Rakamin Academy

HOME CREDIT

Kamu Bisa!

Virtual Internship Experience

Task 5

Home Credit Scorecard Model

Table of Content

Latar Belakang Tugas

Home Credit saat ini sedang menggunakan berbagai macam metode statistik dan Machine Learning untuk membuat prediksi skor kredit. Sekarang, kami meminta anda untuk membuka potensi maksimal dari data kami. Dengan melakukannya, kita dapat memastikan pelanggan yang mampu melakukan pelunasan tidak ditolak ketika melakukan pengajuan pinjaman, dan pinjaman datap diberikan dengan principal, maturity, dan repayment calendar yang akan memotivasi pelanggan untuk sukses. Evaluasi akan dilakukan dengan mengecek seberapa dalam pemahaman analisa yang anda kerjakan. Sebagai catatan, anda perlu menggunakan setidaknya 2 model Machine Learning dimana salah satunya adalah Logistic Regression. Setelah itu, **buatlah slide presentasi yang mengandung analisa hasil pemodelan secara end-to-end beserta rekomendasi bisnisnya (maksimal 10 halaman)**

Tahapan Pengerjaan

1. Download Dataset yang dibutuhkan.

2. Pelajari konteks masalah dari sumber eksternal untuk meningkatkan Subject Matter Knowledge.

3. Pahami deskripsi kolom yang tersedia.

4. Tentukan goal, objective, dan metrics dari masalah yang ada.

5. Lakukan penggalian informasi terkait kondisi data awal.

6. Lakukan proses Data Cleaning dan Data Processing.

7. Lakukan proses penggalian insight mengacu kepada objective yang sudah ditetapkan.

8. Lakukan pemodelan dengan berbagai macam metode (termasuk Logistic Regression) dan hyperparameter-nya.

9. Evaluasi hasil pemodelan.

10. Rekomendasi Bisnis

11. Push file .ipynb mu ke dalam github.

12. Buat file presentasi untuk menjelaskan pekerjaan yang telah dilakukan dan cantumkan link repo github di dalam ppt nya

Tahap 1. Data Yang Digunakan

Untuk data yang digunakan, silahkan akses file berikut ini :

1. Dataset : Click here
2. Columns Description : Click here
3. Pedoman Pembuatan PPT : Click here

Tahap 2. Mempelajari Konteks

Home Credit saat ini sedang menggunakan berbagai macam metode statistik dan Machine Learning untuk membuat prediksi skor kredit. Sekarang, kami meminta anda untuk membuka potensi maksimal dari data kami. Dengan melakukannya, kita dapat memastikan pelanggan yang mampu melakukan pelunasan tidak ditolak ketika melakukan pengajuan pinjaman, dan pinjaman datap diberikan dengan principal, maturity, dan repayment calendar yang akan memotivasi pelanggan untuk sukses. Evaluasi akan dilakukan dengan mengecek seberapa dalam pemahaman analisa yang anda kerjakan. Sebagai catatan, anda perlu menggunakan setidaknya 2 model Machine Learning dimana salah satunya adalah Logistic Regression.

Tahap 3. Memahami deskripsi kolom

Deskripsi Kolom

Disini saya menggunakan data yang sudah di join yaitu **application_train dan application_test.**
Terdapat 122 kolom.

Secara garis besar penjelasan kolom-kolom tersebut adalah:

- Kolom TARGET yaitu menunjukan apakah user membayar loan tepat waktu (0) atau bermasalah (1)
- Kolom CODE_GENDER menunjukan gender dari user
- Kolom AMT_INCOME_TOTAL menunjukan jumlah pendapatan dari user
- Kolom AMT_CREDIT menunjukan jumlah credit dari user
- Kolom NAME_INCOME_TYPE menunjukan tipe pendapatan dari user
- Kolom NAME_EDUCATION_TYPE menunjukan jenjang pendidikan dari user
- Kolom NAME_FAMILY_STATUS menunjukan apakah user sudah menikah atau belum

Dan masih banyak lagi kolom-kolomnya…

Tahap 4. Tentukan goal, objective, dan metrics dari masalah yang ada.

