Rakamin Academy X HOME CREDIT

Home Scorecard Model

Home Credit Indonesia - Data Scientist

Presented by
Nabila Parahita

Link GitHub Repositories

About Company

PT Home Credit Indonesia atau yang lebih dikenal dengan Home Credit merupakan perusahaan pembiayaan multiguna multinasional. Perusahaan ini membangun layanan pembiayaan di toko (pembiayaan non-tunai langsung di tempat) untuk konsumen yang ingin membeli produk-produk seperti alat rumah tangga, alat-alat elektronik, handphone, dan furniture. Perusahaan ini juga membangun layanan pembiayaan berbasis teknologi. Didirikan pada tahun 2013 di Jakarta, saat ini Home Credit telah menjangkau lebih dari 19.000 titik distribusi yang tersebar di 144 kota di Indonesia. Hingga bulan Maret 2019, perusahaan ini telah melayani 3,4 juta pelanggan secara online maupun offline.

Project Portfolio

Dalam industri keuangan, penilaian kelayakan kredit (credit scoring) merupakan proses kritis untuk meminimalkan risiko gagal bayar sekaligus memastikan akses pembiayaan bagi pelanggan yang layak. Home Credit, sebagai penyedia layanan pembiayaan, saat ini mengandalkan metode statistik dan machine learning untuk memprediksi kemampuan pelanggan dalam melunasi pinjaman.

Namun, tantangan utama yang dihadapi adalah

- Menemukan keseimbangan antara meminimalkan risiko kredit (seperti default) dan memaksimalkan inklusi keuangan, yaitu menghindari penolakan terhadap pelanggan yang sebenarnya memiliki kapasitas membayar.
- Penentuan besaran pinjaman (principal), jangka waktu (maturity), dan jadwal pembayaran (repayment calendar) yang sesuai juga berpengaruh besar terhadap motivasi dan kesuksesan pelanggan dalam memenuhi kewajibannya.

GOAL

Oleh karena itu, diperlukan pengembangan model prediktif yang tidak hanya akurat secara teknis, tetapi juga dapat memberikan rekomendasi bisnis yang implementatif.

Dalam proyek ini, kami akan membangun dan mengevaluasi dua model machine learning (Logistic Regression & XGBoost) untuk menganalisis data historis, mengidentifikasi pola, dan menghasilkan insight yang dapat mendukung keputusan kredit yang lebih adil dan efektif.

Hasil akhirnya akan disajikan dalam bentuk presentasi end-to-end yang mencakup analisis data, pemodelan, serta rekomendasi strategis untuk optimasi proses pengajuan pinjaman.

1. Load Data & EDA

```
import math
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns

from difflib import SequenceMatcher
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.metrics.pairwise import cosine_similarity
from scipy.stats import zscore
```

```
[ ] df = pd.read_csv('/content/drive/MyDrive/home-credit-default-risk/application_train.csv')
    df
```

```
[ ] df.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 307511 entries, 0 to 307510
Columns: 122 entries, SK_ID_CURR to AMT_REQ_CREDIT_BUREAU_YEAR
dtypes: float64(65), int64(41), object(16)
memory usage: 286.2+ MB
```

```
df.describe()
```

	SK_ID_CURR	TARGET	CNT_CHILDREN	AMT_INCOME_TOTAL

```
[ ] missing_percent = (df.isnull().sum() / len(df)) * 100
    missing_percent
```

```
[ ] df.drop_duplicates() #membersihkan data dari baris yang sama persis.
```

	SK_ID_CURR	TARGET	NAME_CONTRACT_TYPE	CODE_GENDER	FLAG_OWN_CAR	F
					M	N

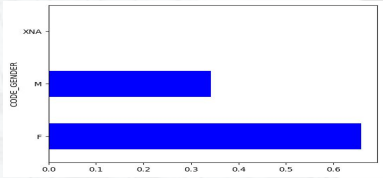
```
[ ] #mengecek missing value pada baris
    emptyrow=df.isnull().sum(axis=1)
    emptyrow
```

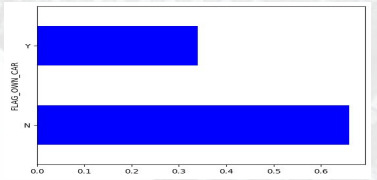
```
[ ] #melihat jumlah missing value kolom
    df.isnull().sum()
```

