PROGRAM KREATIVITAS MAHASISWA

PREDIKSI TREATMENT MENTAL HEALTH

LAPORAN AKHIR PROJEK MACHINE LEARNING

Diusulkan oleh :

Ivan Bryliant	2702363922	2024
Angel Rich Faustine Winata	2702363701	2024
Paundra Rangga Zulkarnaen	2702364080	2024

UNIVERSITAS BINA NUSANTARA

JAKARTA

2025

PREDIKSI TREATMENT MENTAL HEALTH

Ivan Bryliant, Angel Rich Faustine Winata, Paundra Rangga

Computer Science, School of Computer Science, Universitas Bina Nusantara

Jl. Kebon Jeruk Raya No. 27 Kebon Jeruk, Jakarta Barat 11530, Indonesia

ABSTRAK

Kesehatan mental di tempat kerja menjadi isu yang semakin mendesak, terutama di industri teknologi yang memiliki tingkat stres yang tinggi. Untuk mengatasi masalah ini, penelitian ini bertujuan untuk membangun sebuah model prediktif untuk memprediksi kebutuhan *treatment* kesehatan mental karyawan berdasarkan dataset dari Kaggle. Model yang digunakan dalam penelitian ini mencakup beberapa algoritma machine learning, seperti *Logistic Regression, Random Forest, XGBoost, Bagging, KNN, Decision Tree*, dan *Stacking.* Hasil evaluasi model menunjukkan bahwa metode *Stacking* dan *Boosting* memberikan performa terbaik dalam hal classification accuracy (82.011%) dan F1-Score (0.837). Analisis lebih lanjut menggunakan *Confusion Matrix* dan *ROC Curve* mengonfirmasi bahwa model *Boosting* memiliki kemampuan yang sangat baik dalam mengidentifikasi kasus positif dengan tingkat *false positive* yang terkontrol. Dengan model ini, perusahaan dapat mengambil langkah proaktif untuk mendukung kesehatan mental karyawan dan mengurangi stigma terkait perawatan kesehatan mental, sekaligus meningkatkan kesejahteraan dan produktivitas di lingkungan kerja.

Kata Kunci: Kesehatan Mental, Prediksi *Mental Health*, *Machine Learning, Logistic Regression, Random Forest, XGBoost, Bagging, KNN, Decision Tree, Stacking, Classification Accuracy, F1-Score, Confusion Matrix, ROC Curve, False Positive*, Stigma.

ABSTRACT

Mental health in the workplace has become an increasingly urgent issue, particularly in the technology industry, which has high levels of stress. To address this issue, this research aims to build a predictive model to predict the mental health treatment needs of employees based on a Kaggle dataset. The models used in this research include several machine learning algorithms such as Logistic Regression, Random Forest, XGBoost, Bagging, KNN, Decision Tree, and Stacking. The model evaluation results show that the Stacking and Boosting methods provide the best performance in terms of classification accuracy (82.011%) and F1-Score (0.837). Further analysis using Confusion Matrix and ROC Curve confirms that the Boosting model performs excellently in identifying positive cases with a controlled false positive rate. With this model, companies can take proactive steps to support employee mental health, reduce stigma related to mental health treatment, and improve overall well-being and productivity in the workplace.

Keyword*: *Mental health, Mental Health Prediction, Machine Learning, Logistic Regression, Random Forest, XGBoost, Bagging, KNN, Decision Tree, Stacking, Classification Accuracy, F1-Score, Confusion Matrix, ROC Curve, False Positive, Stigma.

PENDAHULUAN

Kesehatan mental adalah kondisi kesejahteraan di mana individu dapat mengatasi tekanan hidup, bekerja secara produktif, dan berkontribusi terhadap komunitasnya. Menurut Organisasi Kesehatan Dunia (WHO), kesehatan mental didefinisikan sebagai keadaan kesejahteraan di mana seseorang menyadari kemampuannya sendiri, dapat menghadapi tekanan normal dalam kehidupan, dapat bekerja secara produktif, dan mampu memberikan kontribusi kepada komunitasnya (WHO, 2021). Kesehatan mental bukan hanya sekedar ketiadaan gangguan mental, tetapi juga mencakup kemampuan individu untuk mengelola stres, membangun hubungan yang sehat, dan mengambil keputusan yang bijaksana.

Dalam beberapa tahun terakhir, kesehatan mental telah menjadi isu global yang semakin mendesak. Menurut WHO, satu dari empat orang di dunia akan mengalami masalah kesehatan mental pada suatu titik dalam hidup mereka (WHO, 2021). Di lingkungan kerja, khususnya di industri teknologi, masalah kesehatan mental semakin menonjol karena tingginya tingkat stres, jam kerja yang panjang, dan budaya kerja yang kompetitif (Bhui et al., 2016). Survei yang dilakukan oleh Open Sourcing Mental Illness (OSMI) pada tahun 2014 menunjukkan bahwa 51% karyawan di industri teknologi pernah mengalami masalah kesehatan mental, seperti kecemasan dan depresi, namun hanya 20% yang merasa nyaman untuk membicarakannya di tempat kerja (OSMI, 2014). Hal ini menunjukkan adanya stigma yang kuat terkait kesehatan mental, yang menghalangi karyawan untuk mencari bantuan yang mereka butuhkan.

Permasalahan ini tidak hanya berdampak pada individu, tetapi juga pada produktivitas perusahaan. Menurut penelitian oleh Deloitte (2020), masalah kesehatan mental di tempat kerja dapat menyebabkan penurunan produktivitas, peningkatan absensi, dan biaya kesehatan yang lebih tinggi. Oleh karena itu, penting bagi perusahaan untuk mengambil langkah proaktif dalam mengidentifikasi dan mendukung karyawan yang mungkin membutuhkan treatment mental health.

TUJUAN

Proyek ini bertujuan untuk membangun model prediktif berdasarkan data survei kesehatan mental dari Kaggle. Dengan memanfaatkan dataset ini, kami akan menganalisis berbagai faktor seperti usia, jenis kelamin, riwayat keluarga, dan kondisi kerja untuk mengidentifikasi karyawan yang berisiko mengalami masalah kesehatan mental. Model ini dirancang untuk memberikan prediksi yang akurat, sehingga perusahaan dapat mengambil langkah proaktif

dalam menyediakan intervensi yang tepat waktu, seperti konseling atau program kesehatan mental.

Dengan adanya model prediktif ini, diharapkan perusahaan dapat menciptakan lingkungan kerja yang lebih sehat dan mendukung, mengurangi stigma terkait kesehatan mental, serta meningkatkan kesejahteraan dan produktivitas karyawan secara keseluruhan.

METODOLOGI

Metode yang akan digunakan untuk membuat model machine learning untuk memprediksi kebutuhan treatment seseorang ialah: *Logistic Regression, Stacking, KNN, Random Forest Regression, Decission Tree Regression, XGBoost Regression, Bagging.* Dari keenam model ini akan dibandingkan mana yang memiliki kinerja terbaik menggunakan berbagai metrik diantaranya: *Accuracy, Confusion Matrix, AUC Score,* dan *F1 Score.*

A. Random Forest

Random Forest Regression adalah metode ensemble learning yang menggunakan banyak pohon keputusan *(decision trees)* untuk membuat prediksi. Setiap *tree* dalam *forest* dihasilkan dari subset acak data dan fitur, dan prediksi akhir didapatkan dengan menggabungkan prediksi dari semua pohon.

- Algoritma Random Forest:
  - *Bootstrap Aggregation (Bagging):* Setiap pohon dihasilkan dari sampel acak dari data pelatihan (dengan penggantian) kami melakukan implementasi bagging secara tersendiri pada model yang akan kami kembangkan.
  - *Random Feature Selection:* Saat membagi setiap node dalam pohon, hanya subset acak fitur yang dipertimbangkan, yang membantu mengurangi korelasi antar pohon.
  - *Averaging:* Prediksi untuk data baru diperoleh dengan merata-ratakan prediksi dari semua pohon dalam hutan (untuk regresi).

