Optimasi Algoritma *Random Forest* dengan Teknik *Boosting* dalam Prediksi *Churn* Pelanggan di Industri Telekomunikasi

Rizka Nurul Septiani Hakim[1], Asmunin[2]

Manajemen Informatika, Universitas Negeri Surabaya
Jl. Ketintang, Ketintang, Kec. Gayungan, Kota Surabaya, Jawa Timur 60231

[1]rizka.20026@mhs.unesa.ac.id

[2]asmunin@unesa.ac.id

Abstrak*— Seiring munculnya jaringan 5G membuat persaingan di industri telekomunikasi semakin meningkat. Hal ini dapat mengakibatkan tren churn pelanggan semakin tinggi. Churn, atau kehilangan pelanggan, adalah isu utama yang memengaruhi keberhasilan suatu perusahaan. Oleh karena itu, churn dapat dikurangi dengan membuat model prediksi untuk mengidentifikasi pelanggan yang berpotensi churn menggunakan algoritma machine learning yaitu Random Forest. Namun, masih ada ruang untuk meningkatkan kinerjanya, dengan optimasi menggunakan teknik boosting maka dapat meningkatkan akurasi dan performa model prediksi. Teknik boosting yang digunakan yaitu Adaboost dan XGBoost, kemudian hasil dibandingkan untuk menentukan model boosting dengan akurasi lebih tinggi yang akan diterapkan pada aplikasi. Aplikasi berbasis web dibangun untuk memprediksi churn pelanggan dengan memasukkan data pelanggan sesuai variabel dan mengunggah file dataset dengan format .csv. Hasil penelitian ini menunjukkan bahwa Algoritma Random Forest yang dioptimasi dengan teknik boosting Adaboost mampu memprediksi churn pelanggan dengan tingkat akurasi yang tinggi yaitu 95.20%.

***Kata kunci*— Random Forest, Teknik Boosting, Churn Pelanggan, Industri Telekomunikasi, Web.**

Abstract*— Along with the emergence of 5G networks, competition in the telecommunications industry is increasing. This can result in higher customer churn trends. Churn, or customer loss, is a major issue that affects the success of a company. Therefore, churn can be reduced by creating a prediction model to identify potentially churned customers using the machine learning algorithm Random Forest. However, there is still room to improve its performance, with optimization using boosting techniques that can improve the accuracy and performance of the prediction model. The boosting techniques used are Adaboost and XGBoost, then the results are compared to determine the boosting model with higher accuracy that will be applied to the application. A web-based application was built to predict customer churn by entering customer data according to variables and uploading dataset files in .csv format. The results of this study show that the Random Forest Algorithm optimized with the Adaboost boosting technique is able to predict customer churn with a high accuracy rate of 95.20%.

***Keywords*— Random Forest, Boosting Technique, Customer Churn, Telecommunication Industry, Web.**

I. PENDAHULUAN

Berkembangnya jaringan internet pada teknologi komunikasi dari 4G ke 5G membawa perubahan signifikan dalam pola perilaku pengguna layanan internet [1]. Laporan *We Are Social* menyatakan bahwa "jumlah pengguna internet di Indonesia mencapai 213 juta orang per Januari 2023. Jumlah ini mengalami peningkatan sebesar 5,44% dibandingkan tahun sebelumnya (*year-on-year/yoy*)". Sehubungan meningkatnya pengguna internet yang diiringi munculnya jaringan 5G, mengakibatkan meningkatnya persaingan antar perusahaan di industri telekomunikasi dalam berlomba-lomba untuk menyediakan kualitas layanan terbaik dan jaringan 5G kepada pelanggan. Sehingga semakin banyak perusahaan telekomunikasi bersaing untuk mempertahankan pengguna layanan mereka yang masih aktif dan menarik pengguna baru yang berpotensi menjadi pelanggan. Meningkatnya persaingan antar penyedia jasa telekomunikasi dapat menjadi faktor perpindahan pelanggan atau yang sering disebut *churn*. *Churn* pelanggan adalah proses peralihan pelanggan dari satu perusahaan ke perusahaan lain dalam periode waktu tertentu [2]. Salah satu masalah utama dalam perushaan telekomunikasi adalah *Churn* pelanggan, terutama dalam konteks layanan internet. Adanya *churn* pelanggan dapat mengakibatkan kerugian besar bagi perusahaan telekomunikasi, baik dari segi pendapatan yang hilang maupun biaya yang harus dikeluarkan untuk mendapatkan pelanggan baru.

Dengan kemajuan teknologi dan analisis data, perusahaan telekomunikasi saat ini dapat memprediksi *churn* pelanggan dengan lebih efektif dan efisien. Pendekatan yang terbukti lebih efektif dalam memprediksi *churn* pelanggan adalah menggunakan metode *Machine Learning*. Contoh Algoritma *Machine Learning* seperti *Random Forest*, *AdaBoost* dan *XGBoost*. *Random Forest* memiliki kinerja yang tinggi karena dapat menggabungkan prediksi dari banyak pohon (*decision trees*), mampu menangani data yang besar dengan banyak variabel, dan beragam tipe data (numerik atau kategorial) tanpa memerlukan *pre-processing* data yang rumit [3]. Kekuatan *Random Forest* dalam memberikan prediksi

yang akurat juga membuatnya kurang *interpretabel*. Karena model terdiri dari banyak pohon keputusan, sehingga sulit untuk memahami secara detail bagaimana setiap keputusan diambil. *Random Forest* bisa mengalami *overfitting* atau keadaan di mana model *machine learning* mempelajari data pelatihan dengan terlalu baik, sehingga tidak dapat menggeneralisasi data baru dengan baik terutama jika terlalu banyak pohon (n_estimators) yang digunakan atau jika tidak ada parameter yang diatur dengan baik [4].

Teknik Boosting memiliki keunggulan dalam mengatasi rendahnya tingkat *interpretable* dengan optimasi menggunakan *Adaboost* dan *XGBoost* karena cenderung menghasilkan model yang lebih sederhana dan mudah diinterpretasikan. Model *Boosting* yang dibangun sederhana tetapi tidak menurunkan performa *Random Forest* sebelumnya dapat dilakukan dengan mengatur parameter model. Parameter dapat membantu mengontrol *overfitting* (ketika model terlalu menyesuaikan dengan data pelatihan) atau underfitting (ketika model terlalu sederhana untuk memahami pola yang ada dalam data) [5]. *Adaboost* bekerja dengan membangun serangkaian model lemah (weak learners), biasanya pohon keputusan dangkal, dan menggabungkannya menjadi model kuat. Sedangkan *XGBoost* bekerja dengan memberikan bobot lebih tinggi pada data yang terklasifikasi salah pada sebelumnya, sehingga memfokuskan pembelajaran pada data yang sulit untuk diprediksi [6]. Dalam proses teknik *boosting* ini, *Adaboost* dan *XGBoost* dipilih sebagai metode untuk mengoptimalkan kinerja algoritma *Random Forest*.

