Optimasi Algoritma *Random Forest* dengan *Hyperparameter Tuning* Menggunakan *GridSearchCV* untuk Prediksi Nasabah *Churn* pada Industri Perbankan

Nisa Amalia[1], Asmunin[2]

Manajemen Informatika, Universitas Negeri Surabaya
Kampus Ketintang, Jalan Ketintang, Surabaya 60231
[1]nisa.20038@mhs.unesa.ac.id
[2]asmunin@unesa.ac.id

Abstrak— ***Seiring dengan perkembangan teknologi, banyak industri bank yang saling berkompetisi untuk menarik perhatian seseorang. Namun dengan ketatnya persaingan tersebut, menyebabkan beberapa bank harus kehilangan nasabah mereka. Nasabah yang berpindah layanan ke bank lain disebut nasabah churn. Salah satu solusi yang dapat dilakukan untuk memprediksi keputusan churn seorang nasabah adalah dengan memanfaatkan teknologi AI. Random Forest dapat membantu memprediksi nasabah yang akan melakukan churn dengan akurasi yang baik. Namun dalam penggunaannya Random Forest mempunyai kekurangan, seperti setiap parameternya hanya diberikan satu nilai sehingga proses pengukuran performanya tidak akan maksimal. Untuk itu diperlukan optimasi dengan Hyperparameter Tuning dengan salah satu metode yang digunakan, yaitu GridSearchCV. Berdasarkan hasil evaluasi model yang memanfaatkan matriks seperti akurasi, recall, presisi, dan F1-Score, didapatkan persentase prediksi yang memuaskan, dengan akurasi mencapai 89.61%, recall sebesar 90.15%, presisi sebesar 88.72%, dan F1-Score sebesar 89.43%.***

Kata kunci—***Perbankan, Churn, AI, Random Forest, Hyperparameter Tuning.***

Abstract— Along with the development of technology, many bank industries compete with each other to attract people's attention. However, the tight competition has caused some banks to lose their customers. Customers who switch services to other banks are called churn customers. One solution that can be done to predict the churn decision of a customer is to utilize AI technology. Random Forest can help predict customers who will churn with good accuracy. But in its use Random Forest has shortcomings, such as each parameter is only given one value so that the performance measurement process will not be maximized. For this reason, optimization with Hyperparameter Tuning is needed with one of the methods used, namely GridSearchCV. Based on the results of model evaluation that utilizes matrices such as accuracy, recall, precision, and F1-Score, a satisfactory prediction percentage is obtained, with accuracy reaching 89.61%, recall of 90.15%, precision of 88.72%, and F1-Score of 89.43%.

Keywords—Banking, Churn, AI, Random Forest, Hyperparameter Tuning.

I. PENDAHULUAN

Seiring dengan perkembangan teknologi dan peningkatan kemampuan analisis data, banyak sektor keuangan yang bertransformasi menjadi teknologi keuangan atau *financial technology* (fintech) [1]. Hal ini menyebabkan banyaknya fasilitas layanan pengiriman uang yang disediakan oleh bank agar dapat menarik perhatian seseorang untuk menggunakan jasa mereka. Meskipun demikian, ketatnya persaingan antar-bank saat ini menyebabkan beberapa bank harus kehilangan nasabah mereka. Menurut *World Retail Banking Report* 2022, sekitar 75% nasabah perbankan yang di survey tertarik pada layanan bank dari perusahaan non-tradisional (teknologi besar atau *fintech*). Selain itu, 55% dari eksekutif bank juga melihat para pesaing non-tradisional ini di sektor keuangan, yang mana hal ini akan menjadi ancaman bagi bank tradisional [2].

Secara definisi perbankan, nasabah *churn* adalah seseorang yang menutup semua rekeningnya dan berhenti melakukan bisnis dengan bank tersebut. Solusi yang dapat dilakukan adalah dengan teknologi kecerdasan buatan atau *Artificial Intelligence* (AI) untuk memprediksi keputusan *churn* seorang nasabah. AI merupakan teknologi yang diciptakan untuk membuat sistem komputer dapat meniru kemampuan intelektual manusia. AI memiliki sub-bagian yaitu *Machine Learning* (ML). Pada kasus prediksi nasabah *churn*, penulis memanfaatkan pendekatan klasifikasi. Klasifikasi merupakan suatu pengelompokan data yang mempunyai kelas label. Banyak algoritma yang dapat digunakan untuk menyelesaikan tugas klasifikasi, salah satunya adalah *Random Forest*.

Random Forest adalah sebuah algoritma yang menggabungkan beberapa prediksi dari banyaknya keputusan yang dibangun untuk memberikan solusi terhadap masalah yang rumit dan dataset yang besar. Beberapa penelitian terdahulu telah melakukan suatu prediksi dengan hasil terbaik menggunakan *machine learning*. Namun, untuk mencapai kinerja yang optimal, *hyperparameter tuning* diperlukan untuk menyesuaikan

parameter algoritma dengan karakteristik data tertentu. *Hyperparameter tuning* adalah proses mencari nilai optimal dari parameter suatu model *machine learning* untuk memperbaiki performa model. *Hyperparameter tuning* memainkan peran kunci dalam meningkatkan kinerja algoritma, salah satunya menggunakan metode *GridSearchCV*.

Meskipun beberapa penelitian sebelumnya telah menerapkan algoritma *machine learning*, namun tidak banyak penelitian yang secara khusus memfokuskan pada optimasi *hyperparameter* menggunakan *GridSearchCV* dalam konteks industri perbankan. Penelitian ini diharapkan dapat memberikan kontribusi penting terhadap pengembangan strategi manajemen *churn* yang lebih unggul di industri perbankan. Dengan begitu industri bank dapat memahami penyebab sebenarnya dari *churn* itu sendiri dan hasil *churn* yang lebih akurat berdasarkan tingkat akurasi hasil prediksi, sehingga meminimalisir terjadinya kerugian serta risiko *churn* pada bisnis industri perbankan ke depannya.