Menentukan **Goal** dan Metrics

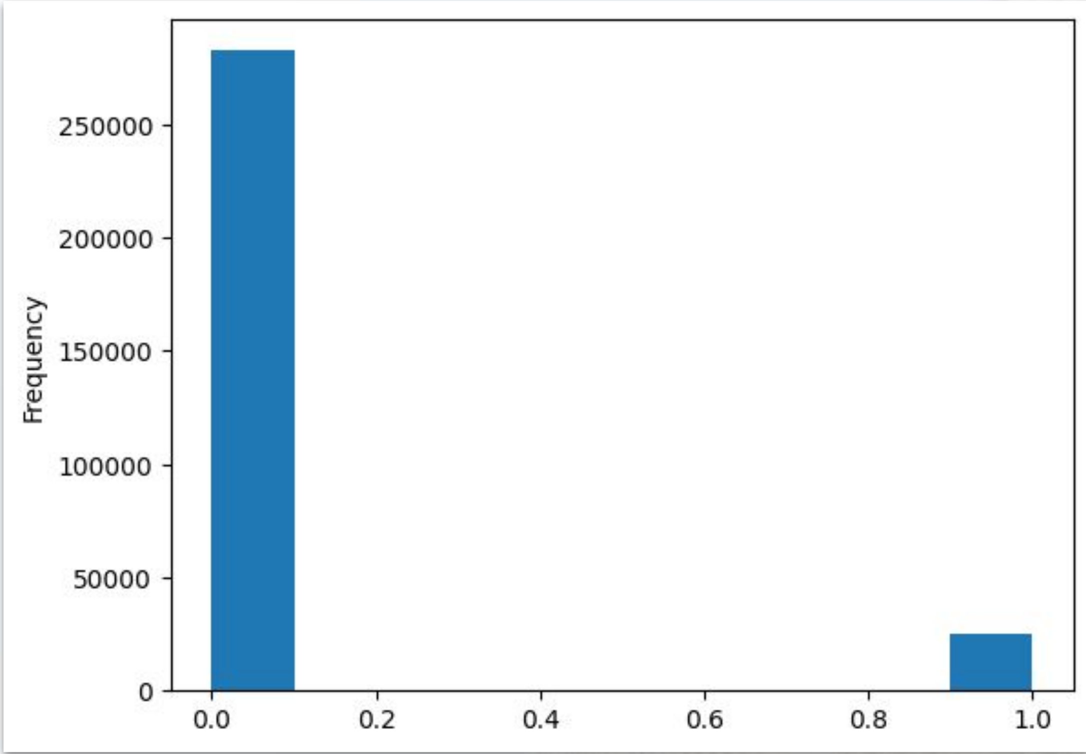
- Goal
  Yaitu membuat sebuah machine learning yang dapat memprediksi apakah user yang akan mengajukan kredit dapat membayar tepat waktu atau akan telat/bermasalah

- Metrics yang akan digunakan adalah ROC-AUC karena **data imbalance**
  - More info about ROC-AUC

- Minimal metrics yang baik adalah 60%

Tahap 5. Lakukan penggalian informasi terkait kondisi data awal.

Informasi Awal dari Data



Target variabel sangat imbalance (proporsi nilai 0 dan 1 sangat timpang)

```
Your selected dataframe has 122 columns.
There are 67 columns that have missing values.
```

	Missing Values	% of Total Values
COMMONAREA_MEDI	214865	69.9
COMMONAREA_AVG	214865	69.9
COMMONAREA_MODE	214865	69.9
NONLIVINGAPARTMENTS_MEDI	213514	69.4
NONLIVINGAPARTMENTS_MODE	213514	69.4
NONLIVINGAPARTMENTS_AVG	213514	69.4
FONDKAPREMONT_MODE	210295	68.4
LIVINGAPARTMENTS_MODE	210199	68.4
LIVINGAPARTMENTS_MEDI	210199	68.4
LIVINGAPARTMENTS_AVG	210199	68.4
FLOORSMIN_MODE	208642	67.8
FLOORSMIN_MEDI	208642	67.8

Dataset memiliki banyak sekali missing values

Informasi Awal dari Data

```
1 # Number of each type of column
2 app_train.dtypes.value_counts()
```

```
float64     65
int64       41
object      16
dtype: int64
```

Dataset memiliki banyak sekali data yang belum numerik

```
1 # Number of unique classes in each object column
2 app_train.select_dtypes('object').apply(pd.Series.nunique, axis = 0)
```

```
NAME_CONTRACT_TYPE             2
CODE_GENDER                    3
FLAG_OWN_CAR                   2
FLAG_OWN_REALTY                2
NAME_TYPE_SUITE                7
NAME_INCOME_TYPE               8
NAME_EDUCATION_TYPE            5
NAME_FAMILY_STATUS             6
NAME_HOUSING_TYPE              6
OCCUPATION_TYPE               18
WEEKDAY_APPR_PROCESS_START     7
ORGANIZATION_TYPE             58
FONDKAPREMONT_MODE             4
HOUSETYPE_MODE                 3
WALLSMATERIAL_MODE             7
EMERGENCYSTATE_MODE            2
dtype: int64
```

Master

Dataset memiliki banyak sekali categorical features

Tahap 6. Lakukan proses Data Cleaning dan Data Processing.

