Rakamin Academy

Homework

Machine Learning Evaluation & Supervised Learning
Stage 3 - Final Project

Estimasi Waktu Pengerjaan

3 - 5 jam

Jumlah Soal

2 Soal

Total Point

100 poin

Teknis Pengerjaan

1. Pekerjaan dilakukan secara **berkelompok, sesuai kelompok Final Project**

2. Masing-masing anggota kelompok tetap perlu submit ke LMS (jadi bukan perwakilan)

3. File yang perlu dikumpulkan:
   a. File **jupyter notebook** (.ipynb) yang berisi source code.
   b. File **laporan homework** (.pdf) yang berisi rangkuman dari apa saja yang telah dilakukan.

4. Upload hasil pengerjaan melalui LMS.
   a. Masukkan semua file ke dalam **1file** dengan format **ZIP**.
   b. Nama File:
   `Supervised - <Nama Kelompok>.zip`

1. Modeling (70 poin)

Lakukan modeling sesuai task yang ditentukan di awal dari hasil data yang telah dilakukan preprocessing dan cleaning pada tahapan sebelumnya:

A. Split Data Train & Test
B. Modeling (Algoritma yang diimplementasikan tidak terbatas yang diajarkan di kelas)
C. Model Evaluation: Pemilihan dan perhitungan metrics model
D. Model Evaluation: Apakah model sudah best-fit? Hindari Overfit/Underfit. Validasi dengan cross-validation
E. Hyperparameter Tuning

Di laporan homework, tuliskan eksperimen apa saja yang telah dilakukan dan metode yang digunakan, dan mengapa memilih dan mengimplementasikan algoritma tersebut. Jelaskan masing-masing hasil dari eksperimen model yang telah dilakukan, alasan menggunakan metrics pada model tersebut, serta hyperparameter yang digunakan dan pengaruhnya terhadap model.

2. Feature Importance (30 poin)

Setelah mendapatkan model yang paling baik, lakukan interpretasi pada model dengan melihat feature importance-nya. Apa yang menyebabkan hasil prediksi model demikian.

Feature importance

Evaluasi feature yang paling penting,
Tarik business insight-nya,
Berikan action items berupa rekomendasi terhadap insight tersebut

Rakamin Academy

LOGISTIC REGRESSION

1. Modeling (70 poin)

Dataset yang digunakan memiliki data target berupa tipe data kategorikal. Oleh karena itu untuk metrics score yang digunakan adalah accuracy, precision, recall, roc_auc dan f1.

Jika kita melihat dari nilai False Positif dan False Negatif, impact yang dapat diberikan terhadap case bisnis disini :

- False Positif : Kita memprediksi bahwa user akan default, padahal user mampu bayar. Akibatnya revenue kita berkurang, karena kehilangan user yang mampu bayar
- False Negatif : Kita memprediksi bahwa user mampu bayar, padahal user gagal bayar (default). Akibatnya kita rugi karena terlanjur memberikan kredit balance kepada user

Dari case di atas, jika mempertimbangkan kerugian nya, nilai False Negatif ini harus diminimalisir. Oleh karena itu dari model, kita akan berfokus ke nilai Recall nya

Untuk model algorithma yang akan digunakan diantaranya : logreg, decision tree, random forest dan xgboost

LOGISTIC REGRESSION :

Untuk model ini, digunakan 2 metode split data. Random split dan cross validation. Untuk metode random_split, semua tahap pre-processing dilakukan kecuali penambahan atau drop feature. Hal ini dikarenakan kami ingin mengetahui performa algorithma dari data set sebelum dilakukan penambahan atau pengurangan feature. Berikut hasilnya :