II. TINJAUAN PUSTAKA

A. Machine Learning

Machine Learning atau yang sering disingkat ML merupakan bidang ilmu komputer yang memfokuskan pada pengembangan teknik dan algoritma yang dapat membuat komputer belajar dari data. Tujuan utama dari *Machine Learning* adalah memberikan kemampuan kepada komputer untuk mengenali pola, membuat keputusan, dan memperbaiki kinerjanya seiring waktu tanpa perlu pemrograman manusia yang eksplisit.

B. Churn

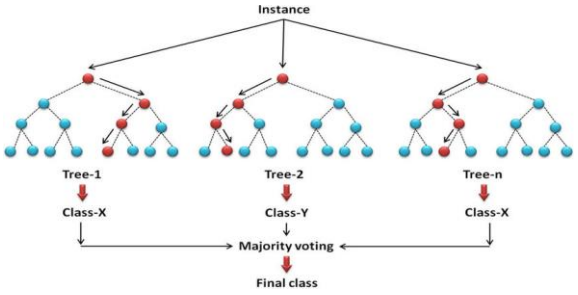
Dalam definisi perbankan sendiri, nasabah *churn* adalah orang yang menutup semua rekeningnya dan berhenti melakukan bisnis dengan bank tersebut [3]. Ada beberapa langkah yang bisa dilakukan oleh perusahaan untuk mengatasi nasabah *churn*, antara lain:

1. Retensi nasabah
2. Fokus terhadap teknologi
3. Fokus pada segmen pasar yang spesifik
4. Meningkatkan produktivitas dan efisiensi

C. Random Forest

Random Forest merupakan salah satu algoritma atau metode *ensemble* untuk klasifikasi yang termasuk dalam *supervised learning* [4]. *Random Forest* dikembangkan oleh Breiman pada tahun 2001. *Random Forest* mempunyai proses seleksi fitur yang mampu mengambil fitur terbaik untuk meningkatkan performa terhadap model klasifikasi [5]. Algoritma *Random Forest* memiliki beberapa kelebihan sebagai berikut [6]:

1. Tingkat akurasi yang cukup baik.
2. Relatif tahan terhadap *outliers* dan *noise*.
3. Lebih cepat dibangdingkan *bagging* dan *boosting*.
4. Sederhana serta mudah diparalelkan



Gambar. 1 Alur *Random Forest*

Random Forest merupakan metode pengembangan dari *decision tree* [7], di mana setiap pohon yang digunakan akan dilatih dengan sampel individu dan setiap atributnya akan dipecah pada pohon yang dipilih antara atribut sub-set data yang bersifat *random*. sedangkan *decision tree* adalah teknik yang digunakan untuk melakukan klasifikasi terhadap sekumpulan objek atau data [8].

D. Hyperparameter

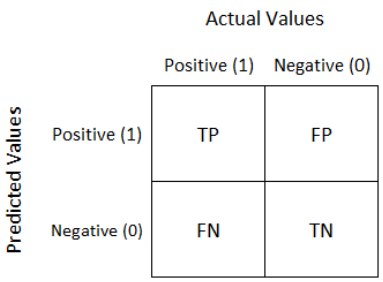
Hyperparameter merupakan penetapan nilai pada parameter sebelum proses pembelajaran dimulai [9]. Untuk mendapatkan hasil prediksi data yang sesuai, tidak hanya memerlukan model *machine learning* yang tepat, tetapi juga nilai *hyperparameter* yang tepat pula. Istilah ini dikenal dengan *hyperparameter tuning*. *Hyperparameter tuning* adalah proses mencari kombinasi nilai *hyperparameter* yang optimal untuk meningkatkan kinerja model. Tujuannya adalah untuk meningkatkan akurasi, generalisasi, atau kinerja model dengan menyesuaikan parameter-parameter yang dapat diubah.

E. GridSeacrhCV

GridSearchCV adalah salah satu metode alternatif yang digunakan untuk mencari parameter terbaik. Cara kerja dari *GridSeacrhCV* adalah dengan mencoba satu per satu kombinasi parameter dan memvalidasi setiap kombinasinya. *GridSearchCV* dapat diterapkan secara maksimum apabila batas atas dan batas bawah dari masing-masing parameter diketahui [10].

F. Confusion Matrix

Confusion Matrix merupakan model pengujian akurasi yang memberikan gambaran rinci tentang performa model. Model pengujian tersebut dapat dibandingkan dengan hasil pada beberapa skenario yang dilakukan [11].