Data Cleansing

Label Encoding and One-Hot Encoding

Let's implement the policy described above: for any categorical variable (`dtype == object`) with 2 unique categories, we will use label encoding, and for any categorical variable with more than 2 unique categories, we will use one-hot encoding.

For label encoding, we use the Scikit-Learn `LabelEncoder` and for one-hot encoding, the pandas `get_dummies(df)` function.

```
[ ]
# Create a label encoder object
le = LabelEncoder()
le_count = 0

# Iterate through the columns
for col in app_train:
    if app_train[col].dtype == 'object':
        # If 2 or fewer unique categories
        if len(list(app_train[col].unique())) <= 2:
            # Train on the training data
            le.fit(app_train[col])
            # Transform both training and testing data
            app_train[col] = le.transform(app_train[col])
            app_test[col] = le.transform(app_test[col])

            # Keep track of how many columns were label encoded
            le_count += 1

print('%d columns were label encoded.' % le_count)
```

```
3 columns were label encoded.
```

```
[ ]
# one-hot encoding of categorical variables
app_train = pd.get_dummies(app_train)
app_test = pd.get_dummies(app_test)

print('Training Features shape: ', app_train.shape)
print('Testing Features shape: ', app_test.shape)
```

Melakukan encoding untuk variables yang masih berupa string dan kategori

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import train_test_split
from sklearn.impute import SimpleImputer

# # Drop the target from the training data
# if 'TARGET' in app_train:
#     train = app_train.drop(columns = ['TARGET'])
# else:
#     train = app_train.copy()

train = app_train.drop(columns=['TARGET'])
label = app_train['TARGET']

x_train, x_test, y_train, y_test = train_test_split(train, label, test_size=0.25)

# Feature names
features = list(train.columns)

# Copy of the testing data
test = app_test.copy()

# Median imputation of missing values
imputer = SimpleImputer(strategy = 'median')

# Scale each feature to 0-1
scaler = MinMaxScaler(feature_range = (0, 1))

# Fit on the training data
imputer.fit(x_train)

# Transform both training and testing data
x_train = imputer.transform(x_train)
x_test = imputer.transform(x_test)
```

Melakukan impute missing values

Tahap 7. Lakukan proses penggalian insight.

Penggalian **Insight**

HOME CREDIT
Kamu Bisa!

Correlations

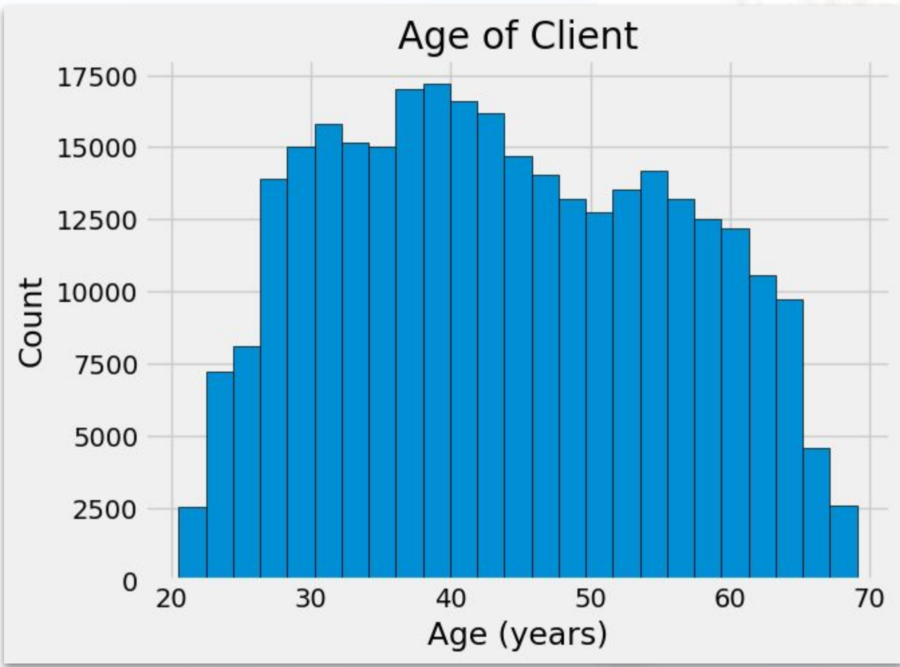
- .00-.19 “very weak”
- .20-.39 “weak”
- .40-.59 “moderate”
- .60-.79 “strong”
- .80-1.0 “very strong”

