Rakamin Academy

HOME CREDIT INDONESIA

Data Scientist

ALGORITMA LOGISTIC REGRESSION & XGBOOST

MOHAMAD ARIF SOFYAN
LINKEDIN : https://www.linkedin.com/in/mohamadarifsofyan/
GITHUB : https://github.com/arifsofyan004

IMPORT DATA

df_application_train

```
[ ] df_application_train = pd.read_csv('/content/home-credit-default-risk/application_train.csv')
    df_application_train.head()
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	FLAG_OWN_REALTY	CNT_CHILDREN	AMT_INCOME_TOTAL	AMT_CREDIT	AMT_ANNUITY	AMT_GOODS_PRICE	NAME_TYPE_SUITE	NAME_INCOME_TYPE	NAME_EDUCATI
0	100002	1	Cash loans	M	N	Y	0	202500.0	406597.5	24700.5	351000.0	Unaccompanied	Working	Secondary / s
1	100003	0	Cash loans	F	N	N	0	270000.0	1293502.5	35698.5	1129500.0	Family	State servant	Higher e
2	100004	0	Revolving loans	M	Y	Y	0	67500.0	135000.0	6750.0	135000.0	Unaccompanied	Working	Secondary / s
3	100006	0	Cash loans	F	N	Y	0	135000.0	312682.5	29686.5	297000.0	Unaccompanied	Working	Secondary / s
4	100007	0	Cash loans	M	N	Y	0	121500.0	513000.0	21865.5	513000.0	Unaccompanied	Working	Secondary / s

```
path_bureau = '/content/home-credit-default-risk/bureau.csv'
path_bureau_balance = '/content/home-credit-default-risk/bureau_balance.csv'
path_POS_CASH_balance = '/content/home-credit-default-risk/POS_CASH_balance.csv'
path_credit_card_balance = '/content/home-credit-default-risk/credit_card_balance.csv'
path_previous_application = '/content/home-credit-default-risk/previous_application.csv'
path_installments_payments = '/content/home-credit-default-risk/installments_payments.csv'
```

```
df_bureau = pd.read_csv(path_bureau)
df_bureau_balance = pd.read_csv(path_bureau_balance)
df_POS_CASH_balance = pd.read_csv(path_POS_CASH_balance)
df_credit_card_balance = pd.read_csv(path_credit_card_balance)
df_previous_application = pd.read_csv(path_previous_application)
df_installments_payments = pd.read_csv(path_installments_payments)

print('Ukuran data POS_CASH_balance:', df_POS_CASH_balance.shape)
print('Ukuran data bureau_balance:', df_bureau_balance.shape)
print('Ukuran data previous_application:', df_previous_application.shape)
print('Ukuran data installments_payments:', df_installments_payments.shape)
print('Ukuran data credit_card_balance:', df_credit_card_balance.shape)
print('Ukuran data bureau:', df_bureau.shape)
```