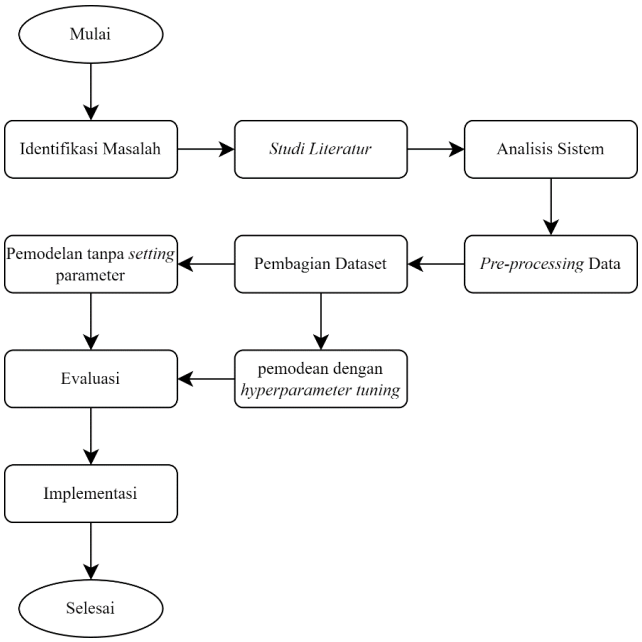
Gambar. 2 Confusion Matriks

Dalam *confusion matrix* terdapat empat istilah representasi hasil klasifikasi [12], yaitu:

1. *True Positive* (TP) :
   Memprediksi positif yang diklasifikasikan dengan benar.
2. *True Negative* (TN) :
   Memprediksi negatif yang diklasifikasikan dengan benar.
3. *False Positive* (FP) :
   Memprediksi positif yang diklasifikasikan dengan keliru.
4. *False Negative* (FN) :
   Memprediksi negatif yang diklasifikasikan dengan keliru.

III. METODOLOGI PENELITIAN

Pada tahap ini, menjelaskan tentang alur dari proses penelitian yang dilakukan. Adapun alurnya seperti pada Gambar 3 berikut:



Gambar. 3 Alur Penelitian

A. Identifikasi Masalah

TABEL I
IDENTIFIKASI MASALAH

Subjek	Pihak bank	Who
Masalah	Tingginya tingkat *churn* dan perlu adanya prediksi nasabah yang berpotensi *churn*	What
Lokasi	Industri perbankan	Where
Solusi	Mengoptimalkan prediksi *churn* dengan menggunakan algoritma *Random Forest* yang telah di-*tuning* dengan *GridSearchCV*	Why

Dalam konteks topik permasalahan, 4W merujuk pada hal-hal sebagai berikut:

1. Who : Siapa yang terlibat dalam masalah ini?
2. What : Apa masalah yang dihadapi?
3. Where : di manakah masalah diamati?
4. Why : Alasan mengapa masalah tersebut perlu untuk diselesaikan dan solusinya?

B. Studi Literatur

Pada penelitian ini, proses studi literatur dilakukan dengan mengakses beberapa situs web seperti *Google Scholar*, *Academia*, *Science Direct*, serta sumber-sumber berita aktual lainnya terkait tingginya tingkat penurunan jumlah nasabah bank.

C. Analisis Sistem

Analisis sistem dilakukan untuk mengetahui kebutuhan dalam pembuatan sistem prediksi nasabah *churn* pada industri perbankan, di antaranya sebagai berikut:

1. Kebutuhan spesifikasi komputer: Processor AMD Ryzen 7, RAM 8 GB, OS Windows
2. Aplikasi dan pustaka yang digunakan: *Visual Studio Code*, *Google Colab*, *Python*, *Numpy*, *Pandas*, *Matplotlib*, *GridSearchCV*, *Scikit-learn*, *Streamlit*, *Joblib.*

D. *Pre-processing Data*

1. Review Dataset

Data yang digunakan berasal dari situs Kaggle *Bank Customer Churn Prediction* dengan data terstruktur atau tabular sebanyak 14 kolom dan 10.000 sampel data. Dataset yang digunakan pada penelitian ini meliputi kolom-kolom sebagai berikut:

TABEL II
INFORMASI DATASET

No	Fitur	Tipe Data	Keterangan
1	*Row Number*	int64	Indikator posisi suatu baris dalam dataset.
2	*Customer ID*	int64	Identifikasi unik untuk setiap pelanggan, biasanya digunakan untuk membedakan satu pelanggan dengan pelanggan lainnya.
3	*Surname*	*object*	Informasi tentang nama pelanggan
4	*Credit Score*	int64	Informasi yang dapat mempengaruhi kemampuan seseorang untuk mendapatkan pinjaman layanan keuangan lainnya.
5	*Geography*	*object*	Informasi lokasi pelanggan berada.
6	*Gender*	*object*	Informasi jenis kelamin pelanggan.
7	*Age*	int64	Informasi usia

			pelanggan.
8	*Tenure*	int64	Informasi berapa lama pelanggan telah menjadi nasabah, biasanya diukur dalam bulan /tahun.
9	*Balance*	float64	Informasi saldo yang dimiliki oleh pelanggan dalam rekening mereka.
10	*NumOf Products*	int64	Informasi jumlah produk layanan yang dimiliki pelanggan.
11	*HasCr Card*	int64	Apakah pelanggan memiliki kartu kredit atau tidak
12	*IsActive Member*	int64	Apakah pelanggan adalah pengguna aktif atau tidak
13	*Estimated Salary*	float64	Informasi perkiraan gaji tahunan pelanggan.
14	*Exited*	int64	Menunjukan apakah pelanggan telah keluar dari layanan tertentu, seringkali digunakan dalam konteks prediksi *churn*.

2. Pembersihan Data

Proses pembersihan data meliputi pengecekan seperti apakah ada *missing value* atau kesalahan lain pada data. Namun, karena data yang dimiliki mempunyai *missing value*, maka diperlukan langkah untuk menghapus kolom yang tidak dibutuhkan pada proses klasifikasi seperti "RowNumber", "Surname", "Geography" dan "CustomerId" menggunakan pustaka Pandas.

3. Normalisasi Data

Bagian ini mengubah semua rentang nilai data kategorial dalam kolom menjadi nilai data numerik dari 0 sampai 1 menggunakan *MinMaxScaler* untuk mentransformasi nilai – nilai pada data. Seperti contoh pada kolom *Gender*, *Female* diganti dengan 0 dan *Male* diganti dengan 1. Hal ini bertujuan untuk membuat data lebih mudah diinterpretasikan, dan meningkatkan kinerja algoritma pembelajaran mesin.