Setelah mendapatkan model prediksi yang telah dioptimasi, selanjutnya membangun aplikasi berbasis web menggunakan *framework flask* untuk implementasi dan penerapan praktis model tersebut. Aplikasi berbasis web akan memungkinkan perusahaan telekomunikasi untuk melakukan prediksi *churn* pelanggan secara *real-time* sehingga dapat melakukan tindakan pencegahan yang tepat untuk mempertahankan pelanggan yang berisiko untuk *churn*. Oleh karena itu, penelitian ini akan mengoptimasi algoritma *Random Forest* menggunakan teknik *Boosting* sehingga dapat meningkatkan akurasi prediksi *churn* dan menerapkan dalam aplikasi berbasis web untuk membantu perusahaan telekomunikasi dalam mempertahankan pelanggan dengan lebih efektif. Penelitian ini bertujuan meningkatkan akurasi model prediksi menggunakan teknik *Boosting* dan mengukur keberhasilan optimasi algoritma *Random Forest* menggunakan teknik *Boosting* dengan membuat metrik evaluasi yaitu *confusion matrix* dan AUC-ROC. Dengan demikian, hasil dari penelitian ini diharapkan akan memberikan wawasan yang berharga kepada perusahaan telekomunikasi dalam mengidentifikasi pelanggan yang berpotensi *churn* dengan mengambil tindakan pencegahan yang sesuai untuk mempertahankan pelanggan mereka, dapat meningkatkan kualitas layanan, dan meningkatkan kinerja bisnis perusahaan telekomunikasi di era yang penuh persaingan ini.

II. TINJAUAN PUSTAKA

A. *Machine Learning*

Machine Learning atau Mesin Pembelajar merupakan sebuah cabang dari kecerdasan buatan yang berfokus pada pembelajaran dari data *(learn from data)*, yaitu mengembangkan sistem yang dapat belajar secara mandiri tanpa perlu diprogram kembali oleh manusia [7]. Terdapat tiga cabang utama dalam *machine learning* yaitu :

1. *Supervised Learning* merupakan model *machine learning* yang proses pembelajarannya dibawah pengawasan. Model ini diberikan data yang telah diberi label sebelumnya, yang berarti data tersebut memiliki pasangan *input* dan *output* yang benar. Contoh algoritma ini adalah *Random Forest, Decision Tree, XGBoost, AdaBoost,* dll.
2. *Unsupervised Learning* merupakan model *machine learning* yang proses pembelajarannya tanpa pengawasan. Model ini diberikan data yang tidak diberi label, yang berarti tidak ada informasi eksplisit tentang *output* yang benar. Contoh algoritma ini adalah *K-means clustering, hierarchical clustering,* dan algoritma *Apriori*.
3. *Reinforcement learning* merupakan mesin atau perangkat lunak yang terus melatih dirinya berdasarkan lingkungannya untuk memecahkan masalah bisnis. Proses belajar yang terus-menerus ini mengurangi keterlibatan manusia sehingga menghemat banyak waktu. Contoh algoritma ini adalah *real-time decision, robot navigation, learning tasks, skills acquisition* dam game AI.

B. *Algoritma Random Forest*

Algoritma *Random Forest* adalah salah satu algoritma *supervised learning* dalam *machine learning* untuk menangani masalah klasifikasi, regresi, dan pengurutan data. Algoritma ini didasarkan pada konsep *ensemble learning*, di mana beberapa model pembelajaran disatukan untuk menghasilkan prediksi yang akurat. Konsep dasar dari *Random Forest* adalah menggunakan sekumpulan pohon keputusan (*decision trees*) yang dibangun secara acak dan independen satu sama lain. Setiap pohon dalam *Random Forest* dibangun dengan membagi dataset menjadi subset yang acak dan menggunakan subset tersebut sebagai training data. Proses pembagian dilakukan dengan memperhatikan fitur-fitur acak, sehingga setiap pohon memiliki keberagaman dalam struktur dan fitur yang digunakan dalam keputusan. Selama tahap prediksi, setiap pohon dalam *Random Forest* memberikan prediksi, dan prediksi akhir ditentukan berdasarkan mayoritas suara (klasifikasi) atau rata-rata (regresi) dari prediksi yang diberikan oleh semua pohon. Penggunaan *ensemble* dari pohon-pohon ini membantu mengurangi *overfitting* dan meningkatkan akurasi prediksi [8].

C. Teknik Boosting

Boosting adalah salah satu teknik dalam *machine learning* yang digunakan untuk meningkatkan kinerja model prediksi dengan menggabungkan beberapa model yang lemah menjadi model yang kuat. Tujuan utama dari *boosting* adalah mengurangi *bias* dan kesalahan pada prediksi model [6].

1. Adaboost

   Adaboost (*Adaptive Boosting*) dikembangkan oleh Yoav Freund dan Robert Schapire pada tahun 1996. *Adaboost* bekerja dengan memberikan bobot pada setiap sampel data dan memfokuskan pelatihan pada sampel yang sulit diklasifikasikan. Tujuan utama *Adaboost* adalah menggabungkan beberapa model prediksi yang lemah menjadi satu model yang kuat [6].

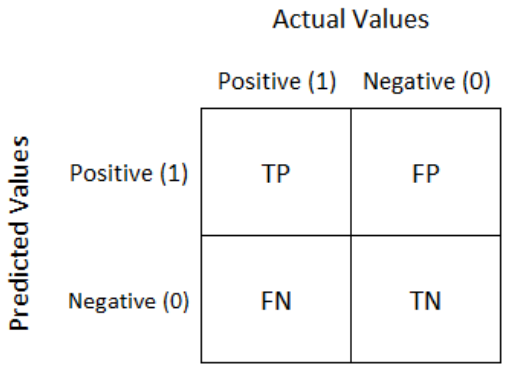
2. XGBoost

   XGBoost (*Extreme Gradient Boosting*) dikembangkan oleh Tianqi Chen pada tahun 2016 sebagai ekstensi dari algoritma *Gradient Boosting*. Algoritma ini efektif dalam membangun serangkaian model prediksi lemah dan menggabungkannya menjadi model yang lebih kuat. Metode ini dapat membantu mengoptimalkan algoritma dengan fleksibel, terutama dalam hal regresi, klasifikasi, dan perangkingan. Untuk menghindari *overfitting*, *XGBoost* membantu kelancaran bobot terakhir yang dipelajari [8].