	0
SK_ID_CURR	0
TARGET	0
NAME_CONTRACT_TYPE	0
CODE_GENDER	0
FLAG_OWN_CAR	0
...	...
AMT_REQ_CREDIT_BUREAU_DAY	41519
AMT_REQ_CREDIT_BUREAU_WEEK	41519
AMT_REQ_CREDIT_BUREAU_MON	41519
AMT_REQ_CREDIT_BUREAU_QRT	41519
AMT_REQ_CREDIT_BUREAU_YEAR	41519

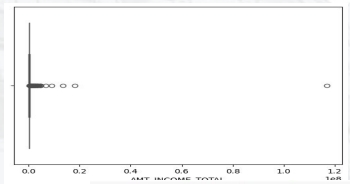
```
[ ] #cek kolom kategorik dan numerik
    cat = df.select_dtypes(include=['object']).columns.tolist()
    num = df.select_dtypes(exclude=['object']).columns.tolist()

    print('categorical variable :', cat)
    print('numerical variable :', num)
```

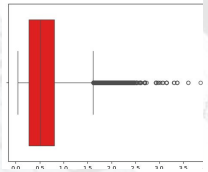
```
categorical variable : ['NAME_CONTRACT_TYPE', 'CODE_GENDER',
numerical variable : ['SK_ID_CURR', 'TARGET', 'CNT_CHILDREN',
```






```
] # distribution plot
  # melihat distribusi data numerik.
  for i in range(len(num)):
      sns.distplot(df[num[i]], color="red")
      plt.show()
```





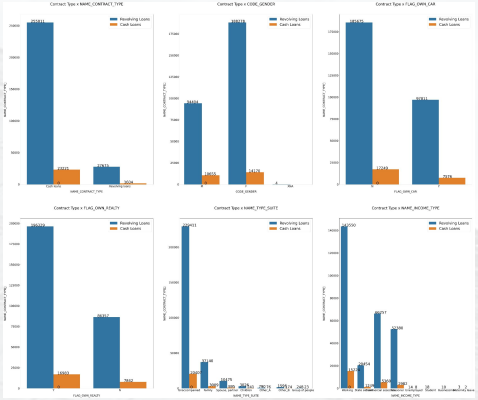
```
[ ] #Numerical Univariate Analysis
    #mendeteksi outlier dan menganalisis distribusi data numerik.
    # boxplot
    for i in range(len(num)):
        sns.boxplot(df[num[i]], color = "red", orient = "h")
        plt.show()
```

Melakukan Import data, mengecek jumlah kolom, mengecek missing value, menghapus duplicate, dan melihat distribusi data numerik dan kategorik

2. Feature Engineering & Data Balancing

Rakamin Academy

```
#Menganalisis distribusi 6 variabel kategorikal pertama (cat[:6]) dalam hubungannya dengan TARGET.
# Menggunakan countplot untuk membandingkan jumlah kategori dalam variabel, dengan pewarnaan berdasarkan TARGET.
fig, axs = plt.subplots(ncols = 3, nrows = 2, figsize = (50, 50))
plt.subplots_adjust(right = 1.5, top = 1.25)

for i, feature in enumerate(cat[:6], 1):
    plt.subplot(2, 3, i)
    ax = sns.countplot(x = feature, hue = 'TARGET', data = df)

    plt.xlabel('{}'.format(feature), size = 30, labelpad = 30)
    plt.ylabel('NAME_CONTRACT_TYPE  ', size = 30, labelpad = 30)
    plt.tick_params(axis = 'x', labelsize = 30)
    plt.tick_params(axis = 'y', labelsize = 30)

    for p in ax.patches:
        ax.annotate('{:.0f}'.format(p.get_height()), (p.get_x()+0.06, p.get_height()+20), size = 40)

    plt.legend(['Revolving Loans', 'Cash Loans'], loc = 'upper right', prop = {'size': 40})
    plt.title('Contract Type x {}'.format(feature), size = 40, y = 1.05)

plt.show()
```



Menganalisis distribusi & variabel kategorikal dan hubungannya dengan target

Feature Engineering

```
#one-hot encoding pada kolom kategorik
target_encoded = pd.get_dummies(df.drop(['TARGET'], axis=1))
target_encoded['TARGET']= df['TARGET']
target_encoded.head()
```

Seperti yang terlihat, kolom **'Gender'** dan **'Education'** telah diubah menjadi bentuk **one-hot encoding**,