```python
# Find correlations with the target and sort
correlations = app_train.corr()['TARGET'].sort_values()

# Display correlations
print('Most Positive Correlations:\n', correlations.tail(15))
print('\nMost Negative Correlations:\n', correlations.head(15))
```

```
Most Positive Correlations:
 OCCUPATION_TYPE_Laborers                              0.043019
FLAG_DOCUMENT_3                                       0.044346
REG_CITY_NOT_LIVE_CITY                                0.044395
FLAG_EMP_PHONE                                        0.045982
NAME_EDUCATION_TYPE_Secondary / secondary special     0.049824
REG_CITY_NOT_WORK_CITY                                0.050994
DAYS_ID_PUBLISH                                       0.051457
CODE_GENDER_M                                         0.054713
DAYS_LAST_PHONE_CHANGE                                0.055218
NAME_INCOME_TYPE_Working                              0.057481
REGION_RATING_CLIENT                                  0.058899
REGION_RATING_CLIENT_W_CITY                           0.060893
DAYS_EMPLOYED                                         0.074958
DAYS_BIRTH                                            0.078239
TARGET                                                1.000000
Name: TARGET, dtype: float64
```

Check correlation features and target



Distributions of age client

Tahap 8. Proses Pemodelan

Proses Pemodelan

```python
from sklearn.linear_model import LogisticRegression

# Make the model with the spe DL_All regularization parameter
log_reg = LogisticRegression(C = 0.0001)

# Train on the training data
log_reg.fit(train, y_train)
```

```
▾ LogisticRegression
LogisticRegression(C=0.0001)
```

Now that the model has been trained, we can use it to make predictions. We want to predict the probabilities of not paying a loan, so we use the model `predict.proba` method. This returns an m x 2 array where m is the number of observations. The first column is the probability of the target being 0 and the second column is the probability of the target being 1 (so for a single row, the two columns must sum to 1). We want the probability the loan is not repaid, so we will select the second column.

The following code makes the predictions and selects the correct column.

```python
from sklearn.metrics import roc_auc_score

# Make predictions
# Make sure to select the second column only
log_reg_pred = log_reg.predict_proba(test)[:, 1]

print(roc_auc_score(y_test, log_reg_pred))
```

```
0.6797145095889988
```

Modelling with Logistic Regression

```python
from sklearn.ensemble import RandomForestClassifier

# Make the random forest classifier
random_forest = RandomForestClassifier(n_estimators = 100, random_state = 50, verbose = 1, n_jobs = -1)
```

```python
# Train on the training data
random_forest.fit(train, y_train)

# Extract feature importances
feature_importance_values = random_forest.feature_importances_
feature_importances = pd.DataFrame({'feature': features, 'importance': feature_importance_values})

# Make predictions on the test data
predictions = random_forest.predict_proba(test)[:, 1]
```

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed:   56.5s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:  1.8min finished
[Parallel(n_jobs=2)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done  46 tasks      | elapsed:    2.1s
[Parallel(n_jobs=2)]: Done 100 out of 100 | elapsed:    3.4s finished
```

```python
print(roc_auc_score(y_test, predictions))
```

```
0.709288394664219l
```

Modelling with Random Forest

Tahap 9. Evaluasi hasil pemodelan.

Evaluasi Model

```
[ ]
from sklearn.metrics import roc_auc_score

# Make predictions
# Make sure to select the second column only
log_reg_pred = log_reg.predict_proba(test)[:, 1]

print(roc_auc_score(y_test, log_reg_pred))
```

```
0.6797145095889988
```

Modelling with Logistic Regression

```
[ ]
from sklearn.ensemble import RandomForestClassifier

# Make the random forest classifier
random_forest = RandomForestClassifier(n_estimators = 100, random_state = 50, verbose = 1, n_jobs = -1)
```

```
# Train on the training data
random_forest.fit(train, y_train)

# Extract feature importances
feature_importance_values = random_forest.feature_importances_
feature_importances = pd.DataFrame({'feature': features, 'importance': feature_importance_values})

# Make predictions on the test data
predictions = random_forest.predict_proba(test)[:, 1]
```

```
[Parallel(n_jobs=-1)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=-1)]: Done  46 tasks      | elapsed:   56.5s
[Parallel(n_jobs=-1)]: Done 100 out of 100 | elapsed:  1.8min finished
[Parallel(n_jobs=2)]: Using backend ThreadingBackend with 2 concurrent workers.
[Parallel(n_jobs=2)]: Done  46 tasks      | elapsed:    2.1s
[Parallel(n_jobs=2)]: Done 100 out of 100 | elapsed:    3.4s finished
```