RUN MODEL

```
[34]: from sklearn.linear_model import LogisticRegression

lr = LogisticRegression()
lr.fit(X_train, y_train)
eval_classification(lr)
```

```
Accuracy (Train Set): 0.69
Accuracy (Test Set): 0.70
Precision (Train Set): 0.70
Precision (Test Set): 0.39
Recall (Train Set): 0.68
Recall (Test Set): 0.58
AUC (train-proba): 0.76
AUC (test-proba): 0.71
F1-Score (Train Set): 0.69
F1-Score (Test Set): 0.46
```

LOGISTIC REGRESSION :

Masing-masing metrics memiliki nilai :

- Accuracy : 70%
- Precision : 39%
- Recall : 58%
- ROC_AUC : 71%
- F1 Score : 46%

Masih terdapat indikasi overfitting pada model. Karena itu Langkah selanjutnya yang dilakukan adalah hyperparameter tuning. Parameter yang digunakan:

- Nilai C : Nilai regularization
- Penalty : l2 (ridge) dan l1 (lasso)

Disini kami menggunakan 2 metode yaitu GridSearch dan RandomizedSearch untuk membandingkan model mana yang lebih cocok.

LOGISTIC REGRESSION :

RandomizedSearchCV :

TUNING HYPERPARAMETER (RANDOMIZED)

[55]:
```
from sklearn.model_selection import RandomizedSearchCV
import numpy as np

penalty = ['l1', 'l2']
C = [float(x) for x in np.linspace(0.0001, 1, 100)]
hyperparameters = dict(penalty=penalty, C=C)

lr = LogisticRegression()
rs = RandomizedSearchCV(lr, hyperparameters, scoring='recall', random_state=42, cv=5)
rs.fit(X_train, y_train)
eval_classification(rs)
```

```
Accuracy (Train Set): 0.70
Accuracy (Test Set): 0.71
Precision (Train Set): 0.71
Precision (Test Set): 0.40
Recall (Train Set): 0.66
Recall (Test Set): 0.58
AUC (train-proba): 0.76
AUC (test-proba): 0.71
F1-Score (Train Set): 0.69
F1-Score (Test Set): 0.47
```

LOGISTIC REGRESSION :

GridSearchCV :

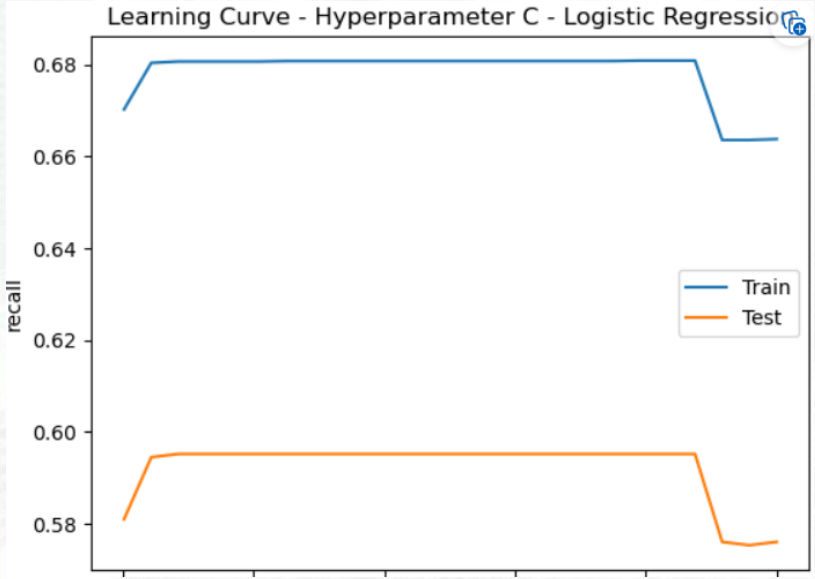
TUNING HYPERPARAMETER (GRID)