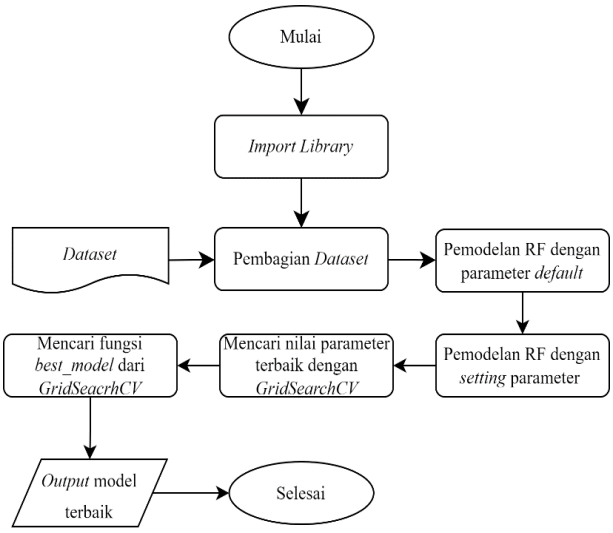
4. *Oversampling*

Karena data yang didapatkan memiliki nilai yang tidak seimbang antara nasabah yang melakukan *churn* dan tidak *churn*, maka untuk mengatasinya dapat menggunakan teknik SMOTE.

E. Pembagian Dataset

Data yang dibagi menjadi dua set, yaitu set data latih dan set data uji. Pada penelitian ini data dibagi dengan rasio 80% data latih dan 20% data uji menggunakan *train_test_split* dari pustaka *scikit-learn*.

F. Pemodelan



Gambar. 4 Klasifikasi *Random Forest*

Model awal dibangun menggunakan parameter *default* dari *RandomForestClassifier* untuk menghasilkan *baseline performance.* Selanjutnya, untuk meningkatkan akurasi model, dilakukan optimasi *hyperparameter* menggunakan *GridSearchCV* untuk mencari kombinasi parameter terbaik. Proses optimasi ini bertujuan untuk menemukan nilai parameter yang dapat memaksimalkan kinerja model.

TABEL III
PARAMETER RANDOM FOREST

n_estimators	[100, 200, 300, 400, 500]	*default* = 100
max_depth	[5, 10, 15, 20, 25, 30]	*default* = none
max_features	['auto', 'sqrt', 'log2']	*default* = 'sqrt'
min_samples_split	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	*default* = 2
min_samples_leaf	[1, 2, 3, 4, 5, 6, 7, 8, 9, 10]	*default* = 1
criterion	['gini', 'entropy', 'log_loss']	*default* = 'gini'

G. Evaluasi

Evaluasi hasil klasifikasi dalam penelitian ini memanfaatkan metrik seperti akurasi, precision, *recall*, dan F1-Score.

H. Implementasi

Setelah data di latih, selanjutnya menyimpan model algoritma yang sudah diolah dalam format Joblib agar dapat diambil oleh sistem aplikasi web. Lalu sistem dikembangkan menjadi website dengan *framework* Streamlit. Hasil prediksi akan menampilkan prediksi apakah nasabah tersebut *churn* atau tidak *churn*.

IV. HASIL DAN PEMBAHASAN

Pada bab ini menjelaskan tentang hasil dan pembahasan dalam melakukan prediksi nasabah *churn* yang mencangkup penerapan model pada python dan implementasi dari hasil pengujian model ke sistem aplikasi berbasis website.

A. Penerapan Model pada Python

1. Import Pustaka

```
#import pustaka untuk data science
import numpy as np
import pandas as pd
import sklearn
import matplotlib.pyplot as plt
import seaborn as sns
```

Gambar. 5 Pustaka Python

Kode di atas merupakan penggunaan beberapa pustaka dalam pemrograman python untuk melakukan analisis data. Pustaka yang digunakan meliputi:

a. Numpy : digunakan untuk bekerja dengan *array* dan juga memiliki fungsi yang bekerja dalam domain matriks.
b. Pandas : digunakan untuk analisis data, manipulasi data, dan pembersihan data.
c. Scikit-Learn : digunakan untuk mendukung *machine learning* dengan berbagai algoritma yang diawasi dan tidak diawasi.
d. Matplotlib : digunakan untuk mem-plot angka-angka berdefinisi tinggi seperti diagram batang.
e. Seaborn : digunakan untuk membuat grafik statistik yang menarik dan informatif.

2. Import Dataset

Selanjutnya mengupload dataset ke Github dan mengimportnya seperti gambar di bawah ini:

```
# URL file CSV dari repositori GitHub
url =
'https://raw.githubusercontent.com/nisaaml
y/Data-Churn/main/Churn_Modelling.csv'
# Membaca file CSV dari URL
df = pd.read_csv(url, delimiter=',')
```

Gambar. 6 Import Data

3. Pembersihan Data

```
#membuang kolom/fitur yang tidak di
perlukan
df.drop(["RowNumber","CustomerId","Geograp
hy","Surname"],axis=1, inplace=True)
```

Gambar. 7 Pembersihan Data

4. Normalisasi Data

Langkah selanjutnya mengubah data kategorikal menjadi angka *integer*, seperti kolom ‘Gender’ di bawah ini:

```
# Membuat objek LabelEncoder
label_encoder = LabelEncoder()

# Melakukan label encoding untuk kolom
'Gender'
df['Gender'] =
label_encoder.fit_transform(df['Gender'])
```