Split dataset

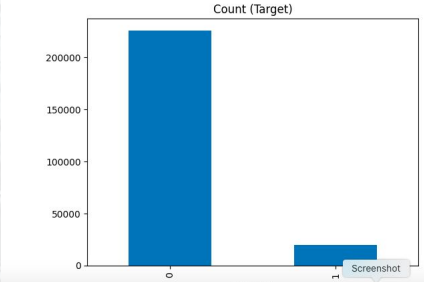
```
#membagi dataset menjadi data latih (training data) dan data uji (testing data)
from sklearn.model_selection import train_test_split

X = target_encoded.drop(['TARGET'], axis = 1)
y = target_encoded['TARGET']

X_train, X_test, y_train, y_test = train_test_split(X, y ,test_size = 0.20)
```

Membagi dataset menjadi data train dan data test

Balancing dataset

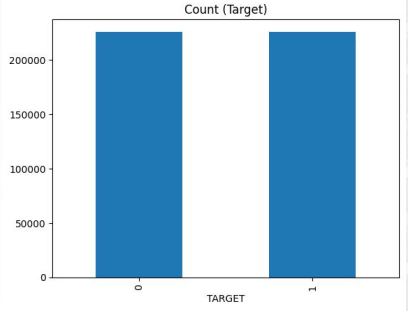
```
#menampilkan distribusi dari nilai target (y_train)
y_train.value_counts().plot(kind = 'bar', title='Count (Target)')
```

```
<Axes: title={'center': 'Count (Target)'}, xlabel='TARGET'>
```




```
#mengatasi masalah ketidakseimbangan kelas
#SMOTE adalah teknik untuk menghasilkan sampel sintetik untuk kelas minoritas agar data menjadi lebih seimbang.

from imblearn.over_sampling import SMOTE

oversample = SMOTE()
X_train, y_train = oversample.fit_resample(X_train, y_train)

print('After Sampling Data:')
print(y_train.value_counts())

y_train.value_counts().plot(kind = 'bar', title='Count (Target)')
```

```
After Sampling Data:
TARGET
0    226067
1    226067
Name: count, dtype: int64
<Axes: title={'center': 'Count (Target)'}, xlabel='TARGET'>
```



Karena data imbalance lebih banyak yang (0) maka dilakukan **SMOTE** teknik untuk menghasilkan sampel yg simetris untuk kelas minoritas agar data lebih seimbang

3. Normalization & Feature Importance

Feature Scaling (Normalization)

```
[ ]  from sklearn.preprocessing import MinMaxScaler

     sc = MinMaxScaler()
     X_train = sc.fit_transform(X_train)
     X_test = sc.transform(X_test)
```

Feature Importance

```
[ ] #menampilkan feature importance
    from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score
    from sklearn.metrics import roc_curve, auc

    def eval_classification(model, pred, xtrain, ytrain, xtest, ytest):
        print("Accuracy (Test Set): %.4f" % accuracy_score(ytest, pred))
        print("Precision (Test Set): %.4f" % precision_score(ytest, pred))
        print("Recall (Test Set): %.4f" % recall_score(ytest, pred))
        print("F1-Score (Test Set): %.4f" % f1_score(ytest, pred))

        fpr, tpr, thresholds = roc_curve(ytest, pred, pos_label=1) # pos_label: label yang kita anggap positive
        print("AUC: %.4f" % auc(fpr, tpr))

    def built_in_feature_importance(model):
        feat_importances = pd.Series(model.feature_importances_, index=X.columns)
        ax = feat_importances.nlargest(25).plot(kind='barh', figsize=(10, 8))
        ax.invert_yaxis()

        plt.xlabel('score')
        plt.ylabel('feature')
        plt.title('feature importance score')

        plt.show()  # Menampilkan plot
```

Screenshot

melatih & mengevaluasi performa model dengan eval_classification(). Menampilkan feature importance dengan built_in_feature_importance().