[56]:
```
from sklearn.model_selection import GridSearchCV
import numpy as np

penalty = ['l1', 'l2']
#C = [float(x) for x in np.linspace(0.0001, 1, 100)]
C = [float(x) for x in np.linspace(0.0021,0.0035, 25)] #di dapat setelah melakukan analyze curve C
hyperparameters = dict(penalty=penalty, C=C)

lr = LogisticRegression()
model = GridSearchCV(lr, hyperparameters, scoring='recall', cv=5)
model.fit(X_train, y_train)
eval_classification(model)
```

```
Accuracy (Train Set): 0.68
Accuracy (Test Set): 0.67
Precision (Train Set): 0.68
Precision (Test Set): 0.36
Recall (Train Set): 0.68
Recall (Test Set): 0.60
AUC (train-proba): 0.73
AUC (test-proba): 0.68
F1-Score (Train Set): 0.68
F1-Score (Test Set): 0.45
```

LOGISTIC REGRESSION :

Dari kedua model, di dapatkan untuk nilai Recall yang lebih baik ada pada GridSearchCV (60%). Nilai ini tentunya masih kurang karena metrics score masih imbalance. Ada indikasi bahwa algorithma Logistic Regression ini kurang cocok, karena sebaran data tidak linier



Gambar disamping merupakan curve C untuk mencari nilai C yang optimum untuk model. Sedangkan nilai C yang optimum didapatkan pada range 0.0021 – 0.0035.

Feature Importance

Rakamin Academy

FEATURE IMPORTANCE

```
[62]: X_train.columns

[62]: Index(['ID', 'LIMIT_BAL', 'SEX', 'EDUCATION', 'MARRIAGE', 'AGE', 'PAY_1',
             'PAY_2', 'PAY_3', 'PAY_4', 'PAY_5', 'PAY_6', 'BILL_AMT1', 'BILL_AMT2',
             'BILL_AMT3', 'BILL_AMT4', 'BILL_AMT5', 'BILL_AMT6', 'PAY_AMT1',
             'PAY_AMT2', 'PAY_AMT3', 'PAY_AMT4', 'PAY_AMT5', 'PAY_AMT6'],
            dtype='object')

[65]: model.best_estimator_.coef_

[65]: array([[-5.36187253e-07, -8.51361534e-02, -2.21371010e-02,
               8.16084508e-03, -2.97186366e-02, -4.83385971e-04,
               1.76722951e-01,  1.70673244e-01,  1.59133728e-01,
               1.45747505e-01,  1.37567701e-01,  1.30082241e-01,
              -1.45766796e-02, -9.60967508e-03, -1.01127567e-02,
              -7.59946269e-03, -5.57016017e-03, -6.59426417e-03,
              -4.06239512e-02, -3.36932226e-02, -3.51723142e-02,
              -3.04983138e-02, -3.11771431e-02, -5.35053668e-02]])
```

Dari Data di atas, feature PAY_1 sampai PAY_6 merupakan feature yang memberikan informasi kepada model paling banyak. Hal ini dikarenakan kecenderungan seseorang melakukan default dapat dipengaruhi oleh status pembayaran user tiap bulannya.

KESIMPULAN

Dari hasil random split di dapatkan score recall : 58%, kemudian setelah dilakukan tuning hyperparameter di dapatkan nilai recall naik menjadi 60%. Walaupun nilai recall ini sudah mencukupi, tetapi masih ada metrics yang belum memenuhi seperti precision dan f1 score. Jadi dapat disimpulkan sementara ini model dengan algorithma logistic regression kurang cocok jika diterapkan dengan dataset yang dimiliki

DECISION TREE

Decision Tree

1. Modeling

```
# Decision Tree Modeling
from sklearn.tree import DecisionTreeClassifier # import decision tree dari sklearn
dt = DecisionTreeClassifier(random_state = 42) # inisiasi object dengan nama dt
dt.fit(X_train, y_train) # fit model decision tree dari data train
```

2. Model Evaluation

```
Accuracy (Test Set): 0.57
Precision (Test Set): 0.29
Recall (Test Set): 0.64
F1-Score (Test Set): 0.40
roc_auc (test-proba): 0.60
roc_auc (train-proba): 1.00
```

Diperoleh beberapa metrik seperti di samping. Karena kita ingin meminimalisir false negative, maka akan difokuskan pada metrik recall. Diperoleh skor recall yang decent sebesar 0.64.

