.shape}, data_test: {data_test.shape}')
```

```
data_train: (1792, 26), data_test: (448, 26)
```

Setelah dilakukan data split maka data kita telah memiliki partisi untuk test dan training dimana data_train 1792 sample dengan 26 feature dan data_test 448 sample dengan 26 feature juga.

1. Data Pre-Processing

B. Handling Missing Values

Pada data_train Terdapat 19 missing values (1.06%) dimensi sebelum (1792, 26) setelah Drop missing values – dimensi sesudah (1773, 26)

Pada data_test Terdapat 5 missing values (1.12%) dimensi sebelum (448, 26) setelah Drop missing values dimensi sesudah (443, 26)

C. Handling Duplicate Values

Pada data_train Terdapat 116 duplicate values (6.54%) dimensi sebelum (1773, 26) setelah Drop duplicate values – dimensi sesudah (1657, 26)

Pada data_test Terdapat 6 duplicate values (1.35%) dimensi sebelum (443, 26) setelah Drop duplicate values dimensi sesudah (437, 26)

```
Data Train Shape : (1792, 26)
Data Test Shape : (448, 26)
#1.Identification Missing Values
  column  missing values  percentage
0  Income           19           1.06
  column  missing values  percentage
0  Income           5           1.12
#After Missing Value Handling
Data Train Shape : (1773, 26)
Data Test Shape : (443, 26)
#2.Identification Duplicated Rows
  duplicated rows  percentage
0           116           6.54
  duplicated rows  percentage
0           6           1.35
#After Duplicated Value Handling
Data Train Shape : (1657, 26)
Data Test Shape : (437, 26)
()
```

1. Data Pre-Processing

D. Handling Outliers

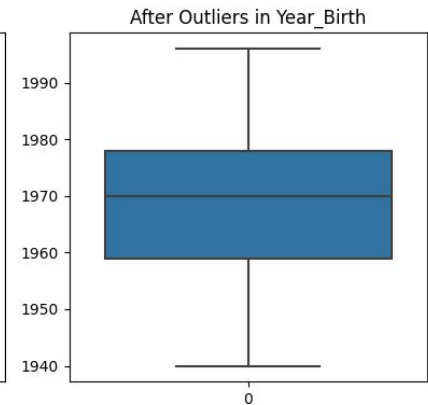
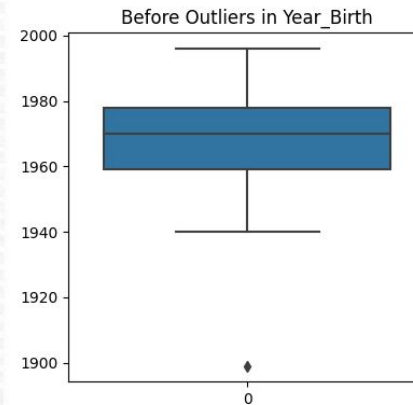
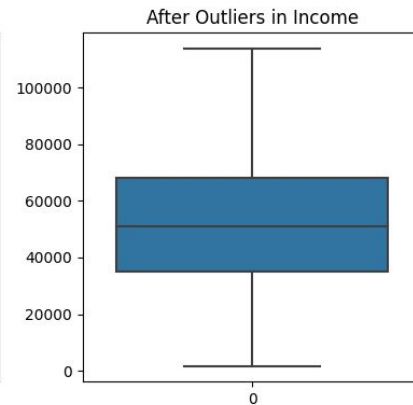
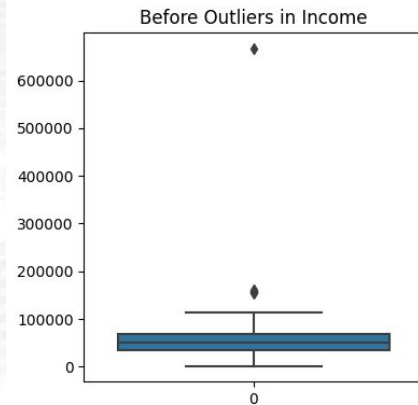
Menggunakan Z-Score threshold std = 3, pada column Income dan Year_Birth

Column Income

| | |
|------------|---------|
| Before | 1657.00 |
| After | 1650.00 |
| Outliers | 7.00 |
| % Outliers | 0.42 |

Column Year_Birth

| | |
|------------|---------|
| Before | 1650.00 |
| After | 1649.00 |
| Outliers | 1.00 |
| % Outliers | 0.06 |



2. Feature Engineering

A. Feature Extractions