Gambar. 8 Mengubah data kategorikal ke integer

Lalu mengubah semua rentang nilai data numerikal menjadi antara 0 dan 1 menggunakan *MinMaxScaler*.

```
# Membuat objek MinMaxScaler
scaler = MinMaxScaler()

# Memilih fitur-fitur yang akan di-scaling
(kecuali target 'Exited')
features_to_scale = ['CreditScore',
'Gender', 'Age', 'Tenure', 'Balance',
'NumOfProducts', 'HasCrCard',
'IsActiveMember', 'EstimatedSalary']

# Melakukan penskalaan fitur-fitur
tersebut
df[features_to_scale] =
scaler.fit_transform(df[features_to_scale]
)

# Simpan objek LabelEncoder
joblib.dump(label_encoder,
'label_encoder.joblib')
# Simpan objek MinMaxScaler
joblib.dump(scaler,
'min max scaler.joblib')
```

Gambar. 9 Membuat objek dengan MinMaxScaler

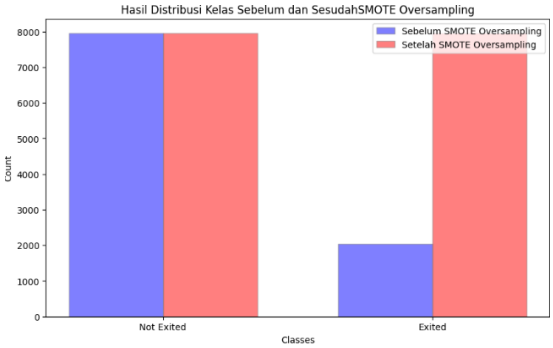
5. Oversampling

Setelah di ubah menjadi nilai numerikal, karena data yang didapatkan memiliki nilai yang tidak seimbang maka untuk mengatasinya dapat menggunakan teknik SMOTE

```
# Memisahkan fitur dan target
X = df.drop('Exited', axis=1)
y = df['Exited']

# Membuat objek SMOTE
smote = SMOTE()

# Melakukan oversampling menggunakan SMOTE
X_resampled, y_resampled =
smote.fit_resample(X, y)
```

Gambar. 10 Membuat objek dengan SMOTE

Lalu membuat program visualisasi diagram batang untuk membandingkan distribusi kelas sebelum dan setelah melakukan *oversampling* dengan teknik SMOTE pada kelas 'Exited.



Gambar. 11 Visualisasi perbandingan fitur 'Exited'

6. Pembagian Dataset

```
# Membagi dataset menjadi subset pelatihan
(80%) dan pengujian (20%)
X train, X test, y train, y test =
train_test_split(X_resampled, y_resampled,
test_size=0.2, random_state=42)

# Menampilkan ukuran dari masing-masing
subset
print("Jumlah data pelatihan:",
len(X_train))
print("Jumlah data pengujian:",
len(X_test))
```

Gambar. 12 Pembagian Dataset

Kode program tersebut menampilkan jumlah data latih dan data uji yang telah dibagi menjadi rasio 80:20, hasilnya yaitu 12.740 data latih dan 3.186 data uji.

7. Pemodelan

```
# Inisialisasi model Random Forest
random_forest_sklearn =
RandomForestClassifier(n_estimators=100,
max_depth=None, max_features=None)

# Melatih model menggunakan data pelatihan
random_forest_sklearn.fit(X_train, y_train

# Melatih model menggunakan data pelatihan
random_forest_sklearn.fit(X_train, y_train

# Membuat prediksi pada data pelatihan dan
data uji
y_train_pred_sklearn =
random_forest_sklearn.predict(X_train)
y_test_pred_sklearn =
random_forest_sklearn.predict(X_test)

# Evaluasi model menggunakan berbagai
metrik evaluasi
print("Metrics on Training Data:")
print("Accuracy:", accuracy_score(y_train,
y_train_pred_sklearn))
print("Recall:", recall_score(y_train,
y_train_pred_sklearn))
print("Precision:",
precision score(y train,
y_train_pred_sklearn))
print("F1-score:", f1_score(y_train,
y_train_pred_sklearn))
print("\nMetrics on Test Data:")
print("Accuracy:", accuracy_score(y_test,
y_test_pred_sklearn))
print("Recall:", recall_score(y_test,
y_test_pred_sklearn))
print("Precision:",
precision_score(y_test,
y_test_pred_sklearn))
print("F1-score:", f1_score(y_test,
y_test_pred_sklearn))
```

Gambar. 13 pemodelan tanpa *setting* parameter

```
# Inisialisasi model
RandomForestClassifier
random_forest_sklearn =
RandomForestClassifier()

# Tentukan grid parameter yang akan
dieksplorasi
param_grid = {
    'n_estimators': [50, 100, 150],
    'max_depth': [None, 5, 10],
    'max_features': [None, 'sqrt',
'log2'],
    'min_samples_split': [2, 5, 10],
    'min_samples_leaf': [1, 2, 4],
    'criterion': ['gini', 'entropy']
}

# Inisialisasi GridSearchCV
grid_search =
GridSearchCV(random_forest_sklearn,
param_grid, cv=5, scoring='accuracy')

# Lakukan pencarian parameter terbaik
grid_search.fit(X_train, y_train)

# Tampilkan parameter terbaik
print("Best Parameters:",
grid_search.best_params_)

# Buat prediksi dengan model terbaik
best model = grid search.best estimator
y_train_pred = best_model.predict(X_train)
y_test_pred = best_model.predict(X_test)

# Evaluasi model pada data latih
print("\nMetrics on Training Data:")
print("Accuracy:", accuracy score(y train,
y_train_pred))
print("Recall:", recall_score(y_train,
y_train_pred))
print("Precision:",
precision_score(y_train, y_train_pred))
print("F1-score:", f1 score(y train,
y_train_pred))

# Evaluasi model pada data uji
print("\nMetrics on Test Data:")
print("Accuracy:", accuracy_score(y_test,
y_test_pred))
print("Recall:", recall_score(y_test,
y_test_pred))
print("Precision:",
precision_score(y_test, y_test_pred))
print("F1-score:", f1_score(y_test,
y_test_pred))
```