D. Churn Pelanggan

Churn pelanggan merupakan istilah yang digunakan dalam industri telekomunikasi untuk menggambarkan keadaan ketika pelanggan secara aktif berhenti menggunakan layanan atau produk dari suatu perusahaan dan beralih ke penyedia layanan atau produk lain. Penyebab *churn* dapat bervariasi, termasuk harga yang tinggi, kualitas layanan yang rendah, kegagalan produk/layanan, ketidakpuasan pelanggan, dan kurangnya nilai tambah. *Churn* pelanggan mencerminkan tingkat perputaran pelanggan dalam sebuah perusahaan, dan tingkat *churn* yang tinggi dapat berdampak negatif terhadap pendapatan, keuntungan, dan reputasi perusahaan tersebut [2].

E. Confusion Matrix



Gambar . 1 Confusion Matrix

Confusion matrix membantu dalam mengukur performa prediksi model dengan memberikan informasi tentang tingkat keakuratan, presisi, recall, dan tingkat kesalahan klasifikasi model. Dari *confusion matrix*, dapat menghitung berbagai metrik evaluasi seperti akurasi, presisi, recall (sensitivitas atau *True Positive Rate*), *spesifisitas* (*True Negative Rate*), dan *F1-score* yang memberikan gambaran komprehensif tentang performa model. *Confusion matrix* terdiri dari empat komponen utama [6]:

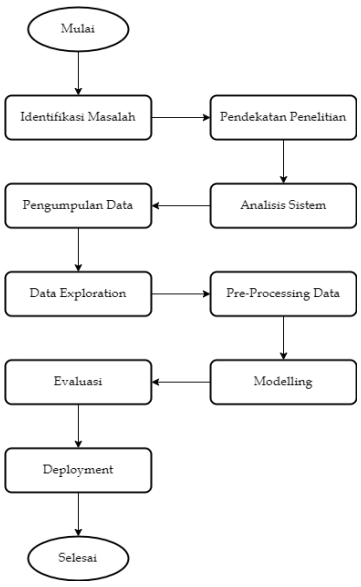
- True Positive (TP) : Jumlah sampel yang benar diprediksi sebagai positif oleh model.
- True Negative (TN) : Jumlah sampel yang benar diprediksi sebagai negatif oleh model.
- False Positive (FP) : Jumlah sampel yang salah diprediksi sebagai positif oleh model (disebut juga sebagai kesalahan Type I).
- False Negative (FN) : Jumlah sampel yang salah diprediksi sebagai negatif oleh model (disebut juga sebagai kesalahan Type II).

Dari *confusion matrix*, terdapat kemampuan untuk menghitung berbagai metrik evaluasi, termasuk akurasi, presisi, recall (sensitivitas atau *True Positive Rate*), dan F1-score yang memberikan gambaran menyeluruh tentang kinerja model.

F. Flask

Pengembangan aplikasi berbasis web menggunakan Flask, yaitu sebuah kerangka kerja web yang ditulis dalam bahasa Python sebagai struktur dasar aplikasi dan tampilan web. Flask dikategorikan sebagai *microframework* karena tidak memerlukan alat atau pustaka tertentu untuk penggunaannya. Dengan bantuan Flask dan bahasa Python, pengembangan dapat membuat situs web yang lebih terstruktur dan mengatur situs web dengan lebih efisien [9].

III. METODE PENELITIAN



Gambar . 2 Metodologi Penelitian

A. *Identifikasi Masalah*

Seiring berkembangnya jaringan 5G membuat tren *churn* pelanggan di industri telekomunikasi semakin tinggi. Hal ini dapat diatasi dengan membuat model prediksi untuk mengidentifikasi *churn* pelanggan dengan memanfaatkan *machine learning* pendekatan *supervised learning* yaitu algoritma *Random Forest*. Untuk meningkatkan akurasi model prediksi *churn* pelanggan perlu dilakukan optimasi algoritma *Random Forest* dengan teknik *Boosting*. Hal ini perlu dilakukan karena *Random Forest* terdiri dari banyak pohon keputusan, sehingga sulit untuk memahami secara detail bagaimana setiap keputusan diambil. Teknik *boosting* dapat mengatasi rendahnya tingkat *interpretable* dan *overfitting* yang dihasilkan dari *Random Forest* [6]. Dengan demikian, penelitian ini bertujuan untuk mengatasi masalah tersebut dengan mengoptimalkan algoritma *Random Forest* menggunakan teknik *boosting* untuk meningkatkan akurasi prediksi *churn* pelanggan dan membantu perusahaan telekomunikasi dalam mengidentifikasi dan mengelola *churn* dengan akurat pada era jaringan 5G yang berkembang.

B. *Pendekatan Penelitian*

Berdasarkan temuan permasalahan di atas penelitian ini melakukan proses pendekatan penelitian yang berhubungan dengan permasalahan dan sumber-sumber terkait, serta beberapa tinjauan pustaka yang nantinya digunakan sebagai acuan untuk memulai penelitian. Pencarian terkait literatur dilakukan oleh peneliti menggunakan berbagai sumber, seperti: jurnal penelitian terdahulu yang relevan (5 tahun terakhir), buku, dan informasi pada internet.

C. *Analisis Sistem*

1. Kebutuhan untuk spesifikasi komputer minimal:
   - CPU AMD Dual Core
   - RAM 8 GB
   - Storage 1 GB
   - OS (Windows/Linux/MacOS)
2. Aplikasi dan modul atau library yang digunakan:
   - Google Colab
   - Visual Studio Code
   - Flask
   - HTML
   - CSS
   - Bootstrap 5
   - Tableau
   - Python
   - Numpy
   - Pandas
   - Seaborn
   - Matplotlib
   - Scikit-Learn
   - Imbalanced-Learn

D. *Pengumpulan Data*

Penelitian ini menggunakan jenis data sekunder yang diambil dari sumber platform online yaitu Kaggle dengan kata kunci "*Internet Service Provider Customer Churn*" pada URL (https://www.kaggle.com/code/edwingeevarughese/internet-service-churn-analysis) yang diunduh dengan format .csv. Dataset awal terdiri dari 72.274 data historis pelanggan dan 11 variabel kemudian setelah melakukan *pre-processing* data menjadi 45.471 data historis serta 10 variabel/fitur. Data tersebut berupa *customer churn* pada salah satu perusahaan telekomunikasi.