```

1 # membuat feature baru berdasarkan status hubungan
2 marital = {
3     'Single': 'Not in relationship',
4     'Together': 'In relationship',
5     'Married': 'In relationship',
6     'Divorced': 'Not in relationship',
7     'Widow': 'Not in relationship',
8     'Alone': 'Not in relationship',
9     'Absurd': 'Not in relationship',
10    'YOLO': 'Not in relationship'
11 }
12 data['Relationship_Status'] = data['Marital_Status'].map(marital)
13
14 # membuat feature baru total_children dari penjumlahan feature kidhome dan teenhome
15 data['Total_Children'] = data['Kidhome'] + data['Teenhome']
16
17 # membuat feature baru berdasarkan jumlah anggota keluarga
18 def fam_size(x):
19     if x['Relationship_Status'] == 'Not in relationship':
20         result = 1 + x['Teenhome'] + x['Kidhome']
21     elif x['Relationship_Status'] == 'In relationship':
22         result = 2 + x['Teenhome'] + x['Kidhome']
23     return result
24 data['Family_Size'] = data.apply(fam_size, axis=1)

```

Beberapa feature baru yang diextract dari feature-feature sebelumnya seperti Relationship_Status, Family_Size, Customer_Lifespan, Year, Total_Purchase, Total_Spending, Total_Offers, dan lainnya. (lihat full di source code)

```

1 # membuat feature baru berdasarkan tanggal bergabung dan diasumsikan data dikumpulkan pada awal juli 2014
2 data['Customer_Lifespan'] = (pd.to_datetime('2014-07-01') - data['Dt_Customer']).dt.days
3
4 # ekstraksi feature Datetime menjadi feature baru
5 data['Year'] = data['Dt_Customer'].dt.year
6 data.drop(['Dt_Customer'],axis=1,inplace=True)
7
8 # membuat feature baru total purchase, total spending, dan total offers
9 data['Total_Purchase'] = data.apply(lambda x: x[purchase_cols[:-1]].sum(), axis=1)
10 data['Total_Spending'] = data.apply(lambda x: x[spending_cols].sum(), axis=1)
11 data['Total_Offers'] = data.apply(lambda x: x[campaign_cols[:-1]].sum(), axis=1)

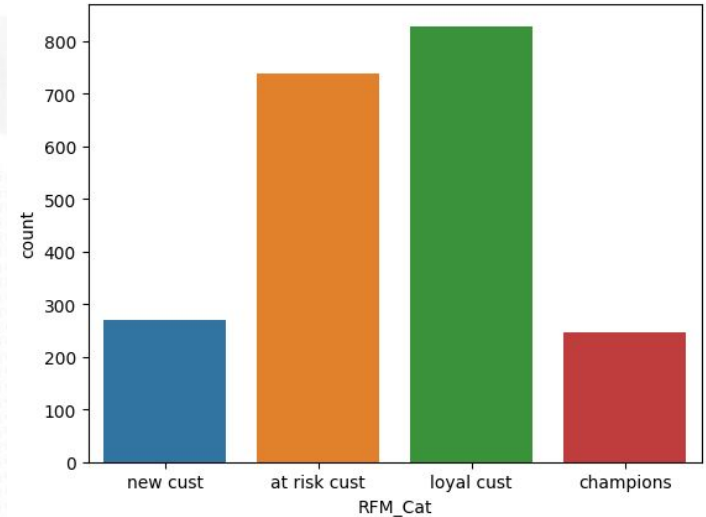
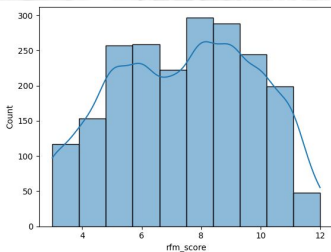
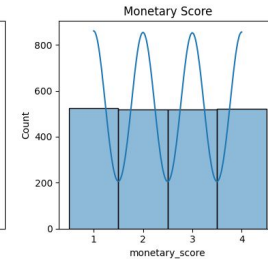
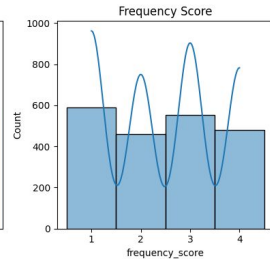
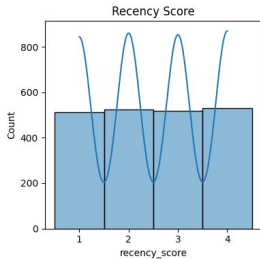
```

2. Feature Engineering

A. Feature Extractions - RFM