```
Ukuran data POS_CASH_balance: (10001358, 8)
Ukuran data bureau_balance: (27299925, 3)
Ukuran data previous_application: (1670214, 37)
Ukuran data installments_payments: (13605401, 8)
Ukuran data credit_card_balance: (3840312, 23)
Ukuran data bureau: (1716428, 17)
```

- Isi data aplikasi pelatihan

Serta ukuran Data Lainya

- Ukuran data POS_CASH_balance: (10001358, 8)
- Ukuran data bureau_balance: (27299925, 3)
- Ukuran data previous_application: (1670214, 37)
- Ukuran data installments_payments: (13605401, 8)
- Ukuran data credit_card_balance: (3840312, 23)
- Ukuran data bureau: (1716428, 17)

Data Understanding

Menjelajahi data yang akan kita analisis lebih dalam.

- application_train memiliki **307.511 entri dan 122 fitur**.

Fitur application_train

- SK_ID_CURR memiliki 307.511 nomor identifikasi unik,
- TARGET adalah variabel target dengan dua nilai (default atau tidak),
- CODE_GENDER memiliki tiga kategori, dan seterusnya untuk fitur lainnya.

Informasi ini memberikan gambaran awal tentang kompleksitas dataset dan variasi fitur yang akan kita teliti dalam analisis berikutnya.

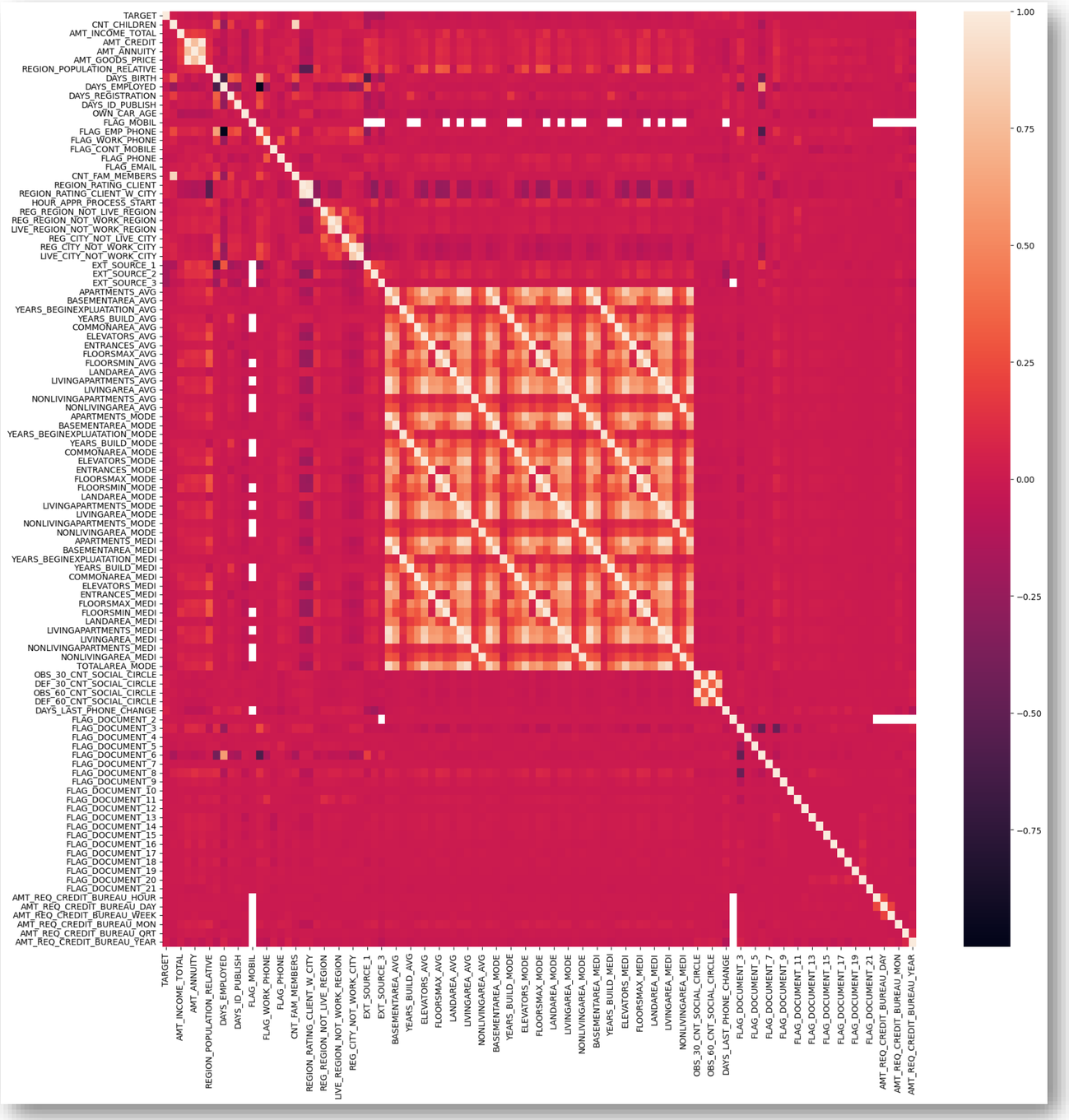
```
[ ]  print('Ukuran data application_train:', df_application_train.shape)

     Ukuran data application_train: (307511, 122)

     df_application_train.nunique()

     SK_ID_CURR                     307511
     TARGET                              2
     NAME_CONTRACT_TYPE                  2
     CODE_GENDER                         3
     FLAG_OWN_CAR                        2
     FLAG_OWN_REALTY                     2
     CNT_CHILDREN                       15
     AMT_INCOME_TOTAL                 2548
     AMT_CREDIT                       5603
     AMT_ANNUITY                     13672
     AMT_GOODS_PRICE                  1002
     NAME_TYPE_SUITE                     7
     NAME_INCOME_TYPE                    8
     NAME_EDUCATION_TYPE                 5
     NAME_FAMILY_STATUS                  6
     NAME_HOUSING_TYPE                   6
     REGION_POPULATION_RELATIVE         81
     DAYS_BIRTH                      17460
     DAYS_EMPLOYED                   12574
     DAYS_REGISTRATION               15688
     DAYS_ID_PUBLISH                  6168
     OWN_CAR_AGE                        62
     FLAG_MOBIL                          2
     FLAG_EMP_PHONE                      2
     FLAG_WORK_PHONE                     2
     FLAG_CONT_MOBILE                    2
     FLAG_PHONE                          2
     FLAG_EMAIL                          2
     OCCUPATION_TYPE                    18
```

Pemilihan Feature

- Jika terdapat pasangan fitur dengan korelasi yang tinggi, salah satunya akan dipilih. Tidak ada standar pasti untuk menentukan nilai korelasi yang dianggap tinggi, tetapi biasanya angka 0.7 digunakan sebagai acuan.

- Fitur yang sangat didominasi oleh salah satu nilai saja akan dibuang pada tahap ini.

- Menghapus kolom yang memiliki persentase nilai yang hilang yang cukup tinggi, yang mungkin sulit untuk diimputasi dengan tepat atau dapat mengganggu kinerja model.



Feature Enginering

Konversi DAYS_BIRTH menjadi usia dalam tahun

```
[ ]  df_application_train['AGE'] = round(-df_application_train['DAYS_BIRTH'] / 365, 1)
```

Konversi DAYS_EMPLOYED menjadi tahun bekerja

```
[ ]  df_application_train['YEARS_EMPLOYED'] = round(-df_application_train['DAYS_EMPLOYED'] / 365, 1)
```

Membuat fitur baru yang lebih informatif dari data yang sudah ada

- Persentase kredit terhadap pendapatan.
- Total permintaan kredit dari berbagai aspek.
- Konversi DAYS_BIRTH menjadi usia dalam tahun
- Konversi DAYS_EMPLOYED menjadi tahun bekerja