Formula Prediksi *Random Forest*:

$$Prediction = \frac{1}{T}\sum_{t=1}^{T} Prediction_t$$

Dimana :

- T adalah jumlah tree dalam forest

- Prediction-t adalah prediksi dari tree ke - t

Random Forest cenderung mengurangi risiko *overfitting* dibandingkan dengan *decision tree* dan dapat menangani data yang memiliki banyak fitur dan hubungan non-linier. Namun, model ini bisa menjadi kurang *interpretable* karena kompleksitas dan banyaknya *tree* yang terlibat.

B. *Decission Tree*

Decision Tree Regression adalah model yang menggunakan struktur pohon keputusan untuk melakukan prediksi. *Decision Tree* ini akan membagi data menjadi subset berdasarkan fitur dan nilai tertentu, sehingga membuat prediksi berdasarkan rata-rata nilai target dalam setiap *leaf node* (daun) pohon.

- Algoritma *Decision Tree*:
  - *Splitting*: Membagi data pada setiap node berdasarkan fitur yang meminimalkan impurity (seperti *Mean Squared Error* atau *Variance*).
  - *Stopping Criteria*: Proses splitting berhenti ketika data dalam node terlalu kecil, atau jumlah level pohon mencapai batas tertentu.
  - *Prediction*: Untuk data baru, pohon traversed dari *root* ke *leaf node* untuk memberikan prediksi berdasarkan rata-rata nilai di *node* tersebut.

Decision Tree dapat menangani data non-linier dan tidak memerlukan normalisasi data, tetapi cenderung *overfitting* jika tidak dipangkas dengan benar.

C. *XGBoost Regression*

XGBoost (Extreme Gradient Boosting) adalah algoritma *boosting* yang kuat dan populer untuk masalah regresi dan klasifikasi. *XGBoost* membangun model prediktif dengan menggabungkan banyak pohon keputusan (dalam hal ini, pohon keputusan sederhana) untuk meningkatkan akurasi dan mengurangi *overfitting*.

- Algoritma *XGBoost*:
  - *Boosting*: *XGBoost* membangun model secara bertahap, di mana setiap model baru mencoba untuk memperbaiki kesalahan dari model sebelumnya.
  - *Gradient Descent*: Model baru dihasilkan dengan meminimalkan fungsi loss menggunakan teknik gradient descent.

- *Regularization*: *XGBoost* menggunakan teknik regularisasi (L1 dan L2) untuk mengurangi kompleksitas model dan mencegah *overfitting.*

XGBoost seringkali sangat efektif dalam mengatasi masalah dengan data yang kompleks dan banyak fitur, serta memiliki parameter tuning yang memungkinkan penyesuaian yang lebih baik terhadap data spesifik.

D. *Logistic Regression*

Logistic Regression adalah metode statistik yang digunakan untuk klasifikasi biner, memprediksi probabilitas bahwa sebuah instance termasuk dalam kategori tertentu. Tidak seperti regresi linier yang memprediksi nilai kontinu, regresi logistik memprediksi nilai diskrit (biasanya 0 atau 1) dengan menggunakan fungsi sigmoid.

- Algoritma *Logistic Regression***:**
  - **Fungsi Sigmoid:** Mengubah output linier menjadi probabilitas antara 0 dan 1.

    Fungsi sigmoid didefinisikan sebagai:

    $\sigma(z) = 1 / (1 + e^{(-z)})$

    di mana z adalah kombinasi linier dari fitur-fitur input.
  - ***Maximum Likelihood Estimation (MLE):*** Digunakan untuk menemukan parameter model (koefisien) yang memaksimalkan kemungkinan mengamati data pelatihan.
  - ***Thresholding*****:** Probabilitas yang dihasilkan oleh fungsi sigmoid dibandingkan dengan ambang batas (biasanya 0,5) untuk menentukan klasifikasi akhir (0 atau 1).

E. *Stacking*

Stacking adalah metode ensemble learning yang menggabungkan prediksi dari beberapa model dasar (*base models*) untuk membuat prediksi akhir. Berbeda dengan bagging atau boosting yang menggunakan model sejenis, stacking dapat menggunakan model yang berbeda-beda. Prediksi dari model dasar digunakan sebagai fitur input untuk model meta-learner, yang kemudian membuat prediksi akhir.

- Algoritma *Stacking*:

- Model Dasar (*Base Models*): Beberapa model yang berbeda dilatih pada data pelatihan.
- Prediksi Tingkat Pertama (*First-Level Predictions*): Model dasar membuat prediksi pada data validasi (atau menggunakan cross-validation).
- Model Meta-Learner (*Meta-Learner Model*): Model baru dilatih menggunakan prediksi dari model dasar sebagai fitur input.
- Prediksi Akhir (*Final Prediction*): Model meta-learner membuat prediksi akhir.

- Formula Prediksi *Stacking*:

  Tidak ada formula tunggal karena tergantung pada model dasar dan meta-learner yang digunakan. Secara umum:
  - Prediksi Model Dasar: P_base = [prediksi model1, prediksi model2, ...]
  - Prediksi Akhir: P_final = MetaLearner(P_base)

F. K-Nearest Neighbors (KNN)

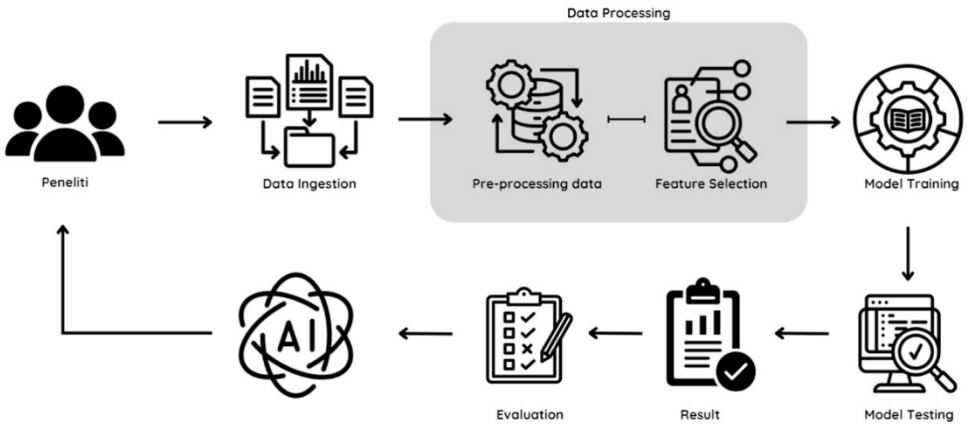
K-Nearest Neighbors (KNN) adalah algoritma supervised learning yang digunakan untuk klasifikasi dan regresi. KNN bekerja dengan menyimpan semua data pelatihan dan memprediksi label atau nilai untuk titik data baru berdasarkan label atau nilai dari k tetangga terdekatnya.

- Algoritma KNN:
  - Penyimpanan Data: Menyimpan semua data pelatihan.
  - Perhitungan Jarak: Menghitung jarak antara titik data baru dan semua titik data pelatihan (biasanya menggunakan Euclidean distance).
  - Pemilihan Tetangga: Memilih k tetangga terdekat berdasarkan jarak.
  - Prediksi:
    - Klasifikasi: Memilih label mayoritas dari k tetangga.
    - Regresi: Merata-ratakan nilai dari k tetangga.
- Formula Prediksi KNN:

  Tidak ada formula matematis eksplisit, tetapi prosesnya dapat dijelaskan sebagai:
  - Jarak(x, xi) = $\sqrt{\Sigma(x - xi)^2}$ (Euclidean distance)
  - Prediksi = Mode(label tetangga - klasifikasi) atau Rata-rata(nilai tetangga - regresi)

METODE PENELITIAN

I. Proses Pembuatan Model Machine Learning (*Workflow*)



1) *Data Ingestion*

   Dataset yang digunakan dalam penelitian ini diperoleh dari Kaggle yang berjudul "Mental Health in Tech Survey" (https://www.kaggle.com/datasets/osmi/mental-health-in-tech-survey/data).