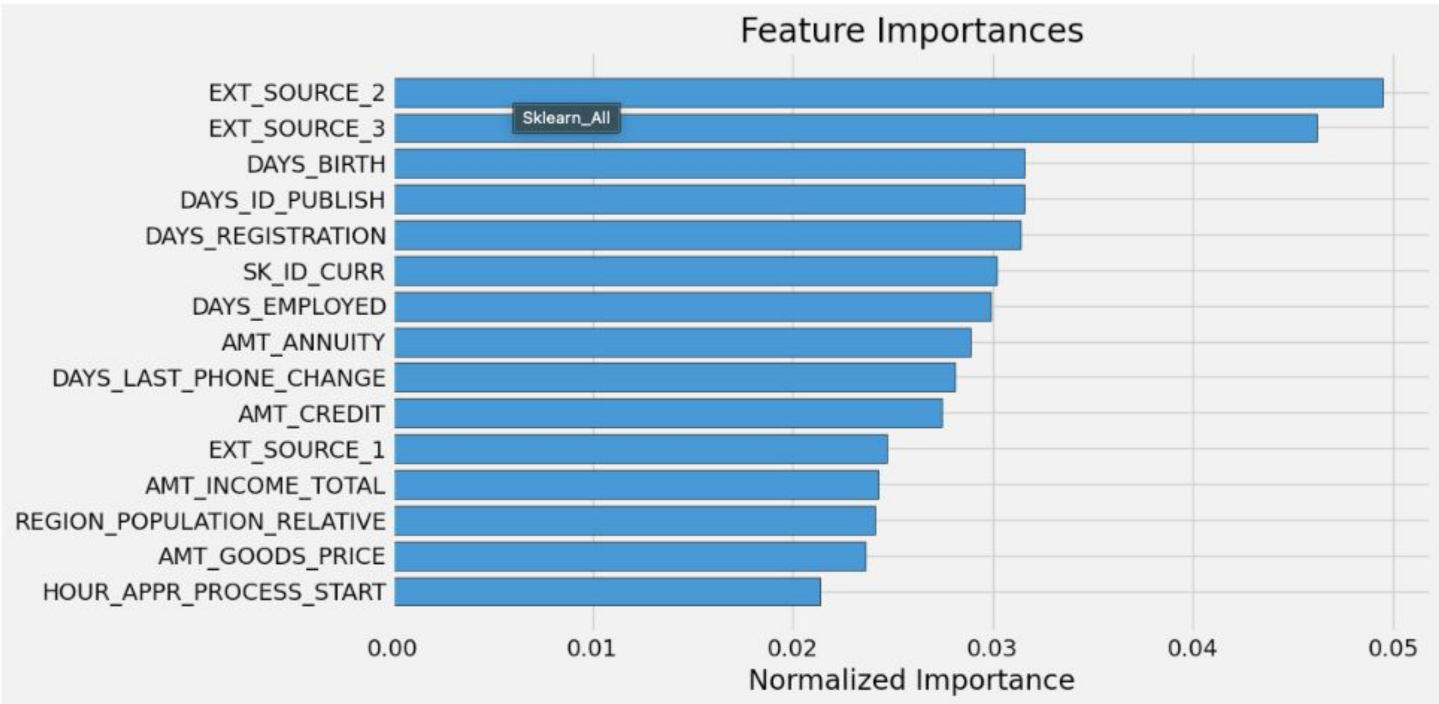
```
[ ]
print(roc_auc_score(y_test, predictions))
```

```
0.7092883946642191
```

Modelling with Random Forest

Tahap 10. Rekomendasi Bisnis

Rekomendasi **Bisnis**



As expected, the most important features are those dealing with `EXT_SOURCE` and `DAYS_BIRTH`. We see that there are only a handful of features with a significant importance to the model, which suggests we may be able to drop many of the features without a decrease in performance (and we may even see an increase in performance.) Feature importances are not the most sophisticated method to interpret a model or perform dimensionality reduction, but they let us start to understand what factors our model takes into account when it makes predictions.

Features importances untuk mengetahui variabel yang mempengaruhi kenapa user gagal bayar/telat bayar

Collabs Link

https://drive.google.com/file/d/1seQj6eQJzdMRFhJJGxIrREuncwrfNJ7p/view?usp=sharing


HOME
CREDIT
Kamu Bisa!