```
1 # membuat feature baru category rfm score (champions, loyal, at risk, new)
2 rfm = pd.DataFrame()
3 rfm['Recency'] = data['Recency']
4 rfm['Frequency'] = data['Total_Purchase']
5 rfm['Monetary'] = data['Total_Spending']
6 rfm.head()
```

```
1 # kalkulasi score berdasarkan quantile masing-masing feature
2 rfm['recency_score'] = pd.qcut(rfm['Recency'], q=[0, 0.25, 0.5, 0.75, 1], labels=[4, 3, 2, 1])
3 rfm['frequency_score'] = pd.qcut(rfm['Frequency'], q=[0, 0.25, 0.5, 0.75, 1], labels=[1, 2, 3, 4])
4 rfm['monetary_score'] = pd.qcut(rfm['Monetary'], q=[0, 0.25, 0.5, 0.75, 1], labels=[1, 2, 3, 4])
```



Melakukan segmentasi customer dengan menggunakan metode rfm. Customer dibagi menjadi 4 segment, yaitu champions, loyal, at risk, dan new customer. Berdasarkan count plot diketahui bahwa loyal customer paling banyak, champions adalah yang paling sedikit pada dataset. Lalu hasilnya digabungkan dengan dataset utama.

2. Feature Engineering

B.Feature Encoding

```
# encoding education
edu = {'Graduation': 1, 'Master': 2, 'PhD': 3}
oe_edu = OrdinalEncoder(categories=[list(edu.keys())])
data['Education'] = oe_edu.fit_transform(data[['Education']])

# encoding marital_status
marital_mapping = {'Single': 1, 'Married': 2, 'Divorced': 3}
data['Marital_Status'] = data['Marital_Status'].map(marital_mapping)

# encoding relationship_status
rel_mapping = {'Not in relationship': 0,
               'In relationship': 1}
data['Relationship_Status'] = data['Relationship_Status'].map(
    rel_mapping)

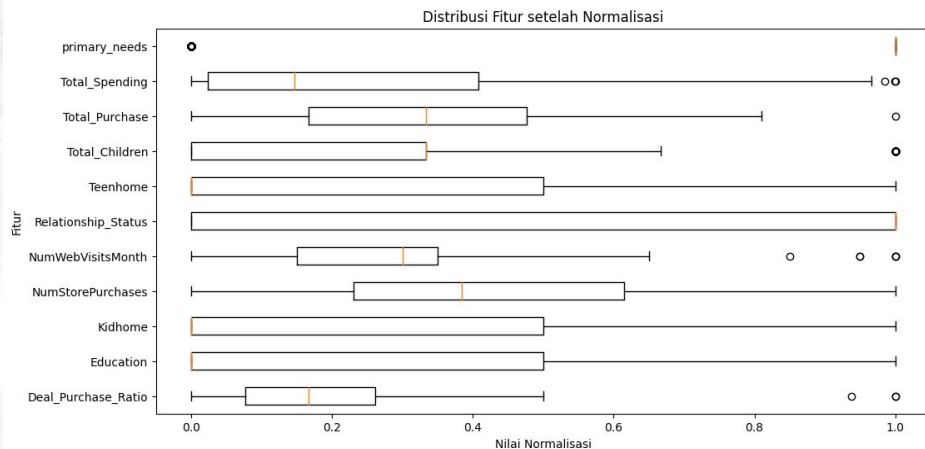
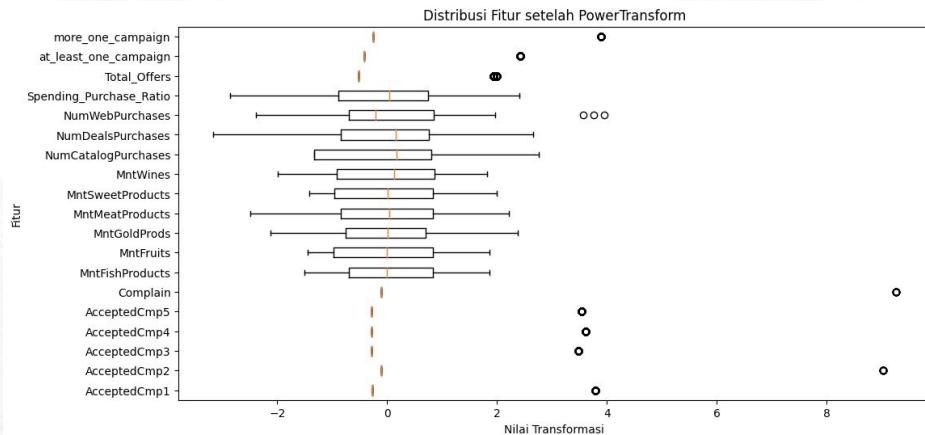
# encoding primary_needs
pr_mapping = {'primary_needs': 0,
              'secondary_needs': 1}
data['primary_needs'] = data['primary_needs'].map(pr_mapping)