```
df_application_train['CREDIT_INCOME_PERCENT'] = df_application_train['AMT_CREDIT'] / df_application_train['AMT_INCOME_TOTAL']
df_application_train['CREDIT_INCOME_PERCENT'].head()
```

```
0    2.007889
1    4.790750
2    2.000000
3    2.316167
4    4.222222
Name: CREDIT_INCOME_PERCENT, dtype: float64
```

```
df_application_train['TOTAL_REQ_CREDIT'] = df_application_train[['AMT_REQ_CREDIT_BUREAU_HOUR', 'AMT_REQ_CREDIT_BUREAU_DAY',
                                                                 'AMT_REQ_CREDIT_BUREAU_WEEK', 'AMT_REQ_CREDIT_BUREAU_MON',
                                                                 'AMT_REQ_CREDIT_BUREAU_QRT', 'AMT_REQ_CREDIT_BUREAU_YEAR']].sum(axis=1)
df_application_train['TOTAL_REQ_CREDIT'].head()
```

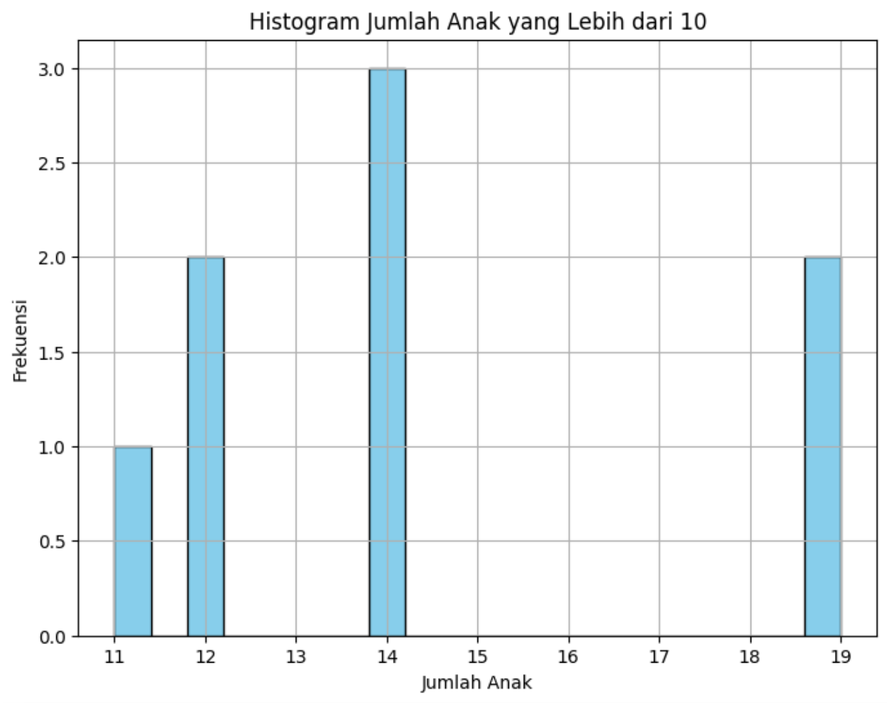
```
0    1.0
1    0.0
2    0.0
3    0.0
4    0.0
Name: TOTAL_REQ_CREDIT, dtype: float64
```

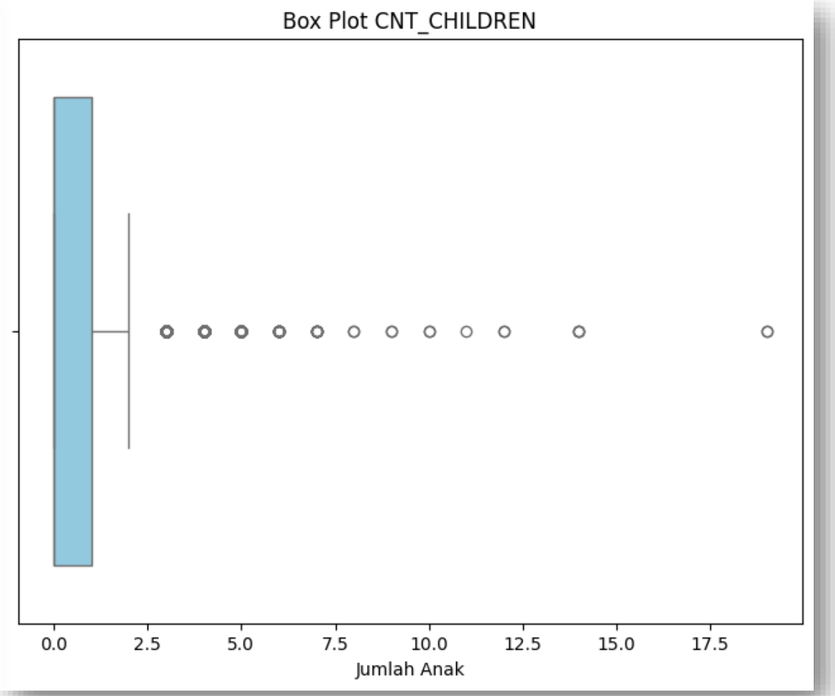
Exploratory Data Analysis