E. *Data Exploration*

Pada tahap ini, mengeksplorasi karakteristisk dan pola agar mengetahui informasi dan wawasan mengenai dataset yang digunakan. Seperti melakukan analisis deskriptif menggunakan "dataset.describe()" dalam statistik deskriptif, seperti mean, median, standar deviasi, nilai minimum dan maksimum untuk memahami distribusi dan karakteristik umum dari variabel-variabel terkait data pelanggan.

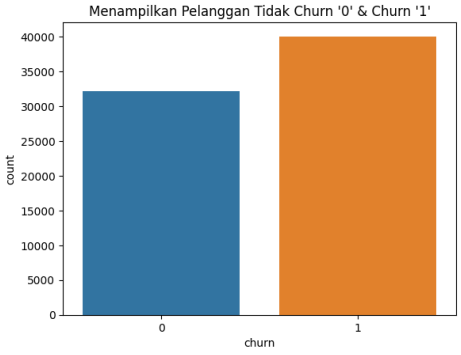
	id	is_tv_subscriber	is_movie_package_subscriber	subscription_age	bill_avg	reamining_contract	service_failure_count	download_avg	upload_avg
count	7.227400e+04	72274.000000	72274.000000	72274.000000	72274.000000	50702.000000	72274.000000	71893.000000	71893.000000
mean	8.463182e+05	0.815259	0.334629	2.450052	18.942483	0.716039	0.274234	43.689911	4.192076
std	4.891022e+05	0.388090	0.471864	2.034989	13.215386	0.697102	0.816621	63.405963	9.818896
min	1.500000e+01	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000
25%	4.222165e+05	1.000000	0.000000	0.930000	13.000000	0.000000	0.000000	6.700000	0.500000
50%	8.477840e+05	1.000000	0.000000	1.980000	19.000000	0.570000	0.000000	27.800000	2.100000
75%	1.269562e+06	1.000000	1.000000	3.300000	22.000000	1.310000	0.000000	60.500000	4.800000
max	1.689744e+06	1.000000	1.000000	12.800000	406.000000	2.920000	19.000000	4415.200000	453.300000

Gambar . 3 Analisis Deskriptif

Kemudian "dataset.info()" untuk menampilkan informasi mengenai dataset seperti jumlah baris yang tidak memiliki nilai *null* atau *NaN* (*Not a Number*) dan tipe data setiap variabel. Berikut *output* :

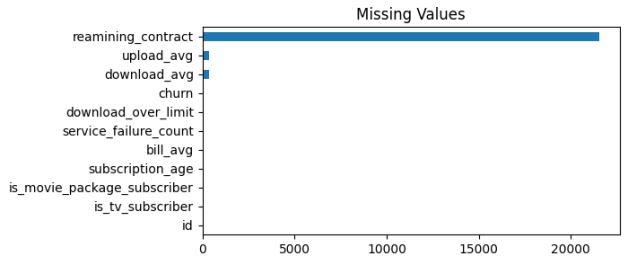
```
dataset.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 72274 entries, 0 to 72273
Data columns (total 11 columns):
 #   Column                       Non-Null Count  Dtype
---  ------                       --------------  -----
 0   id                           72274 non-null  int64
 1   is_tv_subscriber             72274 non-null  int64
 2   is_movie_package_subscriber  72274 non-null  int64
 3   subscription_age             72274 non-null  float64
 4   bill_avg                     72274 non-null  int64
 5   reamining_contract           50702 non-null  float64
 6   service_failure_count        72274 non-null  int64
 7   download_avg                 71893 non-null  float64
 8   upload_avg                   71893 non-null  float64
 9   download_over_limit          72274 non-null  int64
 10  churn                        72274 non-null  int64
dtypes: float64(4), int64(7)
memory usage: 6.1 MB
```

Selanjutnya, visualisasi variabel *churn* menggunakan library matplotlib dan seaborn untuk mengetahui jumlah data pelanggan yang melakukan *churn* (1) sekitar 40.000 pelanggan dan yang tidak *churn* (0) sekitar 32.000 pelanggan.



Gambar . 4 Visualisasi Data Pelanggan *Churn*

Lalu, visualisasi jumlah *missing value* menggunakan "library matplotlib" untuk mengetahui masing-masing variabel yang memiliki nilai *null* seperti variabel *reamining_contract* sekitar lebih dari 20.000 data, *upload_avg* dan *download_avg* sekitar 381 data.



Gambar . 5 Visualisasi Data *Missing Value*

Langkah terakhir memeriksa data duplikat pada dataset menggunakan "dataset.duplicated().sum()"sebagai berikut:

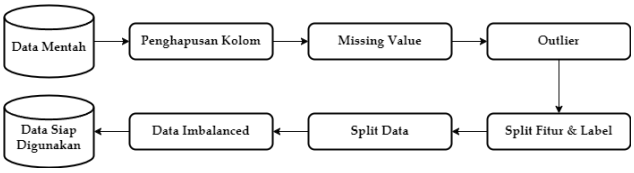
```
[ ]  #Memeriksa jumlah data duplikat
     dataset.duplicated().sum()

     0
```

Gambar . 6 *Output* Duplikat Data

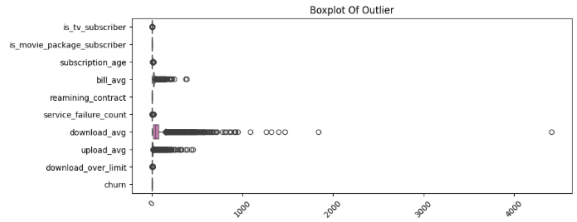
F. *Pre-processing Data*

merupakan tahap untuk membersihkan, mengatur, dan mengubah data menjadi bentuk yang lebih sesuai dan siap digunakan untuk tahap *modelling* agar menghasilkan akurasi model yang lebih baik.