Gambar. 14 pemodelan dengan *setting* parameter

8. Evaluasi

```
Metrics on Training Data:
Accuracy: 1.0
Recall: 1.0
Precision: 1.0
F1-score: 1.0

Metrics on Test Data:
Accuracy: 0.8848085373509103
Recall: 0.8634900193174501
Precision: 0.8963903743315508
F1-score: 0.8796326664480157
```

Gambar. 15 hasil evaluasi tanpa *setting* parameter

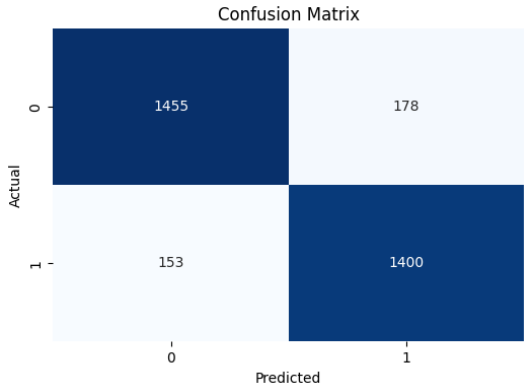
```
Metrics on Training Data:
Accuracy: 1.0
Recall: 1.0
Precision: 1.0
F1-score: 1.0

Metrics on Test Data:
Accuracy: 0.8961079723791588
Recall: 0.901481004507405
Precision: 0.8871989860583016
F1-score: 0.8942829766847652
```

Gambar. 16 hasil evaluasi dengan *setting* parameter

Dapat disimpulkan perbandingan dari matriks evaluasi sebelum dan setelah *setting* parameter adalah sebagai berikut:

a. Akurasi : Setelah *setting* parameter, akurasi model meningkat dari 88.48% menjadi 89.61%. Hal ini menunjukkan bahwa *setting* parameter telah berhasil meningkatkan kemampuan model untuk meng-klasifikasikan data.
b. *Recall* : Setelah *setting* parameter, *recall* juga meningkat dari 86.35% menjadi 90.15%. Hal ini menunjukkan bahwa setelah *setting* parameter, model lebih baik dalam mengidentifikasi jumlah positif sebenarnya dari semua kasus positif.
c. Presisi : Presisi sebelumnya 89.64% dan meningkat menjadi 88.72% setelah *setting* parameter. Meskipun ada penurunan sedikit dalam presisi, tetapi tetap berada pada level yang tinggi. Hal ini menunjukkan bahwa model masih bisa melakukan klasifikasi yang baik terhadap kasus positif.
d. F1-Score : Sebelum *setting* parameter nilai dari F1-Score adalah 87.96% dan meningkat menjadi 89.43%. Hal ini menunjukkan bahwa setelah *setting* parameter, model memiliki keseimbangan yang lebih baik antara *presisi* dan *recall* dalam melakukan klasifikasi.

Setelah proses optimasi selesai, selanjutnya akan dilakukan perhitungan dan visualisasi data dari *confusion matrix* berupa *heatmap* menggunakan pustaka *seaborn* dan matplotlib.



Gambar. 17 hasil confusion matriks

Dapat disimpulkan berdasarkan hasil nilai akurasinya, yaitu:

a. *True Positives* (TP) = 1455, artinya model telah benar memprediksi 1455 sampel positif.
b. *False Positives* (FP) = 153, artinya model telah salah memprediksi 153 sampel positif.
c. *False Negatives* (FN) = 178, artinya model telah salah memprediksi 178 sampel negatif.
d. *True Negatives* (TN) = 1400, artinya model telah benar memprediksi 1400 sampel negatif.

Selanjutnya model yang telah dilatih akan disimpan ke dalam file Joblib seperti gambar di bawah ini:

```
# Simpan model Random Forest yang telah
di-tuning
import joblib
model_filename =
'random forest model.joblib'
joblib.dump(best_model, model_filename)
print(f"Model Random Forest telah disimpan
sebagai '{model_filename}'")

# Memuat kembali model Random Forest
loaded model = joblib.load(model filename)
model filename =
'random_forest_model.joblib'
```

Gambar. 18 Menyimpan model

B. Implementasi Model pada Sistem Aplikasi

1. Halaman ‘Tentang’

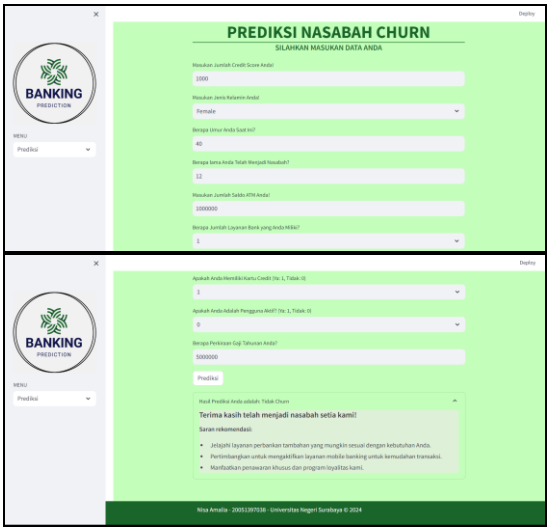
Gambar. 19 halaman ‘Tentang’

Tahap awal pada website akan menampilkan fitur 'Tentang', di mana pada bagian sidebar terdapat

logo dari aplikasi web dan dua fitur yang ditampilkan pada aplikasi web tersebut, yaitu fitur ‘tentang’ dan fitur ‘prediksi’. Selanjutnya di samping kanan, sidebar terdapat header dan di bawahnya terdapat teks yang berisi tentang penjelasan singkat mengenai aplikasi web. Lalu di halaman paling bawah terdapat footer.