```
[ ] df_application_train.describe()
```

	TARGET	CNT_CHILDREN	REGION_POPULATION_RELATIVE	DAYS_REGISTRATION	DAYS_ID_PUBLISH	FLAG_MOBIL	FLAG_WORK_PHONE	FLAG_CONT_MOBILE	FLAG_PHONE	FLAG_EMAIL	REGION_RATING_CLIENT	HOUR_AP
count	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	307511.000000	
mean	0.080729	0.417052	0.020868	-4986.120328	-2994.202373	0.999997	0.199368	0.998133	0.281066	0.056720	2.052463	
std	0.272419	0.722121	0.013831	3522.886321	1509.450419	0.001803	0.399526	0.043164	0.449521	0.231307	0.509034	
min	0.000000	0.000000	0.000290	-24672.000000	-7197.000000	0.000000	0.000000	0.000000	0.000000	0.000000	1.000000	
25%	0.000000	0.000000	0.010006	-7479.500000	-4299.000000	1.000000	0.000000	1.000000	0.000000	0.000000	2.000000	
50%	0.000000	0.000000	0.018850	-4504.000000	-3254.000000	1.000000	0.000000	1.000000	0.000000	0.000000	2.000000	
75%	0.000000	1.000000	0.028663	-2010.000000	-1720.000000	1.000000	0.000000	1.000000	1.000000	0.000000	2.000000	
max	1.000000	19.000000	0.072508	0.000000	0.000000	1.000000	1.000000	1.000000	1.000000	1.000000	3.000000	

- CNT_CHILDREN: Nilai maksimum (19) terlihat agak aneh. Ini bisa menjadi indikasi outlier atau data yang tidak biasa.





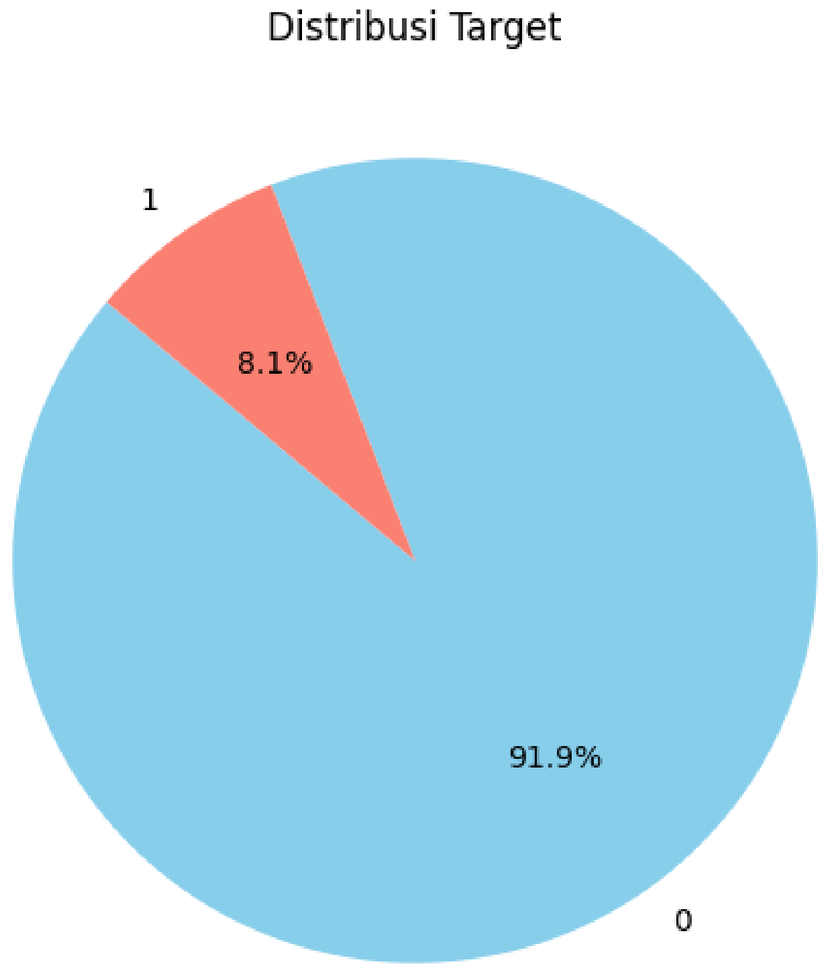
- **Menghapus jumlah anak lebih dari 10 (Outlier)**

Ekplorasi Variabel Target

Variabel target dalam dataset menunjukkan ketidakseimbangan yang signifikan:

- Kategori 1: 8.1%
- Kategori 0: 91.9%

Dengan persentase 8.1% untuk kategori 1 dan 91.9% untuk kategori 0, terdapat ketidakseimbangan yang mencolok antara pelamar yang gagal membayar kredit dengan yang berhasil



Dataset Splitting

```
[ ]  from sklearn.model_selection import train_test_split

     X = df_application_train.drop(columns=['TARGET'])
     y = df_application_train['TARGET']

     X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, stratify=y)
     X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((246002, 53), (61501, 53), (246002,), (61501,))
```

Untuk melatih model prediksi dengan data, kita membagi dataset menjadi data pelatihan dan data uji.

- Data pelatihan terdiri dari 246.002 entri, dan data uji terdiri dari 61.501 entri.
- Setiap data terdiri dari 53 fitur.
- Variabel target (y) juga telah dibagi secara proporsional antara data pelatihan dan data uji.

Imputasi, Skalasi dan Encoding