   Dataset ini terdiri dari 1.259 data dan mencakup 27 atribut yaitu *country*, *state*, *self_employed*, *family_history*, *treatment*, *work_interfere*, *no_employees*, *remote_work*, *tech_company*, *benefits*, *care_options*, *wellness_program*, *seek_help*, *anonymity*, *leave*, *mental_health_consequence*, *phys_health_consequence*, *coworkers*, *supervisor*, *mental_health_interview*, *phys_health_interview*, *mental_vs_physical*, *obs_consequence*, dan *comments*. Data ini memberikan wawasan tentang bagaimana faktor individu dan lingkungan kerja mempengaruhi kesehatan mental karyawan di industri teknologi.

2) *Preprocessing*

   Pada langkah ini, data yang tidak akurat, tidak relevan, atau tidak lengkap diidentifikasi dan kemudian dikoreksi. Proses *data cleaning* yang dilakukan meliputi:

   - *Handling Missing Data* : Mengidentifikasi dan menangani nilai yang hilang atau kosong dalam dataset.
   - *Removing Duplicate Data* : Menghapus baris data yang identik untuk mencegah bias dalam analisis.
   - *Correcting Errors* : Mendeteksi dan memperbaiki kesalahan data, seperti nilai yang tidak mungkin atau tidak masuk akal.

Selain itu, dalam langkah *preprocessing* ini juga dilakukan normalisasi atau standarisasi terhadap data yang sudah ada. Proses ini bertujuan untuk mengubah data agar berada dalam skala yang seragam, sehingga algoritma pembelajaran mesin dapat memprosesnya lebih efisien dan menghasilkan model yang lebih baik. Normalisasi mengubah data ke rentang tertentu (misalnya, antara 0 dan 1), sementara standarisasi mengubah data sehingga memiliki rata-rata 0 dan deviasi standar 1.

Setelah data berhasil diolah dan dipersiapkan dengan benar melalui langkah-langkah di atas, data yang telah dibersihkan dan disiapkan kemudian dibagi menjadi *training data* dan *testing data*. Pembagian ini penting untuk melatih model pada data pelatihan dan kemudian menguji kinerjanya pada data yang belum pernah dilihat sebelumnya, yang memungkinkan untuk mengevaluasi generalisasi model tersebut terhadap data baru.

3) *Feature Selection*

Pada proses ini, dilakukan pemilihan fitur untuk menentukan apakah semua fitur yang tersedia perlu digunakan dalam melatih model *machine learning.* Fitur yang dianggap tidak relevan atau memiliki korelasi rendah dengan *'treatment'* akan dihapus untuk mengoptimalkan kinerja model.

4) *Model Training/Building*

Training model dalam machine learning adalah proses di mana model belajar dari data untuk mengidentifikasi pola dan hubungan guna membuat prediksi atau keputusan. Selanjutnya, dipilih beberapa model yang akan digunakan, termasuk *logistic regression*, KNN, *Decision Tree, Random Forest, Bagging, Boosting,* dan *Stacking*, lalu melatih model-model tersebut dengan training data yang telah dimiliki untuk menghasilkan model terbaik.

5) Model Testing

Model yang telah melalui tahap training akan diuji dengan menggunakan data testing untuk mengevaluasi performa model dan memastikan bahwa model tersebut dapat memprediksi apakah pasien membutuhkan treatment secara atau

tidak secara akurat. Di dalam data testing akan menggunakan berbagai metrik evaluasi seperti *Accuracy* (akurasi), *F1-Score, Precision*, dan *AUC Score*.

6) *Result*

Hasil dari pengujian model akan dirangkum dan disajikan dalam bentuk tabel dan visualisasi untuk memudahkan pemahaman mengenai performa masing-masing model dalam memprediksi harga rumah.

7) *Evaluation*

Model-model yang telah dilatih dan diuji akan dievaluasi menggunakan metrik evaluasi yang ada, yaitu *Accuracy* (akurasi), *F1-Score*, *Precision*, dan *AUC Score*. Setelah mengetahui hasil evaluasi berdasarkan metrik-metrik tersebut, akan dipilih model dengan performa terbaik (*F1-Score*) untuk digunakan dalam prediksi *mental health*

II. Pemrosesan Data (*Preprocessing Data*)

A. Penanganan *Null Value/ Missing Data*

Dari 1.259 data yang mencakup 27 atribut, kami melakukan pengecekan terhadap data yang hilang (*Missing Values*) untuk mengetahui sejauh mana atribut-atribut dalam dataset mengalami kekosongan. Hasil pengecekan jumlah dan persentase data yang hilang ditunjukkan dalam tabel berikut:

Atribut	Total Data Hilang	Presentase Hilang(%)
comments	1095	86.97
state	515	40.91
work_interfere	264	20.97
self_employed	18	1.43
Gender	0	0.00
Timestamp	0	0.00
Age	0	0.00
family_history	0	0.00
treatment	0	0.00
no_employees	0	0.00
Country	0	0.00

remote_work	0	0.00
tech_company	0	0.00
care_options	0	0.00
benefits	0	0.00
seek_help	0	0.00
anonymity	0	0.00
leave	0	0.00
wellness_program	0	0.00
mental_health_consequence	0	0.00
phys_health_consequence	0	0.00
supervisor	0	0.00
coworkers	0	0.00
mental_health_interview	0	0.00
phys_health_interview	0	0.00
mental_vs_physical	0	0.00
obs_consequence	0	0.00

Berdasarkan hasil analisis, atribut "*state*" dan "*comments*" memiliki jumlah *missing values* yang paling banyak, masing-masing sebesar 40.91% dan 86.97%. Karena nilai yang hilang pada atribut tersebut sangat tinggi, kami memutuskan untuk menghapus kedua atribut tersebut dari dataset agar tidak mempengaruhi kualitas model.

Selain itu, kami juga melakukan penghapusan fitur "*Timestamp*", karena atribut ini tidak memiliki relevansi langsung dengan target prediksi yang akan dilakukan.

```
train_df = train_df.drop(['comments'], axis= 1)
train_df = train_df.drop(['state'], axis= 1)
train_df = train_df.drop(['Timestamp'], axis= 1)
```

B. Penanganan Data pada Atribut Gender

Pada atribut "*Gender*", ditemukan berbagai entri yang tidak seragam dan terlalu acak, sehingga diperlukan standarisasi agar lebih mudah diolah dalam model machine learning. Oleh karena itu, data pada atribut *"Gender"* dikelompokkan menjadi tiga kategori utama, yaitu:

```
print(train_df['Gender'].unique())
```

```
[142] ✓ 0.0s
```

```
['Female' 'M' 'Male' 'male' 'female' 'm' 'Male-ish' 'maile' 'Trans-female'
 'Cis Female' 'F' 'something kinda male?' 'Cis Male' 'Woman' 'f' 'Mal'
 'Male (CIS)' 'queer/she/they' 'non-binary' 'Femake' 'woman' 'Make' 'Nah'
 'All' 'Enby' 'fluid' 'Genderqueer' 'Female ' 'Androgyne' 'Agender'
 'cis-female/femme' 'Guy (-ish) ^_^' 'male leaning androgynous' 'Male '
 'Man' 'Trans woman' 'msle' 'Neuter' 'Female (trans)' 'queer'
 'Female (cis)' 'Mail' 'cis male' 'A little about you' 'Malr' 'p' 'femail'
 'Cis Man' 'ostensibly male, unsure what that really means']
```