# encoding rfm_cat
rfm_mapping = {'new cust': 1, 'at risk cust': 2,
               'loyal cust': 3, 'champions': 4}
oe_rfm = OrdinalEncoder(categories=[li(rfm_mapping.keys())])
data['RFM_Cat'] = oe_rfm.fit_transform(data[['RFM_Cat']])
```

Menggunakan Ordinal encoder untuk mengubah categorical feature menjadi nilai ordinal. Beberapa feature yang dilakukan encode seperti education, marital_status, relationship_status, primary_needs, dan rfm_cat

2. Feature Engineering

C. Feature Transformation



Pada dataset ini melakukan 2 jenis transformasi yaitu normalisasi dan log transformasi/power transformasi.

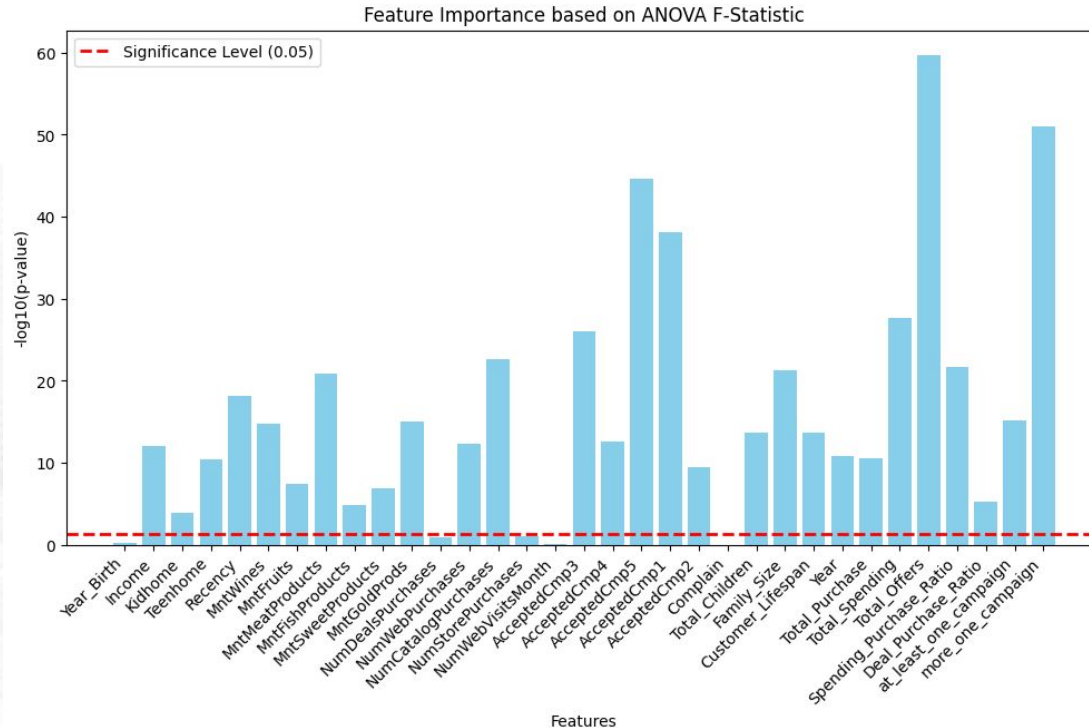
Normalisasi digunakan pada saat: $\text{mean} < \text{median} < \text{mode}$.

Transformasi digunakan pada saat: $\text{nilai skew_val} \leq -1$ atau $\text{skew_val} \geq 1$.

Library yang digunakan adalah MinMaxScaler untuk normalisasi dan PowerTransformer untuk yang transformasi log.

2. Feature Engineering

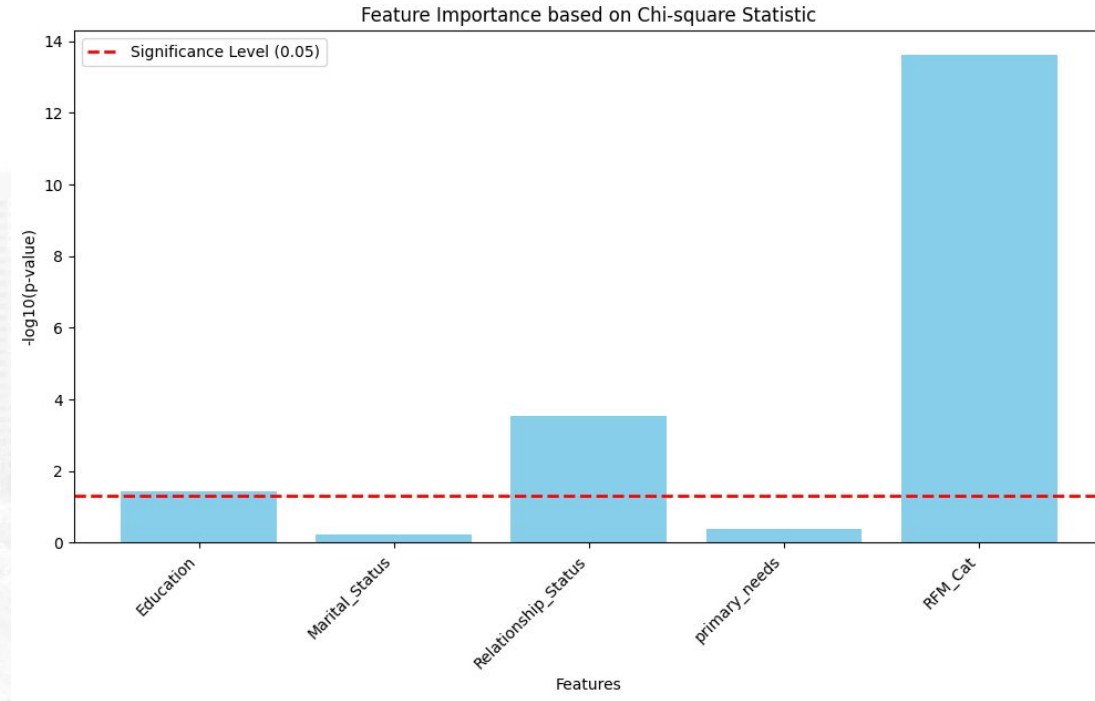
D. Feature Selection - ANOVA



Melakukan feature selection pada numerical feature dengan menggunakan metode anova, dan hasilnya didapatkan bahwa Year Birth, Num Deals Purchase, Num Web Visits Month, Num Store Purchases, Complain, tidak mampu melewati threshold p value < 0.05 .

2. Feature Engineering

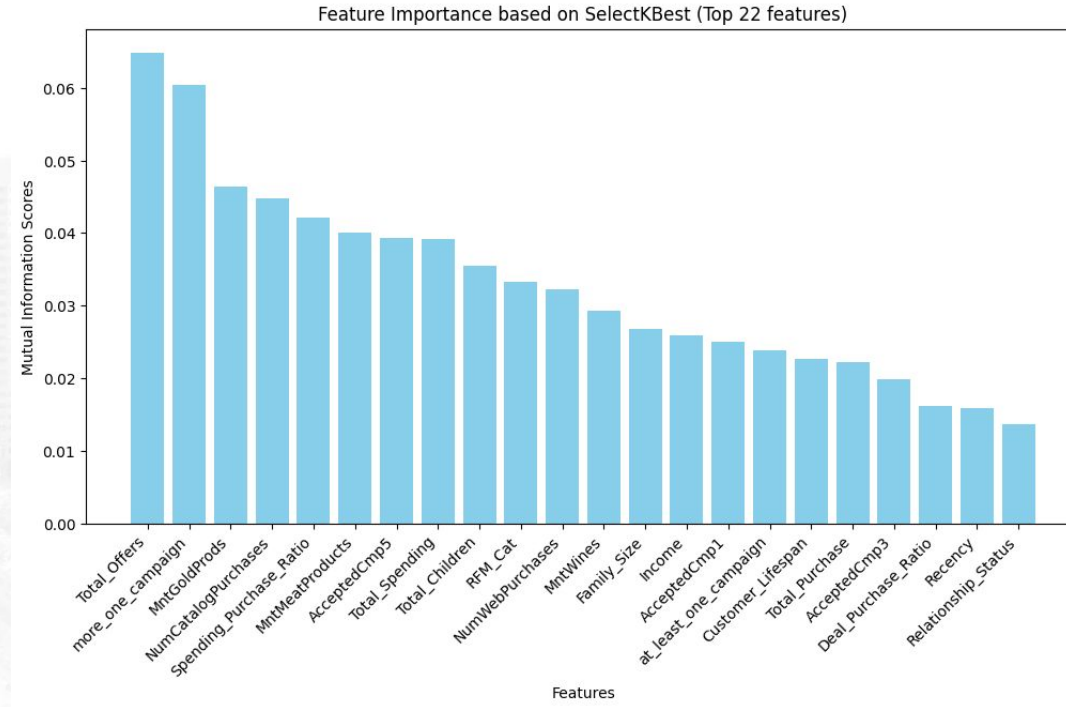
D. Feature Selection - Chi Square



Melakukan feature selection pada categorical feature dengan menggunakan metode chi-square, dan hasilnya didapatkan bahwa Marital Status dan primary needs tidak mampu melewati threshold p value < 0.05 .