```
[ ]  # Tentukan kolom numerik dan kategorikal
     numerical_cols = ['CNT_CHILDREN', 'REGION_POPULATION_RELATIVE', 'DAYS_REGISTRATION', 'DAYS_ID_PUBLISH',
                       'FLAG_MOBIL', 'FLAG_WORK_PHONE', 'FLAG_CONT_MOBILE', 'FLAG_PHONE', 'FLAG_EMAIL',
                       'REGION_RATING_CLIENT', 'HOUR_APPR_PROCESS_START', 'REG_REGION_NOT_LIVE_REGION',
                       'REG_REGION_NOT_WORK_REGION', 'REG_CITY_NOT_LIVE_CITY', 'REG_CITY_NOT_WORK_CITY',
                       'EXT_SOURCE_2', 'EXT_SOURCE_3', 'OBS_30_CNT_SOCIAL_CIRCLE', 'DEF_30_CNT_SOCIAL_CIRCLE',
                       'DAYS_LAST_PHONE_CHANGE', 'FLAG_DOCUMENT_2', 'FLAG_DOCUMENT_3', 'FLAG_DOCUMENT_4',
                       'FLAG_DOCUMENT_5', 'FLAG_DOCUMENT_6', 'FLAG_DOCUMENT_7', 'FLAG_DOCUMENT_8', 'FLAG_DOCUMENT_9',
                       'FLAG_DOCUMENT_10', 'FLAG_DOCUMENT_11', 'FLAG_DOCUMENT_12', 'FLAG_DOCUMENT_13', 'FLAG_DOCUMENT_14',
                       'FLAG_DOCUMENT_15', 'FLAG_DOCUMENT_16', 'FLAG_DOCUMENT_17', 'FLAG_DOCUMENT_18', 'FLAG_DOCUMENT_19',
                       'FLAG_DOCUMENT_20', 'FLAG_DOCUMENT_21', 'CREDIT_INCOME_PERCENT', 'TOTAL_REQ_CREDIT', 'AGE',
                       'YEARS_EMPLOYED']

     categorical_cols = ['CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY', 'NAME_INCOME_TYPE', 'NAME_EDUCATION_TYPE',
                         'NAME_FAMILY_STATUS', 'OCCUPATION_TYPE', 'WEEKDAY_APPR_PROCESS_START', 'ORGANIZATION_TYPE']
```

```
[ ]  numerical_transformer = Pipeline(steps=[
         ('imputer', SimpleImputer(strategy='median')),
         ('scaler', StandardScaler())
     ])

     categorical_transformer = Pipeline(steps=[
         ('imputer', SimpleImputer(strategy='most_frequent')),
         ('onehot', OneHotEncoder(handle_unknown='ignore'))
     ])

     preprocessor = ColumnTransformer(
         transformers=[
             ('num', numerical_transformer, numerical_cols),
             ('cat', categorical_transformer, categorical_cols)
         ])
```

- Imputasi Missing Value: Untuk data numerik, nilai yang hilang diisi dengan median, sementara untuk data kategorikal, nilai yang hilang diisi dengan nilai yang paling sering muncul.
- Skalasi: Data numerik telah dinormalisasi menggunakan metode StandardScaler untuk memastikan rentang nilai yang konsisten.
- Encoding: Data kategorikal telah diubah menjadi bentuk numerik menggunakan OneHotEncoder untuk memungkinkan penggunaan dalam model machine learning.

Modeling Logistic Regression

```python
logistic_model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', LogisticRegression(solver='liblinear', max_iter=1000, penalty='l2'))
])

# Menambahkan parameter yang akan diuji
param_space_lr = {
    "classifier__C": np.logspace(-3, 3, 7),
    "classifier__fit_intercept": [True, False]
}

model_lr = RandomizedSearchCV(logistic_model, param_distributions=param_space_lr, n_iter=10, cv=3, random_state=42)

model_lr.fit(X_train, y_train)

# Tampilkan hasil regresi logistik
print("Best Parameters (Logistic Regression):", model_lr.best_params_)
print("Training Accuracy (Logistic Regression):", model_lr.score(X_train, y_train))
print("Model Best Score (Logistic Regression):", model_lr.best_score_)
print("Test Accuracy (Logistic Regression):", model_lr.score(X_test, y_test))
```

```
Best Parameters (Logistic Regression): {'classifier__fit_intercept': False, 'classifier__C': 0.001}
Training Accuracy (Logistic Regression): 0.91925675401013
Model Best Score (Logistic Regression): 0.9192648842870547
Test Accuracy (Logistic Regression): 0.9193509048633356
```

Hasil dari proses ini adalah sebagai berikut:

- Parameter Terbaik: 'fit_intercept': False, 'C': 0.001
- Akurasi Pelatihan: 91.93%
- Skor Terbaik Model: 91.93%
- Akurasi Uji: 91.94%

- Pemodelan: Model Regresi Logistik dibangun dengan solver 'liblinear', max_iter 1000, dan menggunakan regularisasi L2.

- Pemilihan Parameter: Kami menggunakan RandomizedSearchCV untuk mencari parameter terbaik, dengan rentang yang ditentukan untuk parameter C dan fit_intercept.

Modeling XGBoost

```
xgb_model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', XGBClassifier())
])

# Menambahkan parameter yang akan diuji
param_space_xgb = {
    'classifier__n_estimators': [100, 500, 1000],
    'classifier__max_depth': [3, 5, 7],
}

model_xgb = RandomizedSearchCV(xgb_model, param_distributions=param_space_xgb, n_iter=10, cv=3, random_state=42)

model_xgb.fit(X_train, y_train)