Gambar . 7 Alur *Pre-Processing* Data

1. Penghapusan Kolom : Menghapus kolom atau variabel yang tidak diperlukan seperti kolom 'Id' karena kolom tersebut tidak menunjukkan kontribusi secara signifikan dalam model prediksi *churn* pelanggan. Dengan menggunakan kode program "dataset.drop(columns=['id'],inplace=True)".
2. *Missing Values* : Menghapus baris yang memiliki nilai *NaN* pada dataset, seperti pada kolom '*reamining_contract*', '*download_avg*', dan '*upload _avg*'. Jumlah nilai yang hilang relatif kecil dan baris tersebut dapat dihilangkan tanpa mengurangi signifikansi dataset. Hal ini perlu dilakukan karena dapat mempengaruhi kualitas analisis data dan performa model yang dibangun. Dengan menggunakan kode program "dataset.dropna(inplace=True)".
3. *Outlier* : Menangani data *outlier* (pencilan data) dilakukan menggunakan IQR (*interquartile range*) yaitu rentang antara kuartil pertama (Q1) dan kuartil ketiga (Q3) atau IQR = Q3 - Q1 dalam sebuah distribusi data. *Outlier* adalah nilai yang lebih rendah dari $Q1-1.5\times IQR$ atau nilai yang lebih tinggi dari $Q3+1.5\times IQR$. *Outlier* perlu ditangani karena dapat menyebabkan kesalahan dalam klasifikasi model, menghasilkan *bias* dalam estimasi parameter, dan mengakibatkan hasil yang tidak akurat. Keberadaan *outlier* dapat memengaruhi semua aspek ini. Dengan menggunakan kode program *outlier* dapat dilihat menggunakan visualisasi "library matpotlib dan seaborn" pada tiap variabel.



Gambar . 8 Visualisasi *Outlier* Pada Dataset

Dapat dilihat pada gambar di atas *outlier* yang terbanyak terdapat di variabel *download_avg* dan *upload_avg* sehingga menangani data *outlier* dengan IQR dilakukan pada dua variabel ini seperti kode program berikut:

```
cols_to_check = ['download_avg', 'upload_avg']
for col in cols_to_check:
  Q1 = dataset[col].quantile(0.25)
  Q3 = dataset[col].quantile(0.75)
  IQR = Q3 - Q1
  low_limit = Q1 - 1.5 * IQR
  high_limit = Q3 + 1.5 * IQR
dataset = dataset[(dataset[col] >= low_limit) &
(dataset[col] <= high_limit)]
--Output--
(45471, 10)
```

4. Split Fitur dan Label : Split fitur dan label adalah proses memisahkan data menjadi dua bagian: fitur dan label. Fitur adalah variabel yang digunakan sebagai input untuk memprediksi atau mengklasifikasikan label. Dalam konteks *machine learning*, fitur sering kali merupakan kolom-kolom dalam dataset yang digunakan untuk melatih model. Label adalah variabel yang ingin diprediksi atau diklasifikasikan oleh model. Label juga dikenal sebagai target atau *output*. Dalam tugas klasifikasi, label adalah kategori atau kelas yang ingin diprediksi. Dengan kode program "dataset.drop( )".

```
[ ]  #Pisahkan fitur (X) atau fitur input dan (y) sebagai label:
     X = dataset.drop(['churn'], axis=1)
     y = dataset['churn']
```

Gambar . 9 Split Fitur dan Label

5. Split Data : Split data rasio 80:20 merujuk pada praktik membagi dataset menjadi dua subset, di mana 80% dari data digunakan sebagai data pelatihan (training data) dan 20% sisanya digunakan sebagai data pengujian (testing data). Dengan membagi data dalam rasio ini, dapat memastikan model memiliki kinerja yang baik dalam memprediksi data baru yang belum pernah dilihat sebelumnya. Dengan mengimport "train_test_split" dari library "sklearn.model _selection".

```
#Split The Data
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(X, y, test_size = 0.2, random_state=42)

print('Train set size : ', X_train.shape, y_train.shape)
print('Test set size  : ', X_test.shape, y_test.shape)
```

```
Train set size :  (36376, 9) (36376,)
Test set size  :  (9095, 9) (9095,)
```

Gambar . 10 Split Data

6. Data *Imbalance* : Menggunakan teknik SMOTE *oversampling* yang bertujuan meningkatkan jumlah sampel pada kelas minoritas secara sintetis sehingga sama dengan kelas mayoritas menghasilkan kelas data menjadi seimbang. Dengan mengimport "SMOTE" dari library "imblearn.over_sampling".

```
from imblearn.over_sampling import SMOTE

oversampler = SMOTE(random_state=0)
x_resample, y_resample = oversampler.fit_resample(X_train,y_train)
print(x_resample.shape)
print(y_resample.shape)
```

```
(42204, 9)
(42204,)
```

Gambar . 11 Data *Imbalance*

G. *Modelling*

Modelling adalah proses di mana model atau algoritma yang dipilih kemudian dikembangkan dan dievaluasi berdasarkan data yang telah disiapkan untuk menghasilkan hasil prediksi akurat.

1. Membangun Model *Random Forest* tanpa optimasi dengan mengatur parameter seperti "*n_estimators* dan *random_state*" seperti berikut:

```
##Random Forest##
#Import model, akurasi, klassifikasi, dan confussion matrix
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.ensemble import RandomForestClassifier

# Inisialisasi model dan Setting Parameter Random Forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

# Melatih model Random Forest dengan data pelatihan
rf_model.fit(x_resample, y_resample)

# Memprediksi label pada data pengujian
y_pred_rf = rf_model.predict(X_test)

# Evaluasi Model Random Forest
print("Random Forest:")
print("Accuracy:", accuracy_score(y_test, y_pred_rf))
print("Classification Report:\n", classification_report(y_test, y_pred_rf))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_rf))
```

Gambar . 12 Model Random Forest

2. Membangun Model *Adaboost + Random Forest* dengan mengatur parameter "*n_estimators*, *random_state*, dan *estimators*" seperti berikut:

```
##Random Forest With AdaBoost##
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier

# Inisialisasi model dan Setting Parameter Random Forest
rf_model = RandomForestClassifier(n_estimators=100, random_state=42)

# Buat model dan setting parameter Adaboost dengan base estimator RF
adaboost_rf_model = AdaBoostClassifier(estimator=rf_model, n_estimators=100, random_state=42)

# Latih model Adaboost dengan data pelatihan
adaboost_rf_model.fit(x_resample, y_resample)

# Memprediksi label pada data pengujian
y_pred_adaboost = adaboost_rf_model.predict(X_test)

# Evaluasi model Adaboost+Random Forest
print("AdaBoost+Random Forest:")
print("Accuracy:", accuracy_score(y_test, y_pred_adaboost))
print("Classification Report:\n", classification_report(y_test, y_pred_adaboost))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_adaboost))
```