2. Feature Engineering

D. Feature Selection - Mutual Info Classif



Setelah menggabungkan feature selection dari metode anova dan chi2, selanjutnya menggunakan kbest dan mutual info classification. Dan didapatkan jumlah k yang terbaik adalah 22 feature. Dimana total offers merupakan feature dengan score tertinggi, sedangkan relationship status adalah yang terendah. Selanjutnya melakukan pengecekan multicollinearity menggunakan metode VIF.

2. Feature Engineering

D. Feature Selection - VIF (Redundancy Analysis)

Avg. VIF Score: 490540.08822727663

| | Feature | VIF |
|----|-------------------------|--------------|
| 0 | Income | 6.072868e+00 |
| 1 | Kidhome | 3.017963e+00 |
| 2 | MntWines | 1.288884e+01 |
| 3 | MntMeatProducts | 1.041846e+01 |
| 4 | MntGoldProds | 2.594825e+00 |
| 5 | NumWebPurchases | 3.143823e+00 |
| 6 | NumCatalogPurchases | 5.211826e+00 |
| 7 | AcceptedCmp3 | 1.266840e+01 |
| 8 | AcceptedCmp4 | 1.280350e+01 |
| 9 | AcceptedCmp5 | 1.110481e+01 |
| 10 | AcceptedCmp1 | 9.829085e+00 |
| 11 | Total_Children | 1.088710e+02 |
| 12 | Family_Size | 5.526401e+02 |
| 13 | Customer_Lifespan | 1.217927e+00 |
| 14 | Total_Spending | 9.014238e+00 |
| 15 | Total_Offers | 5.126044e+06 |
| 16 | Spending_Purchase_Ratio | 2.008915e+01 |
| 17 | Deal_Purchase_Ratio | 3.564921e+00 |
| 18 | at_least_one_campaign | 3.810943e+06 |
| 19 | more_one_campaign | 1.854052e+06 |
| 20 | Relationship_Status | 5.471427e+01 |
| 21 | RFM_Cat | 2.922084e+00 |



Avg. VIF Score: 3.4913412803840322

| | Feature | VIF |