Male	"male", "m", "male-ish", "maile", "mal", "male (cis)", "make", "male ", "man","msle", "mail", "malr","cis man", "Cis Male", "cis male"
Transgender	"trans-female", "something kinda male?", "queer/she/they", "non-binary","nah", "all", "enby", "fluid", "genderqueer", "androgyne", "agender", "male leaning androgynous", "guy (-ish) ^_^", "trans woman", "neuter", "female (trans)", "queer", "ostensibly male, unsure what that really means"
Female	"cis female", "f", "female", "woman", "femake", "female ","cis-female/femme", "female (cis)", "femail"

C. Penanganan Outlier pada Atribut Age

Ditemukan beberapa nilai yang tidak masuk akal (*outlier*) pada atribut "*Age"*, seperti angka negatif dan nilai yang jauh dari rentang usia manusia. Oleh karena itu, kami menangani *outlier* dengan mengganti nilai yang tidak logis dengan nilai median dari dataset.

```
s = pd.Series(train_df['Age'])
s[s<18] = train_df['Age'].median()
train_df['Age'] = s
s = pd.Series(train_df['Age'])
s[s>120] = train_df['Age'].median()
train_df['Age'] = s
```

Selain itu, untuk mempermudah analisis dan meningkatkan performa model, kami membuat atribut baru bernama "*age_range*", yang berisi pengelompokan usia ke dalam beberapa rentang sebagai berikut:

- 0-20 (Usia Anak & Remaja)
- 21-30 (Usia Dewasa Muda)
- 31-65 (Usia Produktif)
- 66-100 (Usia Lansia)

```
train_df['age_range'] = pd.cut(train_df['Age'], [0,20,30,65,100], labels=["0-20", "21-30", "31-65", "66-100"]
```

D. Penanganan Data Ambigu pada Atribut 'self_employed' dan 'work_interfere'

Pada atribut "*self_employed*" dan "*work_interfere*", terdapat beberapa nilai yang kosong (NaN) sehingga kami melakukan imputasi data dengan pendekatan berikut:

- Atribut "*self_employed*": Nilai NaN diisi dengan "*No*", karena mayoritas responden bukan wiraswasta.
- Atribut "*work_interfere*": Nilai NaN diisi dengan "*Don't Know*", karena kemungkinan responden tidak yakin atau tidak memberikan jawaban yang jelas mengenai dampak pekerjaan terhadap kesehatan mental mereka.

Dengan langkah ini, dataset menjadi lebih bersih dan dapat memberikan hasil yang lebih baik saat diterapkan dalam model machine learning.

```
train_df['self_employed'] = train_df['self_employed'].replace([defaultString], 'No')
print(train_df['self_employed'].unique())
```

```
train_df['work_interfere'] = train_df['work_interfere'].replace([defaultString], 'Don\'t know' )
print(train_df['work_interfere'].unique())
```

E. Encoding Data

Setelah menangani missing value tersebut, langkah selanjutnya adalah kami melakukan encoding data untuk mengonversi fitur-fitur kategorikal menjadi format numerik, sehingga dapat diproses oleh algoritma *machine learning*.

Kami menggunakan metode Label Encoding dengan LabelEncoder(), yang bekerja dengan cara mengubah setiap nilai unik dalam fitur kategorikal menjadi bilangan bulat. Proses ini diterapkan pada setiap fitur kategorikal dalam DataFrame train_df, di mana:

- Setiap kategori teks dalam fitur akan dikonversi menjadi representasi angka unik.
- Pemetaan nilai kategori ke angka disimpan dalam kamus **labelDict**, sehingga memungkinkan interpretasi kembali dari nilai numerik ke bentuk kategori aslinya jika diperlukan.

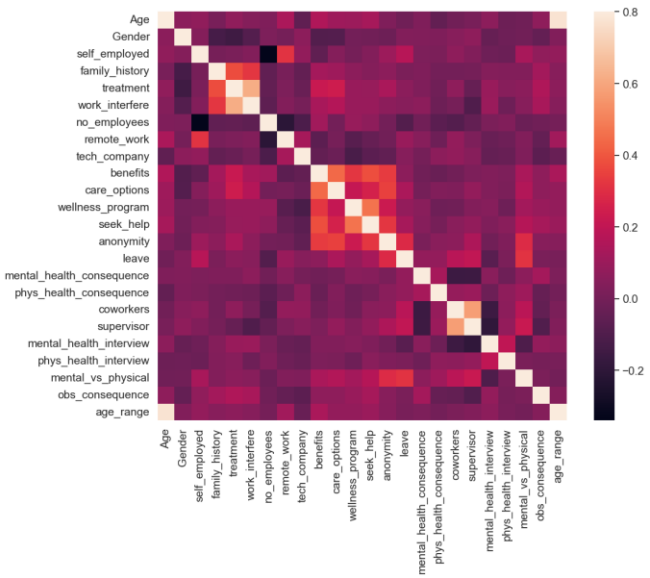
Metode ini membantu memastikan bahwa model dapat mengenali pola dalam data kategorikal tanpa kesulitan dalam memahami nilai non-numerik.

```
for feature in train_df:
    le = preprocessing.LabelEncoder()
    le.fit(train_df[feature])
    le_name_mapping = dict(zip(le.classes_, le.transform(le.classes_)))
    train_df[feature] = le.transform(train_df[feature])
    # Get labels
    labelKey = 'label_' + feature
    labelValue = [*le_name_mapping]
    labelDict[labelKey] =labelValue
```

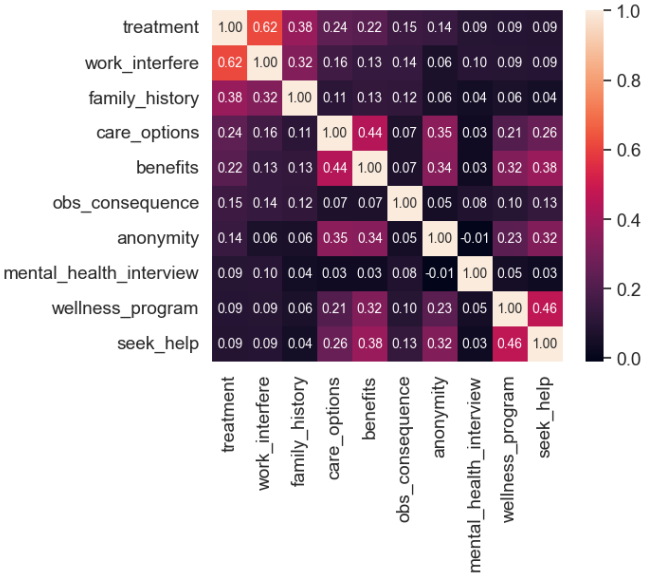
F. Korelasi Matriks

Selanjutnya kami melakukan analisis korelasi antar tiap fitur dalam dataset untuk memahami hubungan antara variabel. Korelasi ini membantu dalam mengidentifikasi fitur-fitur yang memiliki hubungan kuat satu sama lain serta menentukan apakah terdapat multikolinearitas yang dapat mempengaruhi performa model *machine learning.*

Untuk memvisualisasikan hasil analisis korelasi, kami menggunakan *heatmap* untuk memprmudah kita dalam mengidentifikasi tisp fitur dengan korelasi yang tinggi.



Berikut merupakan 10 korelasi tertinggi dalam fitur kami.

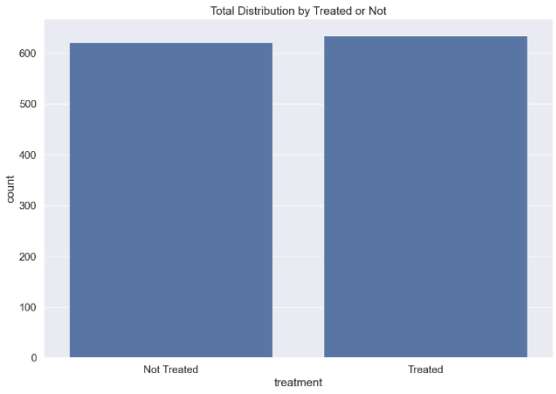


Dari hasil analisis korelasi yang divisualisasikan dalam gambar, dapat disimpulkan bahwa fitur *'work_interfere', 'family_history', 'care_options', 'benefits', 'obs_consequence'* dan *'anomity'* memiliki korelasi tertinggi dengan fitur '*treatment'* yang akan diprediksi.