Gambar . 13 Model *Adaboost + Random Forest*

3. Membangun Model *XGBoost + Random Forest* dengan mengatur parameter "*n_estimators*, *random _state*, *estimators*, *max_depth*, *min_child_weight*, dan *gamma*" seperti berikut:

```
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from sklearn.ensemble import VotingClassifier

rf_model = RandomForestClassifier(n_estimators=100, random_state=42)
xgb_model = XGBClassifier(n_estimators=100, max_depth=4, min_child_weight=1, gamma=0.2, random_state=42)

# Menggabungkan kedua model menggunakan VotingClassifier
voting_model = VotingClassifier(estimators=[('rf', rf_model), ('xgb', xgb_model)], voting='hard')

# Latih model
voting_model.fit(x_resample, y_resample)

# Memprediksi label pada data pengujian
y_pred_xgboost = voting_model.predict(X_test)

# Evaluasi model XGBoost+Random Forest
print("XGBoost+Random Forest:")
print("Accuracy:", accuracy_score(y_test, y_pred_xgboost))
print("Classification Report:\n", classification_report(y_test, y_pred_xgboost))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_xgboost))
```

Gambar . 14 *XGBoost + Random Forest*

H. *Evaluasi*

Dalam mengukur keberhasilan optimasi yang dilakukan, tujuan evaluasi dalam konteks ini adalah untuk menentukan sejauh mana penggunaan teknik *Boosting* telah meningkatkan kinerja algoritma *Random Forest* dalam melakukan prediksi. Dengan menggunakan *confusion matrix* dapat membandingkan akurasi, presisi, recall, dan F1-score pada setiap model yang telah dibangun.

1. *Output* Evaluasi Model *Random Forest*

```
Random Forest:
Accuracy: 0.9503023639362287
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.99      0.96      5221
           1       0.98      0.90      0.94      3874

    accuracy                           0.95      9095
   macro avg       0.96      0.94      0.95      9095
weighted avg       0.95      0.95      0.95      9095

Confusion Matrix:
 [[5150   71]
 [ 381 3493]]
```

Gambar . 15 *Output* Evaluasi *Random Forest*

Hasil evaluasi model *Random Forest* tanpa optimasi menggunakan *confussion matrix* menghasilkan akurasi sebesar 95.03% ini menunjukkan bahwa *Random Forest* mampu memprediksi *churn* dengan baik.

2. *Output* Evaluasi Model *Adaboost+Random Forest*

```
AdaBoost+Random Forest:
Accuracy: 0.9520615722924683
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.99      0.96      5221
           1       0.98      0.90      0.94      3874

    accuracy                           0.95      9095
   macro avg       0.96      0.95      0.95      9095
weighted avg       0.95      0.95      0.95      9095

Confusion Matrix:
 [[5162   59]
 [ 377 3497]]
```

Gambar . 16 *Output* Evaluasi *Adaboost+Random Forest*

Hasil evaluasi *Random Forest* yang dioptimasi dengan *Adaboost* menggunakan *confussion matrix* menghasilkan akurasi sebesar 95.20% ini menunjukkan bahwa *Adaboost+Random Forest* mengalami peningkatan akurasi sebesar 0.17% sehingga mampu memprediksi *churn* dengan lebih baik dari model sebelumnya.

3. *Output* Evaluasi Model *XGBoost+Random Forest*

```
XGBoost+Random Forest:
Accuracy: 0.9500824628916987
Classification Report:
              precision    recall  f1-score   support

           0       0.93      0.99      0.96      5221
           1       0.99      0.89      0.94      3874

    accuracy                           0.95      9095
   macro avg       0.96      0.94      0.95      9095
weighted avg       0.95      0.95      0.95      9095

Confusion Matrix:
 [[5179   42]
 [ 412 3462]]
```

Gambar . 17 *Output* Evaluasi *XGBoost+Random Forest*

Hasil evaluasi *Random Forest* yang dioptimasi dengan XGBoost menggunakan *confussion matrix* menghasilkan akurasi sebesar 95% ini menunjukkan bahwa *XGBoost+Random Forest* mengalami penurunan akurasi sebesar 0.03% hanya selisih sedikit dengan *Random Forest* tanpa optimasi sehingga masih lebih baik *Random Forest* saja dalam memprediksi *churn* pelanggan.

4. Perbandingan Hasil Evaluasi

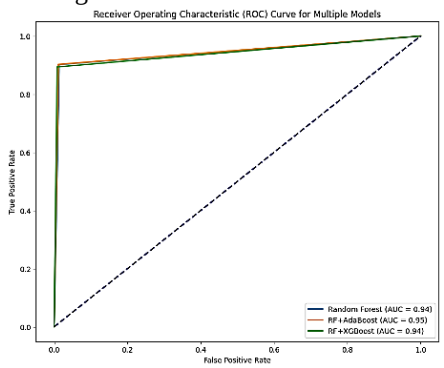
TABEL I
Hasil Persentase Tiap Model

Model	akurasi	presisi	recall	f1-score
Random Forest	95.03%	98%	90%	94%
Random Forest +Adaboost	95.20%	98%	90%	94%
Random Forest +XGBoost	95%	99%	89%	94%

Dapat dilihat dari tabel di atas terdapat penurunan recall dari 90% ke 89% pada *XGBoost+Random Forest* dan peningkatan presisi dari 98% ke 99%. Hal ini dapat disebabkan karena perbedaan tiap model dalam menangani klasifikasi prediksi *churn* dan tidak *churn*. Kemudian, hasil terbaik yaitu model *Random Forest* yang dioptimasi dengan teknik *boosting Adaboost* menghasilkan akurasi sebesar 95.20% ini mengalami peningkatan sebesar 0.17%. Dengan demikian, model terbaik ini kemudian diterapkan sebagai model untuk prediksi *churn* pelanggan.