|----|---------------------|----------|
| 0 | Recency | 3.042867 |
| 1 | MntWines | 5.538016 |
| 2 | MntMeatProducts | 5.603630 |
| 3 | MntGoldProds | 2.004252 |
| 4 | NumWebPurchases | 2.505467 |
| 5 | NumCatalogPurchases | 5.007321 |
| 6 | AcceptedCmp3 | 1.842999 |
| 7 | AcceptedCmp5 | 1.743514 |
| 8 | AcceptedCmp1 | 1.526316 |
| 9 | Total_Children | 3.785204 |
| 10 | Customer_Lifespan | 4.307159 |
| 11 | Total_Offers | 3.193541 |
| 12 | Relationship_Status | 2.818932 |
| 13 | RFM_Cat | 5.959560 |



Avg. VIF Score: 3.56880687225212

| | Feature | VIF |
|----|---------------------|----------|
| 0 | Recency | 3.054826 |
| 1 | MntWines | 5.980497 |
| 2 | MntMeatProducts | 6.564739 |
| 3 | MntGoldProds | 2.090503 |
| 4 | NumWebPurchases | 2.521329 |
| 5 | NumCatalogPurchases | 5.060327 |
| 6 | AcceptedCmp3 | 2.870500 |
| 7 | AcceptedCmp5 | 1.830245 |
| 8 | AcceptedCmp1 | 1.622926 |
| 9 | Total_Children | 3.900620 |
| 10 | Customer_Lifespan | 4.309492 |
| 11 | Total_Offers | 6.041355 |
| 12 | Relationship_Status | 2.825440 |
| 13 | RFM_Cat | 6.032342 |
| 14 | AcceptedCmp2 | 1.124760 |
| 15 | AcceptedCmp4 | 2.481471 |
| 16 | MntFruits | 2.358346 |

Pada Metode VIF, kami melakukan secara manual untuk mengurangi atau menambah feature dimana setiap feature tidak boleh melebihi threshold yakni >10 . Dan mencari nilai rata-rata optimal 2-5. Maka didapatkan 17 features, yang diantaranya terdapat features yang dihasilkan dari extraction seperti RFM_Cat, Total_Offers, Relationship_Status, dan Total_Children.

2. Feature Engineering

E. Feature Imbalance