# Tampilkan hasil XGBoost
print("Best Parameters (XGBoost):", model_xgb.best_params_)
print("Training Accuracy (XGBoost):", model_xgb.score(X_train, y_train))
print("Model Best Score (XGBoost):", model_xgb.best_score_)
print("Test Accuracy (XGBoost):", model_xgb.score(X_test, y_test))
```

```
/usr/local/lib/python3.10/dist-packages/sklearn/model_selection/_search.py:305: UserWarning: The total space of para
  warnings.warn(
Best Parameters (XGBoost): {'classifier__n_estimators': 100, 'classifier__max_depth': 3}
Training Accuracy (XGBoost): 0.9197160998691067
Model Best Score (XGBoost): 0.9192608188002467
Test Accuracy (XGBoost): 0.9194809840490399
```

Hasil dari eksplorasi parameter:

Parameter Terbaik: 'n_estimators': 100, 'max_depth': 3
Akurasi Pelatihan: 91.97%
Skor Terbaik Model: 91.93%
Akurasi Uji: 91.95%

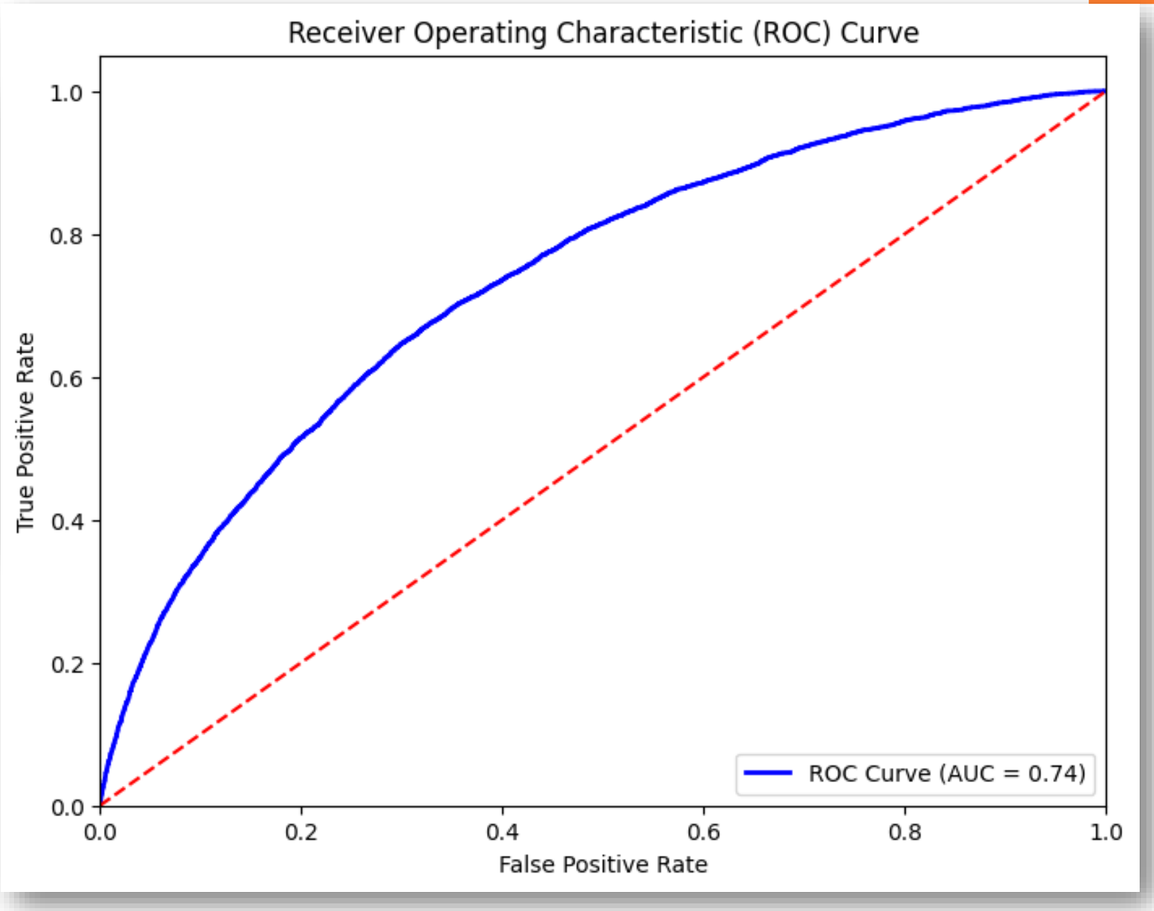
Melalui RandomizedSearchCV, kami mencari parameter terbaik untuk XGBoost.
Parameter yang kami eksplorasi adalah jumlah estimator (n_estimators) dan kedalaman maksimum (max_depth).

Evaluation Logistic Regression

Classification Report:

	precision	recall	f1-score	support
0	0.92	1.00	0.96	56536
1	0.55	0.01	0.01	4965
accuracy			0.92	61501
macro avg	0.74	0.50	0.48	61501
weighted avg	0.89	0.92	0.88	61501

Dari laporan klasifikasi, meskipun akurasi tinggi, model memiliki kinerja yang buruk dalam mengklasifikasikan kelas minoritas (kelas 1).



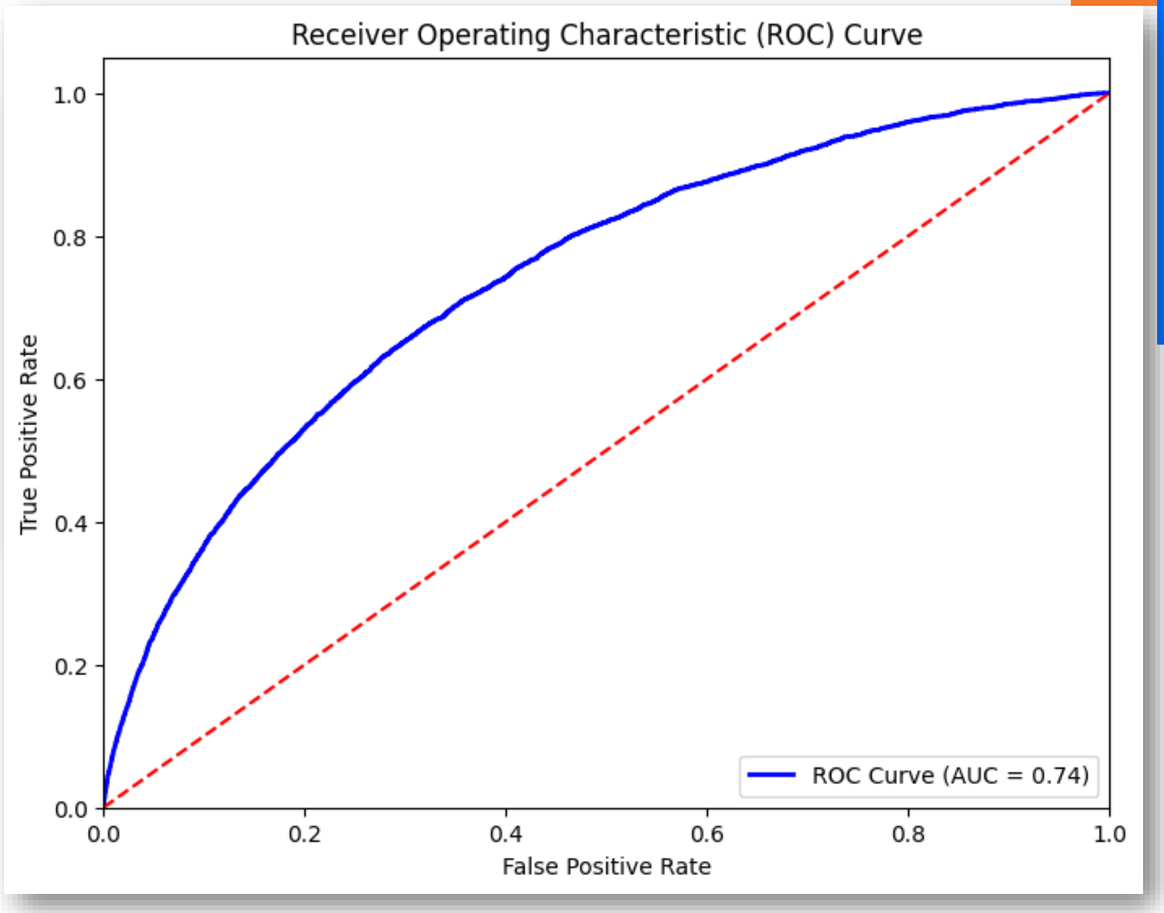