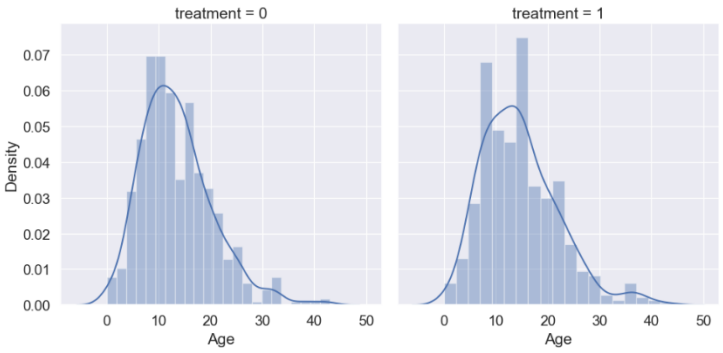
Korelasi yang tinggi ini menunjukkan bahwa variabel-variabel tersebut memiliki hubungan yang erat dengan kondisi treatment, dan oleh karena itu, fitur-fitur tersebut dianggap sangat relevan dalam memprediksi kondisi tersebut.

G. Distribusi Fitur

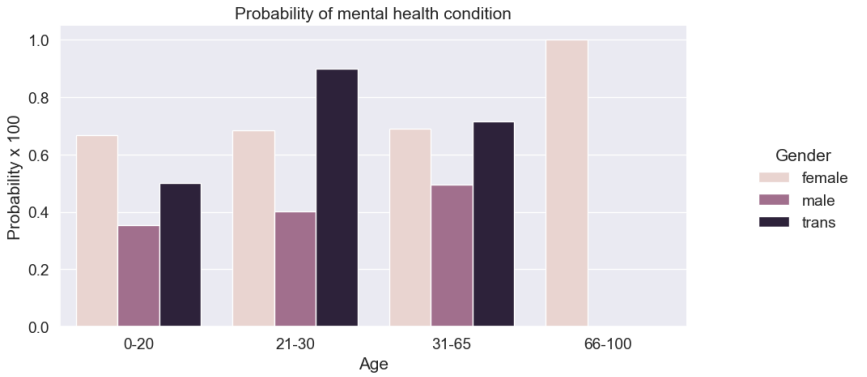
Selanjutnya, kami melakukan analisis distribusi untuk mengevaluasi bagaimana distribusi nilai dari setiap fitur terkait dengan probabilitas kondisi mental health yang ingin diprediksi. Distribusi ini memberikan gambaran yang lebih jelas mengenai bagaimana tiap fitur dapat mempengaruhi prediksi kondisi kesehatan mental, serta membantu dalam memahami pola data secara keseluruhan.

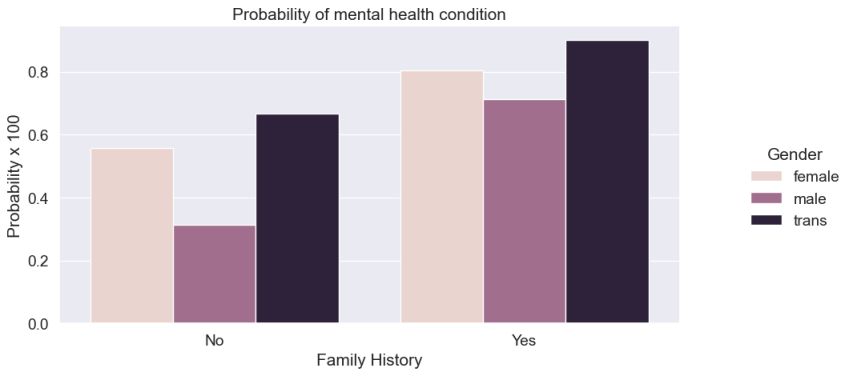


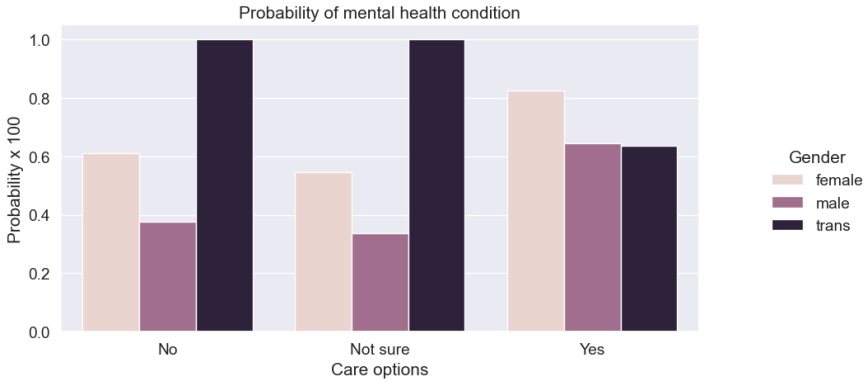
Grafik ini menggambarkan jumlah distribusi antara individu yang mendapatkan treatment dan yang tidak mendapatkan treatment. Dari visualisasi, dapat dilihat bahwa jumlah ***Treated*** dan ***Not Treated*** relatif seimbang, dengan jumlah individu dalam masing-masing kategori sekitar 600. Hal ini menunjukkan bahwa distribusi data dalam hal apakah seseorang mendapatkan treatment atau tidak cukup seimbang, yang sangat penting dalam model prediksi. Ketidakseimbangan dalam distribusi kategori dapat menyebabkan bias dalam prediksi, tetapi dalam hal ini, model dapat bekerja dengan baik tanpa kesulitan terkait distribusi yang tidak merata.

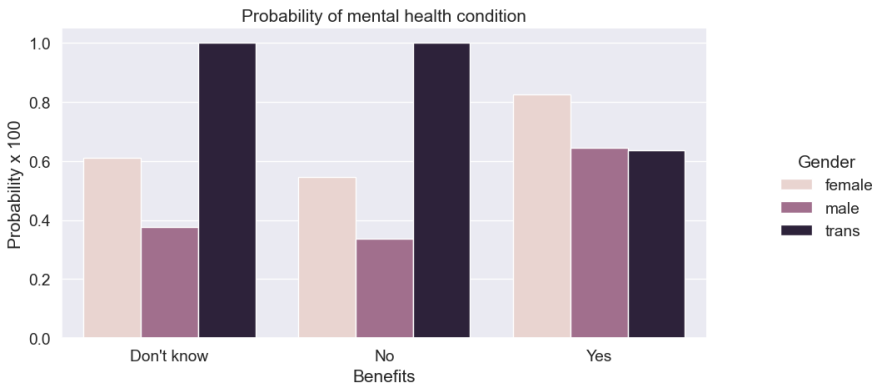


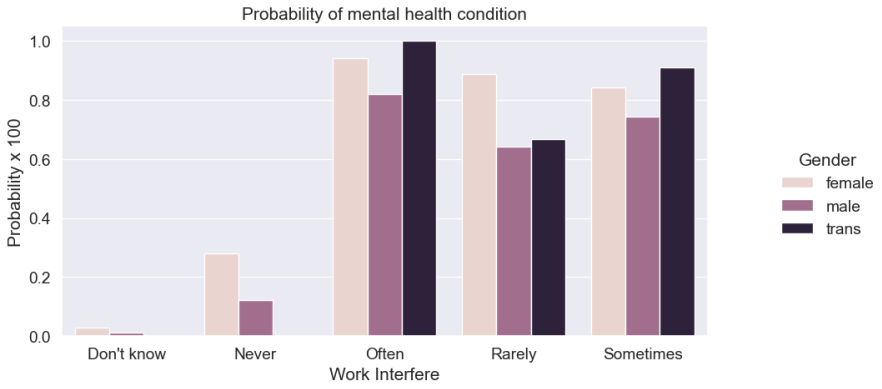
Treated memiliki distribusi usia yang lebih terpusat di sekitar usia muda, sedangkan ***Not Treated*** cenderung memiliki distribusi usia yang lebih tersebar di berbagai kelompok usia. Ini mengindikasikan bahwa orang yang tidak diberi perawatan mungkin lebih beragam dalam hal usia, sedangkan mereka yang mendapat perawatan cenderung lebih muda.