Kemudian jika dilihat dari roc_auc training dan testing, diketahui bahwa model ada indikasi overfit.

```
roc_auc (crossval train): 0.9999988893492378
roc_auc (crossval test): 0.7632493725234545
```

Saat dilakukan data validation, model memang diindikasikan mengalami overfit.

Decision Tree

Tuning Hyperparameter

```
# List of hyperparameter
max_depth = [int(x) for x in np.linspace(1, 110, num = 30)] # Maximum number of levels in tree
min_samples_split = [2, 5, 10, 100] # Minimum number of samples required to split a node
min_samples_leaf = [1, 2, 4, 10, 20, 50] # Minimum number of samples required at each leaf node
max_features = ['auto', 'sqrt'] # Number of features to consider at every split
criterion = ['gini','entropy']
splitter = ['best','random']
```

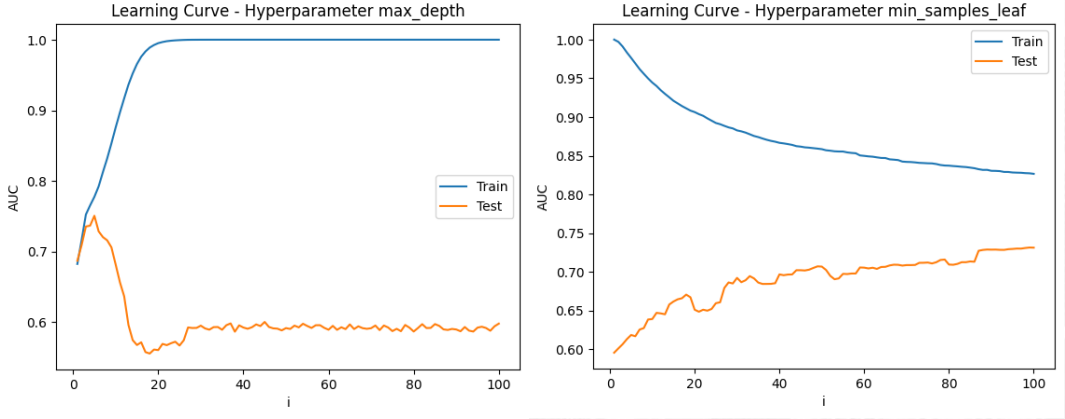
Dilakukan tuning hyperparameter dengan beberapa hyperparameter di sampipng, dan digunakan Random Search untuk mencari kombinasi dari nilai hyperparameter.

```
Accuracy (Test Set): 0.68
Precision (Test Set): 0.38
Recall (Test Set): 0.63
F1-Score (Test Set): 0.47
roc_auc (test-proba): 0.72
roc_auc (train-proba): 0.80
```

Setelah melakukan tuning hyperparameter, diperoleh skor accuracy yang meningkat, dan skor recall yang tetap. Dan hasil roc_auc training dan testing yang lebih baik, karena memiliki selisih yang tidak beda jauh.