```

1 # melakukan imbalance handling pada target
2 from imblearn.over_sampling import SMOTE
3
4 # menampilkan jumlah kelas sebelum oversampling
5 print("Jumlah kelas sebelum oversampling:")
6 print("Kelas 0:", sum(y_train == 0))
7 print("Kelas 1:", sum(y_train == 1))
8
9 # melakukan oversampling dengan SMOTE
10 smote = SMOTE(random_state=42)
11 X_resampled, y_resampled = smote.fit_resample(X_train, y_train)
12
13 # menampilkan jumlah kelas setelah oversampling
14 print("\nJumlah kelas setelah oversampling:")
15 print("Kelas 0:", sum(y_resampled == 0))
16 print("Kelas 1:", sum(y_resampled == 1))

```

```

Jumlah kelas sebelum oversampling:
Kelas 0: 1397
Kelas 1: 251

```

```



Jumlah kelas setelah oversampling:
Kelas 0: 1397
Kelas 1: 1397






```

Menggunakan metode oversampling pada library SMOTE. Dimana feature target dengan values 1, awalnya berjumlah 251 menjadi 1397.


5. GIT





<https://github.com/hilmanman92/market-insider/tree/staging>




  hilmanman92 / market-insider



|  |  |  |  | 


[Code](#) | [Issues](#) | [Pull requests](#) | [Actions](#) | [Projects](#) | [Wiki](#) | [Security](#) | [Insights](#) | [Settings](#)


 **market-insider** Public



 Pin |  Unwatch ¹ |  Fork ⁰ |  Star ⁰


 staging |  3 Branches |  0 Tags

|  Add file |  Code

This branch is 7 commits ahead of, 5 commits behind `master` |  Contribute

 hilmanman92 | Edit Feature Selection (Mutual Info Classif) | cc86a35 · 2 minutes ago | 7 Commits

| | | |
|---|--|---------------|
|  Final_Project_Marketing_Campaign_(Staging).i... | Edit Feature Selection (Mutual Info Classif) | 2 minutes ago |
|  README.md | Initial commit | 2 weeks ago |

 README

market-insider

Final Project of Market Insider

About

Final Project of Market Insider

 Readme

 Activity

 0 stars

 1 watching

 0 forks

Releases

No releases published

[Create a new release](#)

Packages

No packages published

[Publish your first package](#)

Languages

 Jupyter Notebook 100.0%

 © 2023 GitHub, Inc. | [Terms](#) | [Privacy](#) | [Security](#) | [Status](#) | [Docs](#) | [Contact](#) | [Manage cookies](#) | [Do not share my personal information](#)