2. Halaman ‘Prediksi’

Pada halaman ‘Prediksi’, *user* diwajibkan untuk mengisi 9 variabel yang tersedia untuk mengidentifikasi nasabah yang berpotensi meninggalkan layanan bank. Terdapat dua hasil prediksi sebagai *output* dari percobaan pada penelitian ini. Percobaan pertama dilakukan dengan hasil prediksi ‘Tidak *Churn’*.

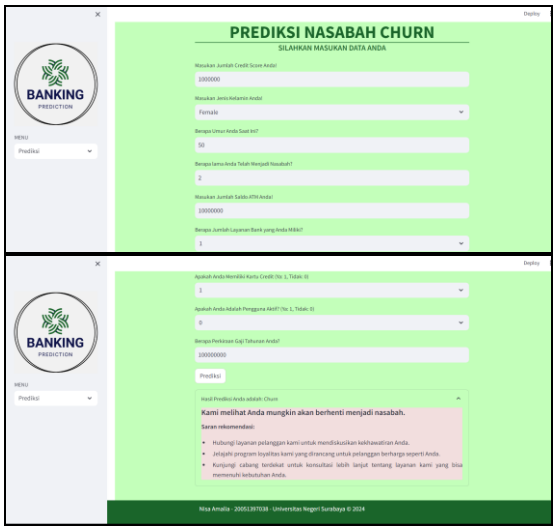


Gambar. 20 halaman ‘Prediksi’ tidak *churn*

Berdasarkan contoh percobaan pertama yang dilakukan setelah mengisi seluruh variabel, hasil didapatkan bahwa nasabah tersebut tidak *churn*. Hal ini dapat dianalisis bahwa dengan *credit score* yang tinggi, saldo ATM yang besar, dan gaji yang tinggi setiap tahunnya, menunjukkan stabilitas keuangan dan kepercayaan nasabah terhadap penggunaan bank yang dimilikinya. Lalu ketika nasabah menggunakan kartu kredit dapat dilihat juga bahwa nasabah memiliki tingkat keterlibatan yang tinggi pada layanan bank tersebut. Selanjutnya nasabah juga merupakan pengguna aktif dari bank tersebut, hal ini membuktikan bahwa nasabah merasa puas dengan layanan bank yang diberikan.

Selain itu juga terdapat saran atau rekomendasi yang diberikan ketika nasabah diprediksi tidak *churn*. Menambahkan saran atau rekomendasi untuk nasabah yang tidak *churn* dapat membantu perusahaan meningkatkan keterlibatan dan loyalitas pelanggan melalui promosi layanan tambahan atau program loyalitas, sehingga memperkuat hubungan jangka panjang dengan nasabah.

Selanjutnya pada percobaan kedua dilakukan dengan hasil prediksi ‘*Churn’*.



Gambar. 21 halaman ‘Prediksi’ tidak *churn*

Berdasarkan percobaan kedua yang telah dilakukan setelah mengisi seluruh variabel, hasil didapatkan bahwa nasabah tersebut *churn*. Meskipun nasabah memiliki *credit score* yang sangat tinggi, saldo ATM besar, jangka waktu yang lama menggunakan bank, nasabah sebagai pengguna aktif, dan gaji tahunan yang tinggi, namun prediksi *churn* menunjukkan bahwa pengguna tersebut berpotensi meninggalkan layanan bank. Hal ini dapat terjadi disebabkan oleh kemungkinan umur yang sudah mencapai 50 tahun bisa menandakan bahwa nasabah tersebut sudah mendekati masa pensiun.

Oleh karena itu, pada usia tersebut, keputusan untuk melakukan *churn* dalam layanan bank juga bisa dipengaruhi oleh strategi investasi dan perencanaan keuangan yang lebih luas. Saran atau rekomendasi yang diberikan berdasarkan hasil prediksi *churn* dapat memberikan nilai tambah dengan memungkinkan perusahaan untuk mengambil langkah-langkah proaktif untuk mempertahankan pelanggan yang berisiko *churn* dan meningkatkan keterlibatan pelanggan yang setia.

C. Pengujian Sistem

Testing yang dilakukan pada sistem aplikasi website ini adalah dengan *Black Box Testing*. Dengan *Black Box Testing* pengujian dilakukan untuk meng-identifikasi kesalahan, cacat, atau ketidaksesuaian antara perilaku. Hal ini membantu memastikan bahwa sistem yang dihasilkan memenuhi kebutuhan pengguna dengan spesifikasi yang telah ditetapkan. Berikut merupakan tabel hasil pengujian sistem.