Probability of mental health condition
Probability x 100
1.0
0.8
0.6
0.4
0.2
0.0
0-20
21-30
31-65
66-100
Age
Gender
female
male
trans


Probability of mental health condition
Probability x 100
0.8
0.6
0.4
0.2
0.0
No
Yes
Family History
Gender
female
male
trans


Probability of mental health condition
Probability x 100
1.0
0.8
0.6
0.4
0.2
0.0
No
Not sure
Yes
Care options
Gender
female
male
trans


Probability of mental health condition
Probability x 100
1.0
0.8
0.6
0.4
0.2
0.0
Don't know
No
Yes
Benefits
Gender
female
male
trans



Secara keseluruhan, grafik-grafik ini memberikan informasi mengenai bagaimana usia, jenis kelamin, perawatan, manfaat, dan gangguan pekerjaan berhubungan dengan kondisi kesehatan mental pada individu.

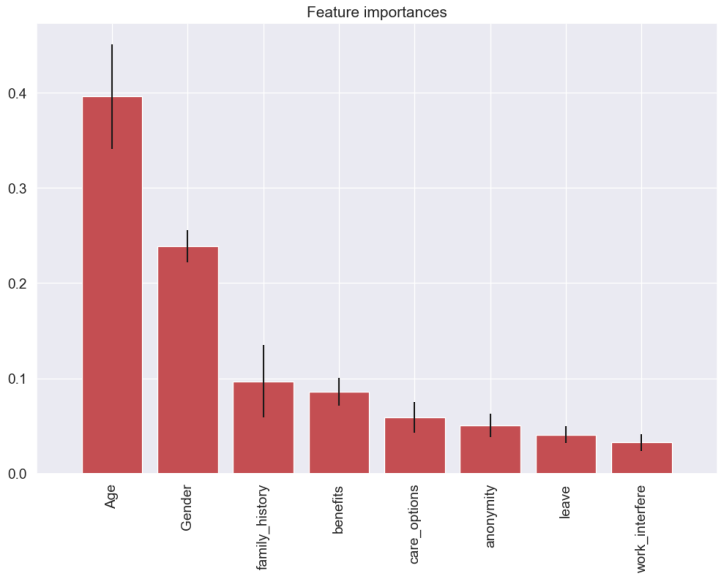
H. *Scalling Data*

Kami melakukan scaling pada fitur "*Age"* untuk memastikan bahwa nilai fitur berada dalam skala yang seragam, sehingga tidak ada fitur yang dominan dalam proses pelatihan model. *Scaling* ini bertujuan untuk menghindari model memberikan bobot yang berlebihan pada fitur dengan rentang nilai yang lebih besar. Proses *scaling* ini akan membantu model untuk lebih efektif dalam mempelajari pola dari data.

```
# Scaling Age
scaler = MinMaxScaler()
train_df['Age'] = scaler.fit_transform(train_df[['Age']])
train_df.head()
```

I. *Feature Importance*

Untuk memilih fitur-fitur yang paling berpengaruh terhadap hasil prediksi, kami melakukan *feature importance*. Dari 27 fitur yang tersedia dalam dataset, kami memutuskan untuk menggunakan hanya 7 fitur utama yang memiliki kontribusi terbesar terhadap prediksi treatment, sesuai dengan hasil dari analisis *feature importance* yang ditunjukkan pada gambar. Pemilihan fitur yang relevan ini bertujuan untuk meningkatkan performa model dan mengurangi kompleksitas model yang tidak perlu.



Setelah melakukan berbagai tahapan pada pemrosesan data, selanjutnya kami membagi data menjadi dua bagian: *data training* dan *data testing*, dengan proposi pembagian (70:30). *Data training* akan digunakan untuk melatih model machine learning, sementara *data testing* akan digunakan untuk mengevaluasi performa model setelah dilatih.

III. IMPLEMENTASI

1. Spesifikasi Perangkat Keras:
    - Perangkat: Acer Swift X
    - Prosesor: 12th Gen Intel® Core™ i7-1260P dengan kecepatan 2.10 GHz
    - RAM Terpasang: 16.0 GB (15.7 GB dapat digunakan)
    - Sistem Operasi: 64-bit operating system, x64-based processor
    - GPU: NVIDIA GeForce RTX 3050 Ti (spesifikasi dapat bervariasi tergantung model perangkat)
    - *Catatan:* Spesifikasi GPU dapat diperiksa melalui Device Manager atau software monitoring GPU.

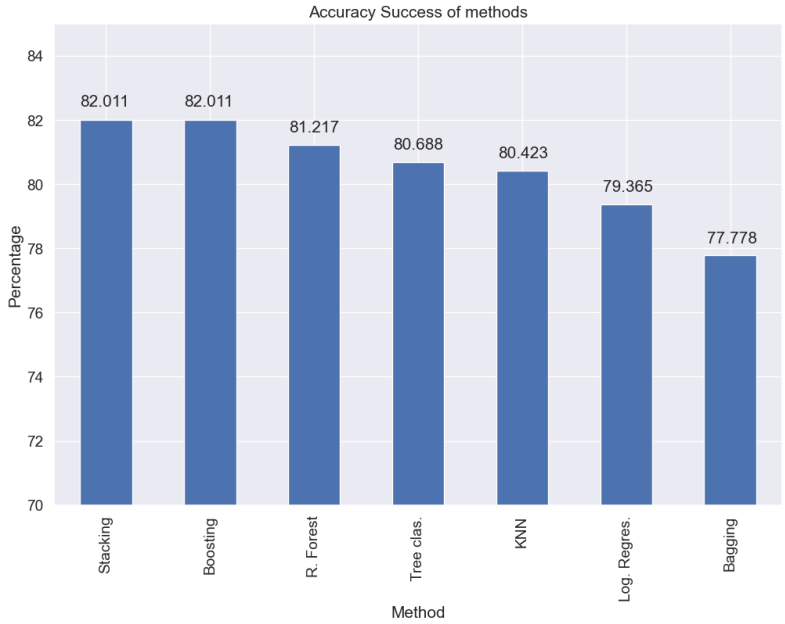
2. Perangkat Lunak dan Platform:
    - **Visual Studio Code:** digunakan sebagai *Integrated Development Environment* (IDE) untuk menulis dan mengembangkan kode Python. VS Code menyediakan berbagai fitur seperti *syntax highlighting, autocompletion, debugging*, dan

extensions yang sangat mendukung pengembangan aplikasi *machine learning* secara efisien.

- Versi Python: 3.12.6
- **Flask:** digunakan untuk deployment model machine learning. Dengan Flask, kami membangun aplikasi web sederhana yang dapat menerima input dari pengguna, melakukan prediksi menggunakan model machine learning, dan mengirimkan hasil prediksi kembali ke pengguna. Flask memungkinkan aplikasi untuk dijalankan secara lokal atau di server dan dapat diakses melalui antarmuka berbasis web.
- Python Libraries yang Digunakan:
  - NumPy: Untuk manipulasi array dan operasi numerik, sangat berguna dalam pemrosesan data skalar.
  - Pandas: Untuk manipulasi dan analisis data, termasuk fungsi-fungsi untuk pembersihan dan transformasi data.
  - Matplotlib & Seaborn: Untuk visualisasi data dalam bentuk grafik, histogram, dan heatmap.
  - Scikit-learn: Untuk preprocessing data, pengembangan model, dan implementasi berbagai algoritma machine learning.
  - SciPy: Untuk operasi statistik dan distribusi probabilitas.
  - mlxtend: Untuk algoritma *Stacking*, yang digunakan dalam model ensemble untuk meningkatkan performa prediksi.
  - Joblib: Untuk menyimpan dan memuat model *machine learning*, memungkinkan model untuk digunakan kembali tanpa perlu pelatihan ulang.