ROC AUC Score: 0.735
Dengan skor 0.735, model Regresi Logistik kita memiliki kinerja yang baik dalam memisahkan kelas positif dan negatif.

Evaluation XGBoost

Classification Report:

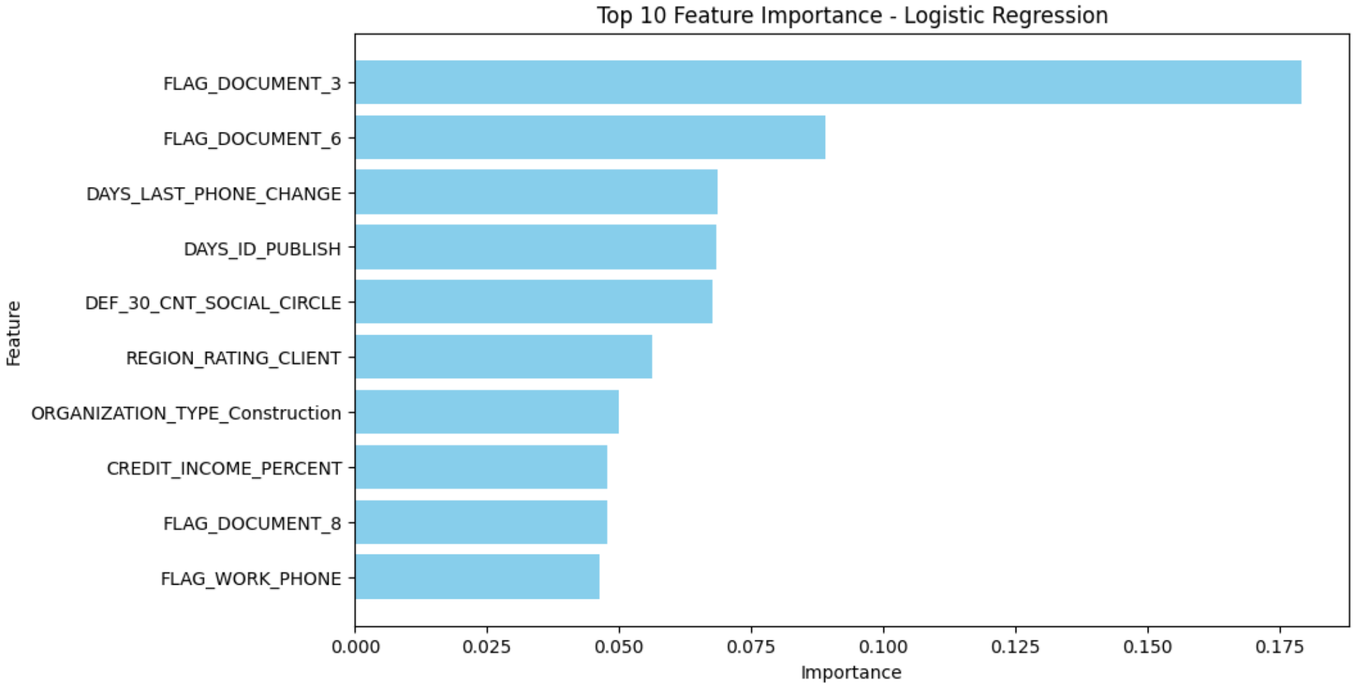
	precision	recall	f1-score	support
0	0.92	1.00	0.96	56536
1	0.55	0.02	0.03	4965
accuracy			0.92	61501
macro avg	0.73	0.51	0.49	61501
weighted avg	0.89	0.92	0.88	61501

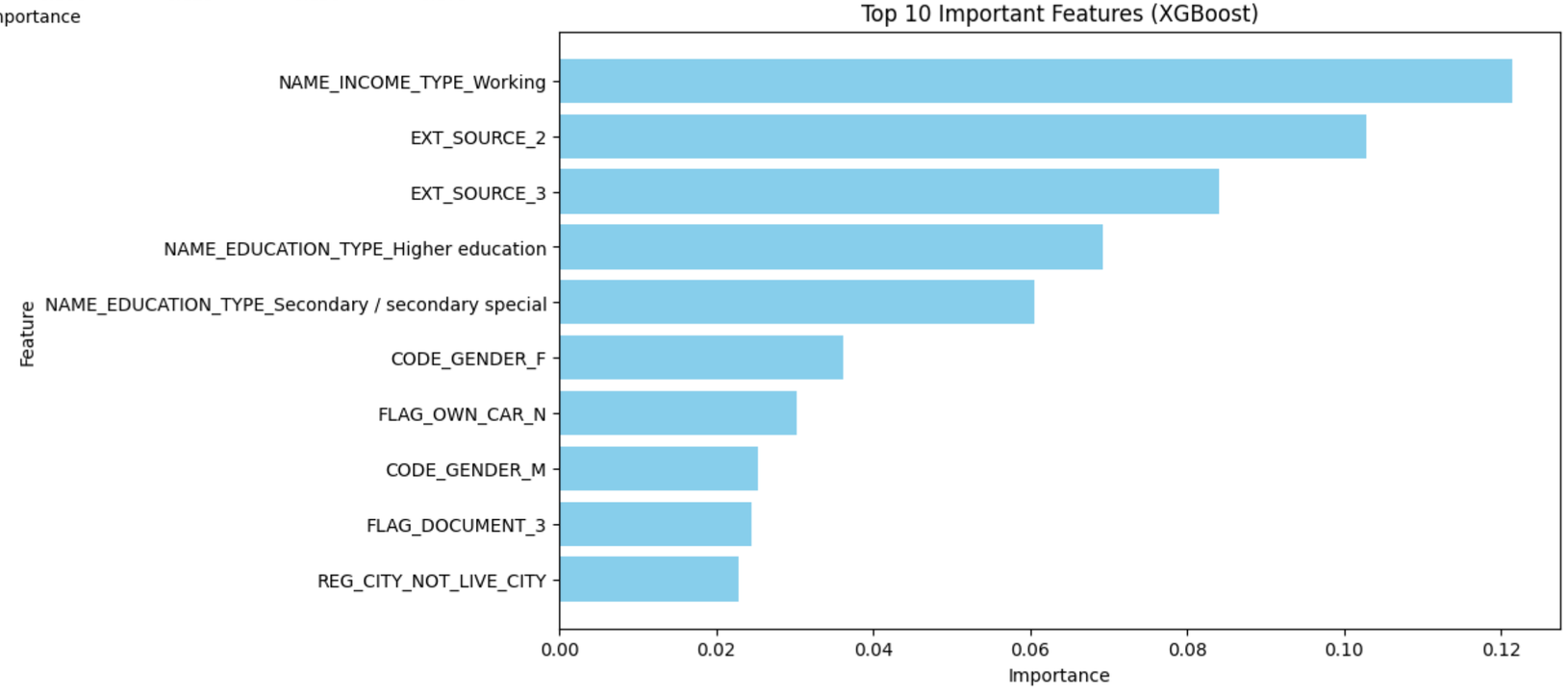
Dari laporan klasifikasi, meskipun akurasi tinggi, model memiliki kinerja yang buruk dalam mengklasifikasikan kelas minoritas (kelas 1).



ROC AUC Score: 0.741
Dengan skor 0.741, model XGBoost kita memiliki kinerja yang baik dalam memisahkan kelas positif dan negatif.

Feature Importance





THANK YOU