Decision Tree

Tuning Hyperparameter



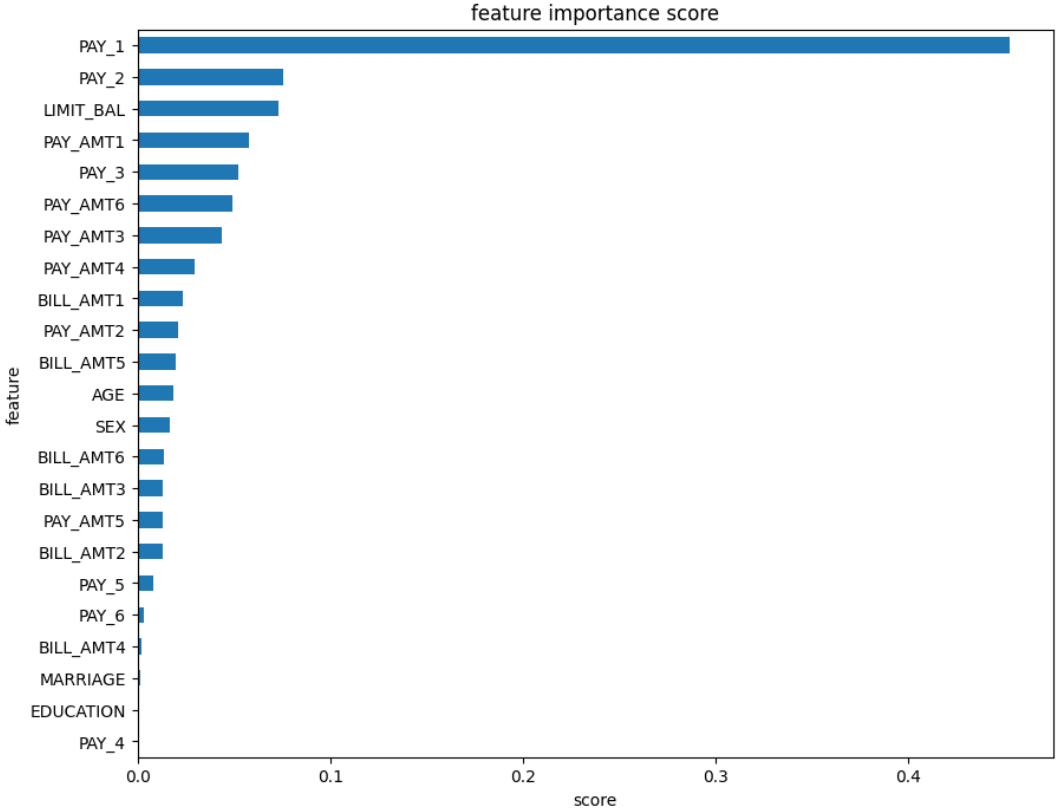
Dari max_depth, nilai kedalaman ideal berada pada sekitaran range 6 hingga 11, dan dari min_samples_leaf, nilai kedalaman ideal berada pada sekitaran range 60 hingga 63.

```
# List of hyperparameter
max_depth = [6,7,8,9,10,11] # Maximum number of levels in tree
min_samples_leaf = [60,61,62,63] # Minimum number of samples required at each leaf node
```

```
Accuracy (Test Set): 0.64
Precision (Test Set): 0.34
Recall (Test Set): 0.64
F1-Score (Test Set): 0.45
roc_auc (test-proba): 0.71
roc_auc (train-proba): 0.84
```

Kemudian dilakukan penyesuaian Hyperparameter, dan dengan Grid Search untuk mencari kombinasi nilai hyperparameter, diperoleh skor accuracy yang meningkat dibanding tanpa tuning hyperparameter, dan skor recall yang tetap. Dan hasil roc_auc training dan testing yang lebih baik, karena memiliki selisih lebih kecil dibanding dengan model tanpa tuning hyperparameter.

Feature Importance (Dari Model Decision Tree)



Feature Pay_1, Pay_2, dan limit_bal dianggap merupakan feature importance paling penting dan memberikan informasi paling banyak di model.

Jika dilihat dari sisi bisnis, feature Pay_1 dan 2 memberikan banyak informasi karena default payment cenderung mudah dilihat hanya dengan melihat bagaimana perilaku pembayaran client pada bulan pertama dan kedua, apakah mereka lebih suka telat bayar atau tepat bayar.

Dan tenti limit kredit dari client juga sangat berpengaruh untuk melihat default payment dari client, karena limit ditentukan dari bagaimana Riwayat pembayaran kredit client sebelumnya.

Selamat Mengerjakan!