IV. HASIL DAN PEMBAHASAN

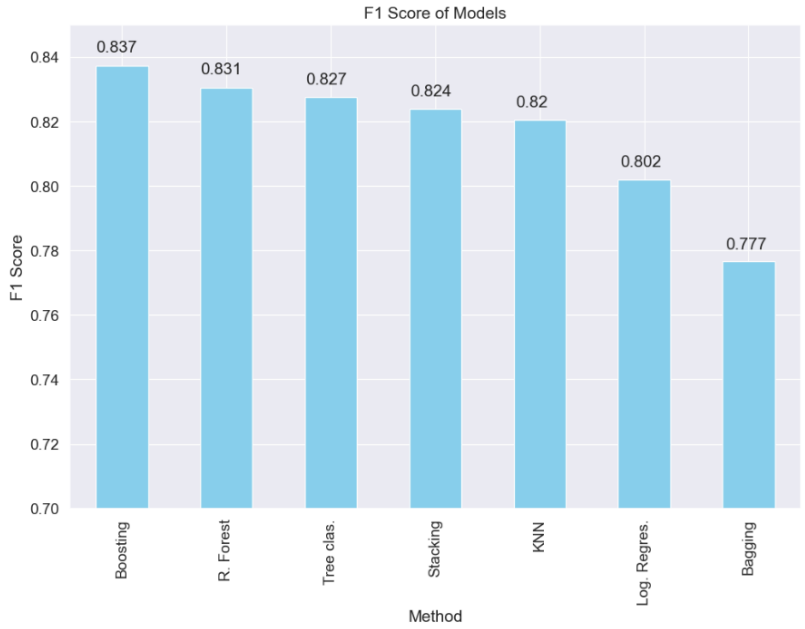
1. *Classification Accuracy*



Classification accuracy adalah metrik evaluasi yang digunakan untuk mengukur kinerja model klasifikasi dalam memprediksi kelas yang benar dari data. Dalam grafik yang divisualisasikan, sumbu vertikal merepresentasikan persentase prediksi yang benar dari keseluruhan data uji untuk setiap metode klasifikasi. Semakin tinggi batang pada grafik, semakin baik model tersebut dalam mengklasifikasikan data dengan benar.

Berdasarkan hasil yang divisualisasikan, metode ***Stacking*** dan ***Boosting*** menunjukkan nilai *classification accuracy* yang sama tinggi, yaitu 82.011%. Ini mengindikasikan bahwa kedua model ini lebih unggul dalam tugas klasifikasi dibandingkan dengan metode lainnya yang diuji dalam percobaan ini.

2. *F1-Score*



F1-Score adalah metrik evaluasi yang lebih komprehensif dalam klasifikasi dibandingkan dengan *accuracy,* terutama pada kasus dataset yang tidak seimbang. *F1-Score* mengukur rata-rata harmonik antara *precision* dan *recall,* memberikan gambaran seimbang tentang kemampuan model dalam mengklasifikasikan kelas positif.

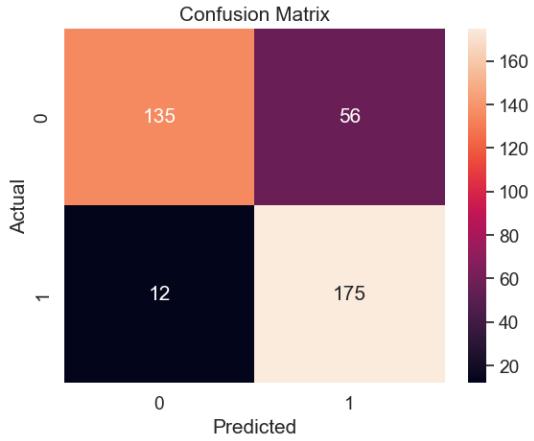
- ***Precision*** mengukur proporsi prediksi positif yang benar dari total prediksi positif, yang menggambarkan ketepatan model dalam memprediksi kelas positif.
- ***Recall*** mengukur proporsi instance positif yang sebenarnya yang berhasil diidentifikasi oleh model, atau dengan kata lain, sensitivitas model terhadap kelas positif.

Grafik "*F1 Score of Models*" menunjukkan nilai *F1-Score* yang dicapai oleh berbagai metode klasifikasi. Sumbu vertikal menunjukkan nilai *F1-Score*, di mana nilai yang lebih tinggi mengindikasikan performa yang lebih baik dalam menyeimbangkan *precision* dan *recall.* Berdasarkan hasil tersebut, ***Boosting*** dan ***Random Forest*** menunjukkan nilai *F1-Score* tertinggi, yaitu 0.837 dan 0.831 secara berurutan. Ini menandakan bahwa kedua model ini memiliki performa terbaik dalam tugas klasifikasi berdasarkan metrik *F1-Score*, menunjukkan kemampuan yang unggul dalam mencapai keseimbangan antara ketepatan dan kelengkapan dalam mengidentifikasi kelas positif.

Performa ***Boosting*** berdasarkan metrik-metrik lain:

1. ***Confusion matrix***

Confusion matrix adalah alat visualisasi yang merinci kinerja model klasifikasi dengan membandingkan prediksi model dengan nilai aktual. Matriks ini membagi hasil prediksi menjadi empat kategori utama, yang masing-masing mengindikasikan performa model dalam klasifikasi:



- *True Positives* (TP): Prediksi positif benar, 175),
- *True Negatives* (TN): Prediksi negatif benar, 135),
- *False Positives* (FP): Prediksi positif salah, 56),
- *False Negatives* (FN): Prediksi negatif salah, 12).

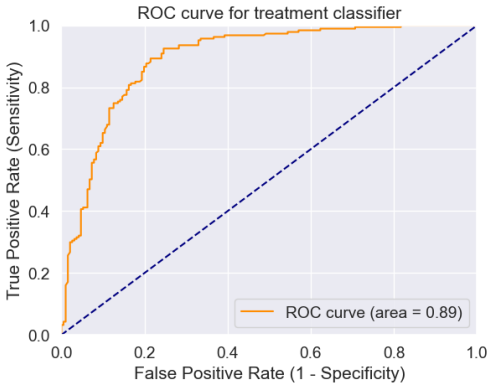
Dalam konteks performa model Boosting, confusion matrix ini menunjukkan bahwa model memiliki kemampuan yang sangat baik dalam mengidentifikasi kelas positif. Dengan 175 prediksi positif yang benar dari 187 kasus positif aktual, model ini memiliki *recall* yang tinggi. Artinya, model cenderung berhasil menangkap sebagian besar kasus positif.

Namun, terdapat juga sejumlah *False Positives* (56) yang cukup signifikan, yang menunjukkan bahwa model sering kali salah memprediksi kelas negatif sebagai positif. Ini menunjukkan adanya kecenderungan model untuk lebih konservatif dalam mengklasifikasikan kelas negatif sebagai negatif, yang bisa berisiko dalam beberapa konteks jika terlalu banyak *false positives*.

Secara keseluruhan, *confusion matrix* ini mengonfirmasi bahwa model *Boosting* memiliki performa yang solid dalam klasifikasi, khususnya dalam mengidentifikasi kelas positif. Hal ini juga sejalan dengan nilai *accuracy* dan *F1-Score* yang tinggi yang telah dibahas sebelumnya.