4. Logistic Regression

Logistic Regression

```
[ ] #Train Model using Logistic Regression

    from sklearn.linear_model import LogisticRegression

    # Create an instance of the model.
    logreg = LogisticRegression()

    # Training the model.
    logreg.fit(X_train,y_train)

    # Do prediction.
    y_pred = logreg.predict(X_test)

    print(eval_classification(logreg, y_pred, X_train, y_train, X_test, y_test))
```

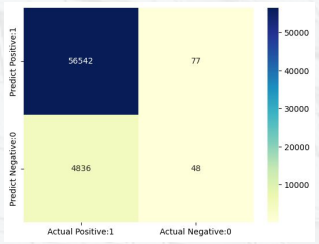
```
Accuracy (Test Set): 0.9201
Precision (Test Set): 0.3840
Recall (Test Set): 0.0098
F1-Score (Test Set): 0.0192
AUC: 0.5042
```

Metrik	Interpretasi di Dataset Home Credit
Accuracy (92.01%)	Model lebih sering menebak "tidak gagal bayar" (0), sehingga akurasi tinggi tetapi bisa menyesatkan.
Precision (38.40%)	Dari semua orang yang diprediksi gagal bayar (1), hanya 38.40% yang benar-benar gagal bayar.
Recall (0.98%)	Model **hampir tidak bisa** mendeteksi pelanggan yang benar-benar gagal bayar. Hanya 0.98% dari semua peminjam gagal bayar yang terdeteksi.
F1-Score (1.92%)	Model sangat buruk dalam menyeimbangkan precision dan recall untuk kelas gagal bayar.
AUC (50.42%)	Model hampir **tidak bisa membedakan** antara peminjam yang akan gagal bayar dan yang tidak. Seperti **tebakan acak**.

```
[ ] # Print the Confusion Matrix and slice it into four pieces
    from sklearn.metrics import confusion_matrix

    cm = confusion_matrix(y_test, y_pred)

    print('Confusion matrix\n\n', cm)

    print('\nTrue Positives(TP) = ', cm[0,0])

    print('\nTrue Negatives(TN) = ', cm[1,1])

    print('\nFalse Positives(FP) = ', cm[0,1])

    print('\nFalse Negatives(FN) = ', cm[1,0])
```

```
Confusion matrix

 [[56542    77]
 [ 4836    48]]

True Positives(TP) =  56542

True Negatives(TN) =  48

False Positives(FP) =  77

False Negatives(FN) =  4836
```

```
] # visualize confusion matrix with seaborn heatmap
  import seaborn as sns
  cm_matrix = pd.DataFrame(data = cm, columns = ['Actual Positive:1', 'Actual Negative:0'],
                                      index = ['Predict Positive:1', 'Predict Negative:0'])

  sns.heatmap(cm_matrix, annot = True, fmt = 'd', cmap = 'YlGnBu')
```



5. XGBoost

Rakamin Academy

```
[ ] from xgboost import XGBClassifier

    # Create an instance of the model.
    xg = XGBClassifier(random_state = 50)

    # Training the model.
    xg.fit(X_train, y_train)

    # Do prediction.
    y_pred = xg.predict(X_test)

    print(eval_classification(xg, y_pred, X_train, y_train, X_test, y_test))
    print("="*25)
```

```
Accuracy (Test Set): 0.9197
Precision (Test Set): 0.4212
Recall (Test Set): 0.0285
F1-Score (Test Set): 0.0533
AUC: 0.5125
None
==========
```

```
[ ] from sklearn.metrics import classification_report

    print(classification_report(y_test, y_pred))
```

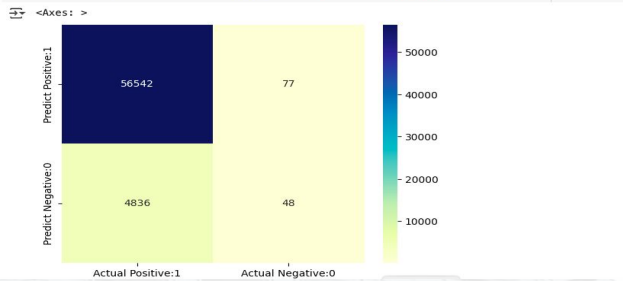
	precision	recall	f1-score	support
0	0.92	1.00	0.96	56619
1	0.42	0.03	0.05	4884
accuracy			0.92	61503
macro avg	0.67	0.51	0.51	61503
weighted avg	0.88	0.92	0.89	61503

Model tampak memiliki akurasi tinggi, Dari semua yang diprediksi sebagai gagal bayar (TARGET = 1), hanya 42.12% yang benar-benar gagal bayar.