TABEL IV
BLACK BOX TESTING

Pengujian	Inputan	Hasil yang diharapkan	Hasil yang didapat
Input valid	Nilai valid untuk semua kolom	Hasil prediksi muncul	Sesuai
Input tidak valid	Nilai tidak valid (misal: huruf untuk usia)	Pesan kesalahan muncul ketika klik 'prediksi'	Sesuai
Respon waktu prediksi	Nilai valid untuk semua kolom	Hasil prediksi dalam waktu yang diterima (< 3 detik)	Sesuai

V. KESIMPULAN DAN SARAN

A. Kesimpulan

Berdasarkan penelitian yang dilakukan, maka dapat ditarik kesimpulan sebagai berikut:

1. Penerapan algoritma *Random Forest* untuk prediksi nasabah *churn* pada industri perbankan dilakukan dengan me-*review* dataset, pembersihan data, melakukan normalisasi dan teknik SMOTE, pembagian data menjadi set pelatihan dan pengujian, pelatihan model *random forest* menggunakan set pelatihan, dan melakukan evaluasi dengan *confusion matrix*.
2. Performa algoritma *random forest* yang di optimasi dengan *hyperparameter tuning* menggunakan *GridSearchCV* untuk prediksi nasabah *churn* pada industri perbankan yang dilakukan dengan pencarian sistematis melalui berbagai kombinasi *hyperparameter*, seperti *n_estimators*, *max_depth, max_features, min_samples_split, min_samples_leaf, criterion* memberikan hasil prediksi terbaik dengan akurasi 89.61%, *recall* sebesar 90.15%, presisi sebesar 88.72%, dan F1-*Score* sebesar 89.43%. Dengan menyesuaikan nilai parameter algoritma secara sistematis, *GridSearchCV* membantu menemukan kombinasi *hyperparameter* terbaik untuk meningkatkan akurasi dan konsistensi model dalam memprediksi nasabah *churn*.

B. Saran

Penelitian ini jauh dari kata sempurna, sehingga perlu adanya perbaikan dan pengembangan lebih lanjut untuk keberhasilan dalam melakukan prediksi nasabah *churn* pada industri perbankan. Beberapa saran yang diberikan sebagai berikut:

1. Mengembangkan sistem web dengan berbagai fitur yang lebih banyak dan relevan.
2. Melanjutkan dan menerapkan teknik optimasi yang berbeda dalam penggunaan setiap model agar dapat mengetahui dan membandingkan serta mengevaluasi kinerja dari masing-masing model yang digunakan.
3. Menambahkan variabel penelitian yang dibutuhkan sesuai ketentuan bank yang berlaku, agar proses dalam melakukan prediksi lebih mendetail.
4. Menerapkan standar keamanan data yang relevan, seperti kebijakan, prosedur, dan kontrol keamanan yang ketat untuk melindungi data nasabah

REFERENSI

[1] Pratiwi, A. R., & Dermawan, D. A. (2021). Pengaruh Customer Relationship Management (CRM) terhadap Loyalitas Pelanggan dengan Kepuasan Pelanggan sebagai Variabel Intervening (Studi Pelanggan ShopeePay pada Aplikasi Shopee di Kota Surabaya). *Journal of Emerging Information System and Business Intelligence (JEISBI)*, 2(3), 87-93.

[2] De Lima Lemos, R. A., Silva, T. C., & Tabak, B. M. (2022). Propension to customer churn in a financial institution: A machine learning approach. *Neural Computing and Applications*, 34(14), 11751-11768.

[3] Chitra, K., & Subashini, B. (2011). Customer retention in banking sector using predictive data mining technique. In *ICIT 2011 The 5th International Conference on Information Technology*.

[4] Nudin, S. R., Warsito, B., & Wibowo, A. (2022, July). Impact of soft skills competencies to predict graduates getting jobs using random forest algorithm. In *2022 1st International Conference on Information System & Information Technology (ICISIT)* (pp. 49-54). IEEE.

[5] Devella, S., Yohannes, Y., & Rahmawati, F. N. (2020). Implementasi Random Forest Untuk Klasifikasi Motif Songket Palembang Berdasarkan SIFT. *JATISI (Jurnal Teknik Informatika dan Sistem Informasi)*, 7(2), 310-320.

[6] Gusnina, M., & Salamah, U. (2022). Student Performance Prediction in Sebelas Maret University Based on the Random Forest Algorithm. *Ingenierie des Systemes d'Information*, 27(3), 495.

[7] Zhou, X., Lu, P., Zheng, Z., Tolliver, D., & Keramati, A. (2020). Accident prediction accuracy assessment for highway-rail grade crossings using random forest algorithm compared with decision tree. *Reliability Engineering & System Safety*, 200, 106931.

[8] Amalda, R. N. (2021). *Penerapan Algoritma C5. 0 pada Sistem Pengambilan Keputusan Guna Melihat Kelayakan Penerima Keringanan UKT Mahasiswa ITK* (Doctoral dissertation, Institut Teknologi Kalimantan).

[9] Ghawi, R., & Pfeffer, J. (2019). Efficient hyperparameter tuning with grid search for text categorization using KNN approach with BM25 similarity. *Open Computer Science*, 9(1), 160-180.

[10] Ramadhan, M. M., Sitanggang, I. S., Nasution, F. R., & Ghifari, A. (2017). Parameter tuning in random forest based on grid search method for gender classification based on voice frequency. *DEStech transactions on computer science and engineering*, 10(2017).

[11] Sari, V. R., Firdausi, F., & Azhar, Y. (2020). Perbandingan Prediksi Kualitas Kopi Arabika dengan Menggunakan Algoritma SGD, Random Forest dan Naive Bayes. *Edumatic: Jurnal Pendidikan Informatika*, 4(2), 1-9.

[12] Nurhidayat, A., Asmunin, A., & Suyatno, D. F. (2021). Prediksi Kinerja Akademik Mahasiswa Menggunakan Machine Learning dengan Sequential Minimal Optimization untuk Pengelola Program Studi. *JIEET (Journal of Information Engineering and Educational Technology)*, 5(2), 84-91.