5. Evaluasi Menggunakan AUC-ROC

Selain dengan *confussion matrix* evaluasi model juga menggunakan AUC-ROC menghasilkan output sebagai berikut:



Gambar . 18 Visualisasi AUC-ROC

I. *Deployment*

Deployment adalah tahap implementasi model yang telah dikembangkan ke dalam aplikasi berbasis web sehingga dapat digunakan secara langsung oleh pengguna. Dalam kasus prediksi *churn* pelanggan di industri telekomunikasi, tujuan *deployment* adalah agar perusahaan telekomunikasi dapat menggunakan model AI melalui aplikasi berbasis web untuk memprediksi pelanggan yang berpotensi *churn*. Dengan mengetahui pelanggan yang berpotensi *churn*, perusahaan telekomunikasi dapat melakukan tindakan yang tepat untuk mencegah pelanggan berpindah ke layanan lainnya. Peneliti mengembangkan aplikasi berbasis web menggunakan *framework* dari python yaitu flask untuk menampilkan model yang telah dikembangkan dalam memprediksi *churn* yang dapat diakses melalui *localhost* atau menggunakan server lokal.

IV. HASIL DAN PEMBAHASAN

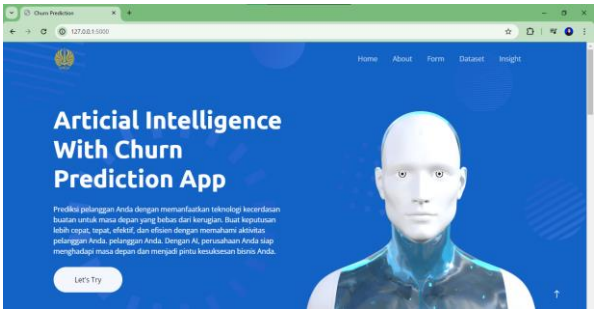
Pada tahap ini, peneliti mengembangkan aplikasi berbasis web menggunakan flask yang dapat diakses melalui *localhost* atau menggunakan server lokal. Sebelum model dideploy ke dalam aplikasi, model di import terlebih dahulu ke dalam bentuk .joblib. Setelah mendapatkan model.joblib kemudian dimasukkan ke dalam kodingan pengembangan aplikasi berbasis website, tools yang digunakan yaitu visual studio code. Lalu load model dengan kode program "model = joblib.load ("model/adaboost_rf_model.joblib")".

Kemudian melakukan aktivasi virtual environment yang dibuat menggunakan Python's venv atau virtualenv di lingkungan Windows. Mengaktifkan virtual environment dengan menjalankan kode program "myenv/Scripts/activate" di terminal pada visual studio code, itu akan mengaktifkan virtual environment tersebut dengan mengatur ulang beberapa variabel lingkungan (seperti PATH dan PYTHONPATH) untuk mengarah ke direktori myenv sehingga versi Python dan paket yang digunakan di lingkungan tersebut menjadi aktif dan tersedia secara lokal. Kemudian aplikasi berbasis website dapat diakses melalui localhost dengan kode program sebagai berikut:

```
PS C:\Users\ASUS\Documents\SKRIPSI\Deployment>
myenv/Scripts/activate
(myenv)PS
C:\Users\ASUS\Documents\SKRIPSI\Deployment>
python app.py
 * Tip: There are .env or .flaskenv files present.
 * Serving Flask app 'app'
 * Debug mode: on
 * Running on http://127.0.0.1:5000
```

1. Implementasi Sistem

Setelah menjalankan link diatas pada *browser* akan otomatis menampilkan halaman *website* yang pertama yaitu fitur "*Home*". Pada tampilan *Home* menampilkan ucapan singkat selamat datang kepada *user*, *navbar* yang menampilkan semua fitur memudahkan *user* menjelajahi setiap fitur dan tombol "Let's Try" yang mengarahkan ke halaman *our fitur* yang berada di bawah *Home*.

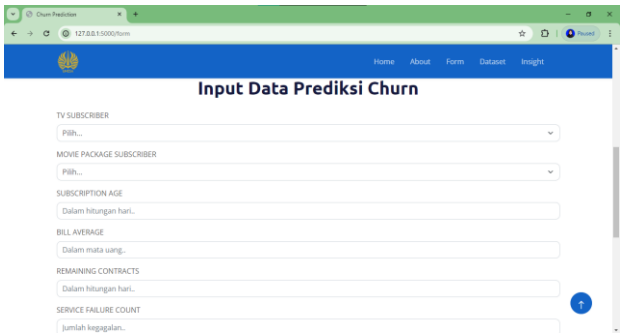


Gambar . 19 Tampilan Fitur *Home*

Kemudian terdapat fitur "*About*" yang menampilkan informasi singkat tentang aplikasi dan akurasi model yang digunakan.

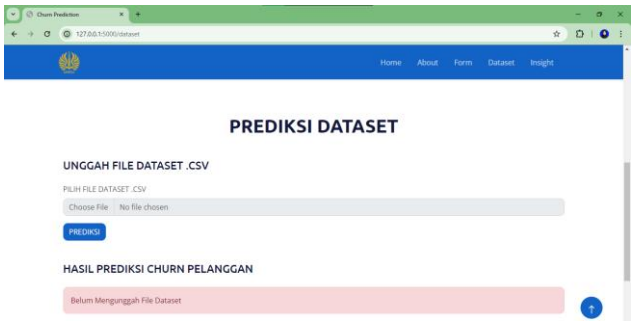
Gambar . 20 Tampilan Fitur *About*

Selanjutnya terdapat fitur "*Form*" yang menampilkan 10 *form* atau kolom untuk *user* mengisi agar dapat memprediksi pelanggan berdasarkan variabel yang telah disediakan.



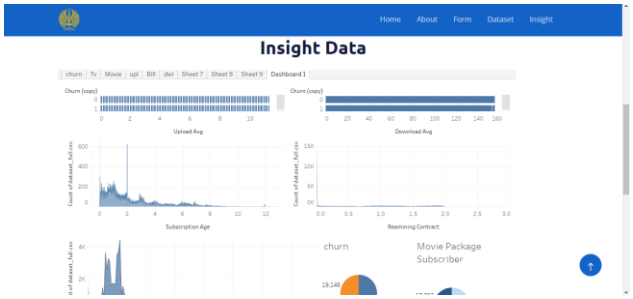
Gambar . 21 Tampilan Fitur *Form*

Lalu terdapat fitur "*Dataset*" yang menampilkan kolom untuk *user* mengunggah file dataset dengan format .csv dengan *output* persentase *churn* dan tidak *churn* sehingga dapat memprediksi *churn* pelanggan berdasarkan keseluruhan dataset.