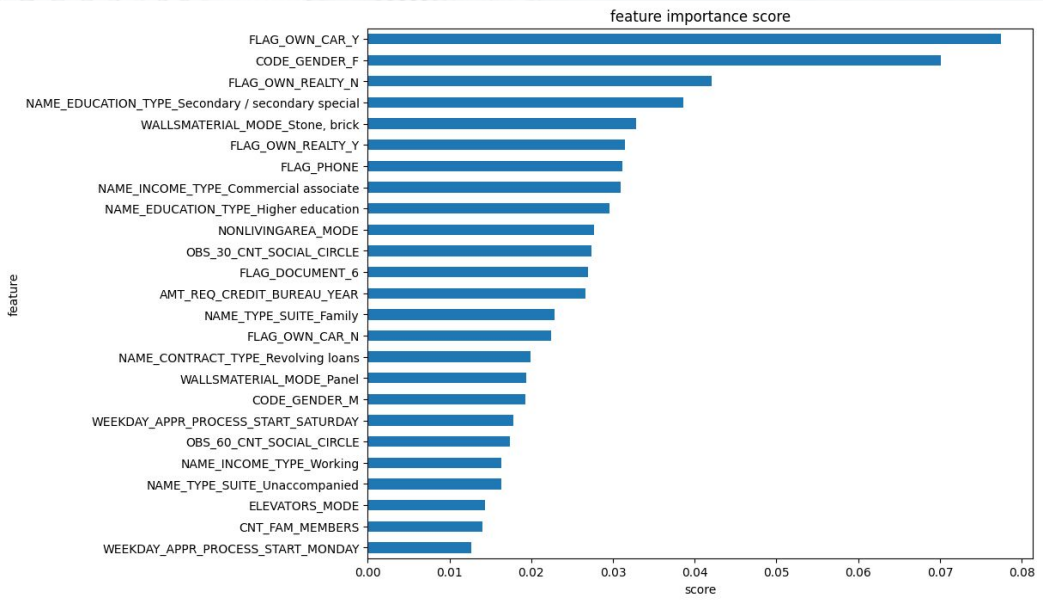
```
] TP = cm[0,0]
  TN = cm[1,1]
  FP = cm[0,1]
  FN = cm[1,0]

  accuracy = (TP+TN) / float(TP+TN+FP+FN)
  print('Classification accuracy : {0:0.4f}'.format(accuracy))

  class_error = (FP+FN) / float(TP+TN+FP+FN)
  print('Classification Error : {0:0.4f}'.format(class_error))

  precision = TP / float(TP + FP)
  print('Precision : {0:0.4f}'.format(precision))

  recall = TP / float(TP + FN)
  print('Recall or Sensitivity or TPR : {0:0.4f}'.format(recall))
```

```
Classification accuracy : 0.9201
Classification Error : 0.0799
Precision : 0.9986
Recall or Sensitivity or TPR : 0.9212
```

```
[ ] # visualize confusion matrix with seaborn heatmap
    cm_matrix = pd.DataFrame(data = cm, columns = ['Actual Positive:1', 'Actual Negative:0'],
                                        index = ['Predict Positive:1', 'Predict Negative:0'])

    sns.heatmap(cm_matrix, annot = True, fmt = 'd', cmap = 'YlGnBu')
```



Prediction Analysis



Berdasarkan analisis ***feature importance*** dari **model XGBoost**, dapat disimpulkan bahwa model home credit scoring ini secara signifikan dipengaruhi oleh **tiga kategori fitur utama**: **kepemilikan aset, latar belakang pendidikan, dan riwayat kredit**.

Fitur seperti **FLAG**, **OWN_CAR_N**, dan **CODE_GENDER_N** menempati peringkat **tertinggi**, menunjukkan bahwa status kepemilikan mobil dan properti **(FLAG_OWN_REALTY_Y/N)** serta demografi dasar seperti gender memiliki **pengaruh dominan** dalam penilaian kelayakan kredit. Selain itu, tingkat pendidikan **(NAME_EDUCATION_TYPE)** dan material bangunan rumah **(WALLSMATERIAL_MODE)** juga berkontribusi penting, mengindikasikan bahwa stabilitas finansial dan latar belakang sosio-ekonomi pemohon menjadi pertimbangan kritis.

Di sisi lain, fitur seperti riwayat interaksi dengan biro kredit **(AMT_REQ_CREDIT_BUREAU_YEAR)** dan lingkaran sosial **(OBS_30_CNT_SOCIAL_CIRCLE)** turut memengaruhi, meskipun dengan porsi yang lebih kecil.

Link GitHub Repositories

Thank You