2. **Kurva ROC (*Receiver Operating Characteristic*)**

Kurva ROC adalah grafik yang digunakan untuk memvisualisasikan kinerja model klasifikasi biner pada berbagai ambang batas klasifikasi. Kurva ini menggambarkan hubungan antara *False Positive Rate* (FPR) (sumbu horizontal) dan *True Positive Rate* (TPR) (sumbu vertikal):



- ***False Positive Rate* (FPR):** Proporsi instance negatif yang salah diprediksi sebagai positif, juga dikenal sebagai ***(1 - Specificity)***.
- ***True Positive Rate* (TPR):** Proporsi instance positif yang berhasil diprediksi dengan benar, juga dikenal sebagai ***Sensitivity***.

Garis diagonal putus-putus pada kurva ROC menunjukkan kinerja model acak**,** yang tidak dapat membedakan antara kelas positif dan negatif. Sebaliknya, kurva yang lebih jauh dari diagonal menunjukkan model dengan kemampuan diskriminasi yang lebih baik antara kelas-kelas tersebut.

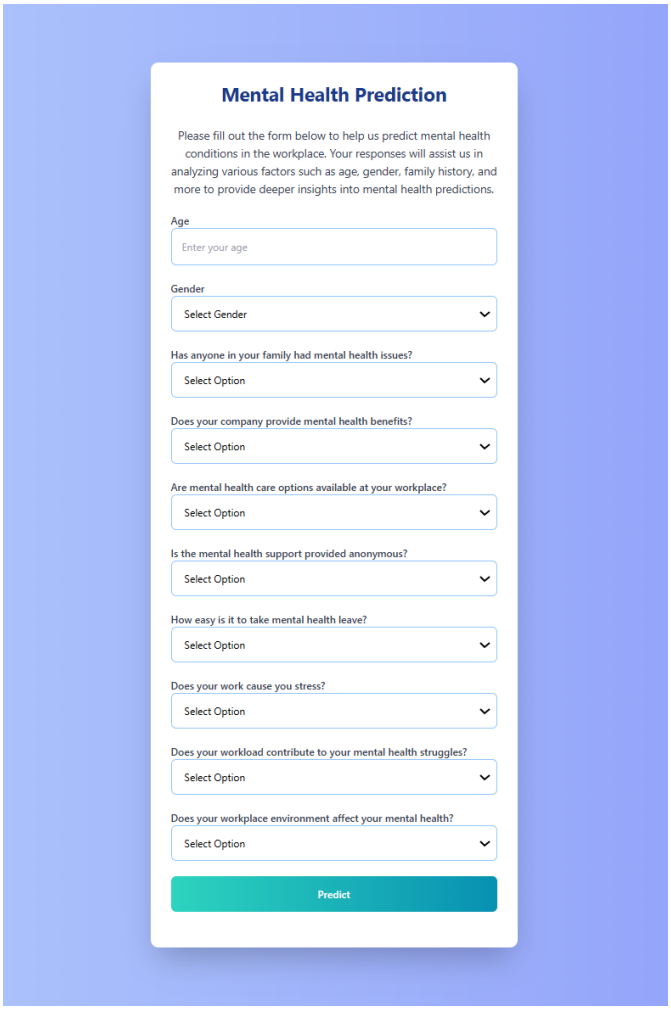
Kurva ROC untuk model treatment classifier (*Boosting)* menunjukkan *Area Under the Curve* (AUC) sebesar 0.89. Nilai AUC ini mengukur luas area di bawah kurva ROC, yang secara umum semakin tinggi nilai AUC (mendekati 1), semakin baik performa model dalam membedakan antara kelas positif dan negatif.

Dengan AUC sebesar 0.89, yang mendekati nilai maksimum 1, model Boosting ini menunjukkan kemampuan yang sangat baik dalam membedakan antara kelas positif dan negatif. Ini mengindikasikan bahwa model dapat secara efektif memisahkan kelas positif dan negatif dengan tingkat *false positive rate* yang rendah dan *true positive rate* yang tinggi pada berbagai ambang batas klasifikasi.

Setelah melakukan evaluasi dan membandingkan kinerja berbagai metode klasifikasi, model Boosting yang telah dikembangkan menunjukkan performa yang sangat baik dalam berbagai metrik evaluasi, termasuk accuracy, F1-Score, confusion matrix, dan ROC-AUC.

Langkah selanjutnya adalah mendeploy model ini ke dalam lingkungan produksi agar dapat digunakan dalam proses prediksi secara real-time. Deployment model akan memastikan bahwa model dapat memberikan prediksi secara konsisten dan akurat terhadap data baru yang masuk.

Berikut adalah hasil dari deploy yang telah kami buat:



User dapat menginput data sesuai dengan kondisi yang dialaminya. Setelah selesai, user dapat mengklik tombol Predict untuk memprediksi apakah dia membutuhkan treatment atau tidak.

Hasil prediksi adalah sebagai berikut:

Mental Health is Stable.
Maintain a healthy lifestyle, manage stress well, and keep communicating with your support system to ensure your mental well-being stays strong.

Mental Health at Risk.
It is recommended to seek professional help, such as counseling or therapy, and discuss your condition with HR or your supervisor for appropriate support.

KESIMPULAN

Berdasarkan hasil evaluasi kinerja beberapa model machine learning untuk memprediksi kebutuhan treatment, dapat disimpulkan bahwa metode ***Stacking*** dan ***Boosting*** menunjukkan performa terbaik secara keseluruhan dalam hal *classification accuracy* dan *F1-Score*. Kedua model ini secara konsisten mencapai tingkat akurasi dan *F1-Score* tertinggi dibandingkan dengan metode lain seperti *Random Forest, Decision Tree,* KNN, *Logistic Regression*, dan

Bagging, menandakan keunggulan mereka dalam tugas klasifikasi pada dataset yang digunakan.

Analisis lebih mendalam terhadap model Boosting menggunakan *confusion matrix* dan kurva ROC mengonfirmasi performa yang solid. *Confusion matrix* mengungkapkan bahwa model Boosting sangat efektif dalam mengidentifikasi kasus positif (*recall* tinggi), dengan jumlah *false negative* yang rendah. Meskipun terdapat jumlah *false positive* yang lebih signifikan, model ini tetap menunjukkan keseimbangan yang baik dalam memprediksi kelas positif dan negatif. Lebih lanjut, kurva ROC dengan nilai AUC sebesar 0.89 menegaskan kemampuan diskriminatif model Boosting yang sangat baik, mampu membedakan antara kelas positif dan negatif secara efektif pada berbagai ambang batas klasifikasi.

Secara ringkas, model Boosting muncul sebagai salah satu metode yang paling menjanjikan untuk memprediksi kebutuhan treatment dalam studi ini. Kinerja yang unggul dalam metrik *accuracy*, *F1-Score* (0.837), serta interpretasi dari *confusion matrix* dan kurva ROC, menunjukkan bahwa Boosting memiliki kemampuan klasifikasi yang handal dan efektif. Meskipun perlu mempertimbangkan implikasi dari *false positive* dalam konteks aplikasi prediksi treatment, performa keseluruhan model Boosting memberikan dasar yang kuat untuk pertimbangan lebih lanjut dalam pengembangan sistem prediksi kebutuhan treatment yang akurat dan efisien.

DAFTAR PUSTAKA

Bhui, K. S., Dinos, S., Stansfeld, S. A., & White, P. D. (2016). A synthesis of the evidence for managing stress at work: A review of the reviews reporting on anxiety, depression, and absenteeism. *Journal of Environmental and Public Health*, 2016, 1-12.

Deloitte. (2020). *Mental health and employers: The case for investment*. Deloitte Insights.

Open Sourcing Mental Illness (OSMI). (2014). *Mental health in tech survey 2014*. Kaggle. https://www.kaggle.com/datasets/osmi/mental-health-in-tech-survey/data

World Health Organization (WHO). (2021). *Mental health: Strengthening our response*. https://www.who.int/news-room/fact-sheets/detail/mental-health-strengthening-our-response