Gambar . 22 Tampilan Fitur *Dataset*

Dan yang terakhir terdapat tampilan fitur "*Insight*" yaitu berupa *dashboard* Tableau untuk prediksi *churn* pelanggan di industri telekomunikasi memberikan visualisasi interaktif mengenai tren *churn*, segmentasi pelanggan, faktor-faktor pengaruh, dan prediksi *churn*. Memungkinkan *user* mudah menganalisis dan pengambilan keputusan berdasarkan data dengan cepat.



Gambar . 23 Tampilan Fitur Insight

V. KESIMPULAN DAN SARAN

Berdasarkan penelitian yang telah dilakukan, kesimpulan yang dapat diambil adalah sebagai berikut:

1. Penerapan teknik *boosting* pada algoritma *Random Forest* yang dilakukan yaitu menggunakan *Adaboost* dan *XGBoost* dengan mengatur parameter menunjukkan bahwa penggunaan teknik *boosting* dengan *Adaboost* dapat meningkatkan akurasi algoritma *Random Forest* sebesar 0.17%. Sedangkan teknik *boosting XGBoost* mengalami penurunan akurasi sebesar 0.03%. Dengan demikian, model *Random Forest+Adaboost* yang menghasilkan akurasi lebih tinggi digunakan untuk memprediksi *churn* pelanggan di Industri Telekomunikasi.
2. Keberhasilan optimasi algoritma *Random Forest* dengan teknik *boosting* dapat diukur melalui peningkatan akurasi model prediksi menggunakan *confusion matrix* dan AUC-ROC. Dengan *confusion matrix*, akurasi model *Random Forest* tanpa optimasi sebesar 95.03%, model *Random Forest* dengan optimasi *XGBoost* mengalami penurunan akurasi 0.03% menjadi 95.00%. Hal ini dapat terjadi disebabkan *overfitting* akibat kompleksitas model yang berlebihan. Sedangkan model *Random Forest* yang dioptimasi *Adaboost* mengalami peningkatan 0.17% menjadi 95.20%. Hal ini disebabkan *Adaboost* lebih efektif dalam menangani kelemahan model *Random Forest*, seperti menekan *overfitting*, meningkatkan generalisasi, dan meningkatkan kinerja model secara keseluruhan. Selain itu, penelitian ini juga menghasilkan akurasi AUC-ROC algoritma *Random Forest* sebesar 94%, *Random Forest+ XGBoost* 94%, dan *Random Forest+Adaboost* 95% sehingga mengalami peningkatan 1% membuktikan bahwa teknik *boosting* dengan *Adaboost* berhasil meningkatkan akurasi model *Random Forest* agar dapat mengidentifikasi *churn* pelanggan dengan lebih baik dan akurat.

3. Pembangunan aplikasi berbasis website menggunakan *framework* flask dan menerapkan model *Random Forest* yang telah dioptimasi pada aplikasi dengan format "model.joblib", serta integrasi dengan beberapa library dan modul untuk pemrosesan data dan model. Dengan fitur yang telah dibuat seperti fitur *Form*, *Dataset*, dan *Insight* pengguna dapat memprediksi *churn* pelanggan di industri telekomunikasi lebih cepat, akurat, dan efisien.

Adapun saran yang dapat peneliti berikan untuk pengembangan penelitian di masa mendatang adalah sebagai berikut:

1. Melanjutkan penelitian dengan variasi teknik optimasi yang berbeda dalam mengevaluasi kinerja setiap model dan efektivitas masing-masing untuk mengetahui model mana yang lebih baik dalam meningkatkan model prediksi *churn* pelanggan.
2. Penelitian selanjutnya dapat menggunakan teknik evaluasi yang berbeda seperti *cross validation*, *log loss*, dll.
3. Penelitian selanjutnya diharapkan dapat menggunakan data primer yaitu data pelanggan industri telekomunikasi yang berada di negara Indonesia sehingga dapat membantu perusahaan telekomunikasi yang ada di Indonesia.

REFERENSI

[1] Putra LD, Sutabri T. Analisis Teknologi Jaringan 5G pada Komunikasi Generasi Pendatang Dengan Menggunakan Metode Inquiry. IJM: Indonesian Journal of Multidisciplinary. 2024 Apr 7;2(2).

[2] Wicaksono A, Padilah TN. Pengaruh Jumlah Record Dataset Terhadap Algoritma Klasifikasi Berdasarkan Data Customer Churn. Jurnal Ilmiah Informatika. 2021 Jun 29;6(1):1-0.

[3] Dachi JM, Sitompul P. Analisis Perbandingan Algoritma XGBoost dan Algoritma Random Forest Ensemble Learning pada Klasifikasi Keputusan Kredit. Jurnal Riset Rumpun Matematika dan Ilmu Pengetahuan Alam (JURRIMIPA). 2023 Jul 5;2(2):87-103.

[4] Abdulsalam SO, Ajao JF, Balogun BF, Arowolo MO. A churn prediction system for telecommunication company using random forest and convolution neural network algorithms. EAI Endorsed Transactions on Mobile Communications and Applications. 2022 Jul 27;7(21).

[5] Wirya MA. *Deteksi Penyakit Alzheimer Pada Citra Magnetic Resonance Imaging Menggunakan Machine Learning dengan Metode Convolutional Neural Network* (Bachelor's thesis, Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta).

[6] Dinata NA, Abdurrahman G, Fitriyah NQ. Perbandingan Optimasi Algoritma Random Forest Menggunakan Teknik Boosting Terhadap Kasus Klasifikasi Churn Pelanggan Di Industri Telekomunikasi. Jurnal Aplikasi Sistem Informasi dan Elektronika. 2023 Sep 7;5(1):28-37.

[7] Christia A, Hadi AS, Febriana A, Budihardjo A, Wiradarmo AA, Elfriede DP, Ardianto E, da Silva EN, Sari F, Kusumadewi FN, Widjojo H. Kecerdasan Buatan: Arah dan Eksplorasinya. Prasetiya Mulya Publishing; 2024 Jan 31.

[8] Primandari, Arum Handini. Implementasi Metode Random Forest Dan Xgboost Pada Klasifikasi Customer Churn. 2020.

[9] Aisyah N. Implementasi Algoritma K-Nearest Neighbor untuk Sistem Rekomendasi Topik Penelitian pada Jurusan Ilmu Komputer Universitas Lampung Berbasis Klasifikasi.