Emotion Detection from Text: A Web-Based Sentiment Analysis Application



disusun oleh :

Anggun Vannia : A11.2022.14106

Teknik Informatika

Fakultas Ilmu Komputer

Universitas Dian Nuswantoro

2025

# Business Understanding (BU)

Sistem deteksi emosi berbasis web adalah aplikasi yang dirancang untuk mengenali emosi pengguna secara otomatis melalui data yang dikumpulkan secara online. Data yang diperoleh diproses melalui model pembelajaran mesin yang telah dilatih dengan dataset emosi untuk menghasilkan hasil yang akurat. Sehingga sistem ini dapat menganalisis jenis data teks, untuk mengidentifikasi emosi seperti senang, sedih, tenang, lelah, atau netral.

Dalam era digital, pengenalan dan pemahaman emosi dari teks sangat penting bagi banyak perusahaan untuk meningkatkan pengalaman pengguna, layanan pelanggan, dan memahami kebutuhan audiens. Aplikasi ini relevan di berbagai bidang, seperti:

* Layanan Pelanggan: Untuk mendeteksi keluhan pelanggan, kepuasan, atau apresiasi, memungkinkan respons yang lebih tepat.
* Media Sosial: Untuk memahami reaksi audiens terhadap konten yang dipublikasikan, seperti komentar positif, nesgatif, atau netral.
* Sistem Pendidikan dan Pelatihan: Untuk mendeteksi tingkat stres, kebahagiaan, atau ketidakpuasan pada siswa atau peserta pelatihan.

Dengan perkembangan teknologi dan semakin banyaknya data teks yang tersedia secara online, sistem deteksi emosi berbasis web memiliki potensi besar untuk terus berkembang. Kombinasi teknik pembelajaran mesin modern, seperti deep learning, dengan dataset emosi yang lebih beragam dan representatif akan meningkatkan akurasi dan efisiensi sistem ini, menjadikannya alat yang sangat berharga di berbagai industri.

# Permasalahan Project

Dalam pengembangan sistem deteksi emosi berbasis teks, terdapat berbagai tantangan yang harus diatasi untuk memastikan akurasi dan keandalan model. Tantangan ini muncul karena sifat kompleks dari emosi manusia, keragaman cara emosi diekspresikan, dan keterbatasan dalam data atau teknologi yang digunakan. Sistem harus mampu memahami variasi ekspresi emosional yang sering kali tidak eksplisit atau terkesan subjektif. Selain itu, keterbatasan data dan keanekaragaman budaya juga menambah tingkat kesulitan dalam membangun model yang dapat digunakan secara universal.

Beberapa tantangan dalam pengembangan sistem deteksi emosi meliputi:

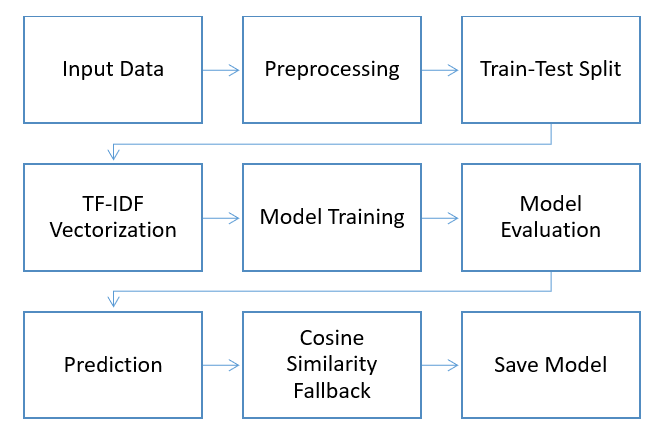
* Variasi Bahasa: Ekspresi emosi dapat berbeda tergantung pada variasi bahasa dan cara pengguna menyampaikan perasaan mereka.
* Ambiguitas Emosi: Beberapa teks mungkin mengandung emosi yang ambigu atau campuran, sehingga sulit untuk dikategorikan ke dalam satu emosi sehingga menghasilkan output neutral ketika diinput.
* Jumlah Data Terbatas: Jumlah data pelatihan yang kecil bisa memengaruhi akurasi model, terutama untuk emosi yang jarang muncul.

# Tujuan

Tujuan utama dari sistem deteksi emosi berbasis teks adalah untuk memahami dan mengklasifikasikan emosi pengguna atau pelanggan secara otomatis dan akurat, sehingga dapat memberikan wawasan yang bermanfaat untuk berbagai kebutuhan, seperti meningkatkan pengalaman pengguna, mendukung pengambilan keputusan, dan mempermudah analisis sentimen dalam skala besar. Berikut adalah beberapa tujuan spesifik dari sistem ini:

* Memberikan Insight kepada Pengguna/Pelanggan: Menghasilkan pemahaman tentang perasaan pengguna atau pelanggan yang dapat dimanfaatkan oleh tim layanan pelanggan, pemasaran, atau tim produk.
* Meningkatkan Pengalaman Pengguna: Mengimplementasikan sistem responsif yang dapat menangani teks dengan emosi tertentu, sehingga dapat memberikan solusi yang lebih cepat dan personal.
* Mempermudah Analisis Emosi secara Otomatis: Menyediakan alat analisis emosi otomatis yang memudahkan perusahaan dalam memantau dan menganalisis sentimen pelanggan atau audiens dalam jumlah besar tanpa intervensi manual.

# Alur Penyelesaian



* 1. Pembacaan dan Persiapan Data

Data dibaca dari file CSV menggunakan pandas.read\_csv(), kemudian disimpan dalam sebuah DataFrame (df). Setelah itu, dilakukan pengecekan data untuk memastikan bahwa tidak ada nilai NaN pada kolom 'text' dan 'label' menggunakan dropna().

* 1. Preprocessing Data

Label Encoding: Kolom label yang merupakan variabel kategorikal (berisi emosi dalam bentuk angka) diubah ke format numerik menggunakan astype('category').cat.codes. digunakan untuk mengonversi label-label kategori menjadi nilai numerik yang lebih mudah digunakan dalam model.

Pembagian Data: Data dibagi menjadi data pelatihan (80%) dan data uji (20%) menggunakan train\_test\_split(). Proses pembagian ini menggunakan parameter stratify=df['label'] untuk memastikan distribusi label pada kedua subset tetap seimbang.

* 1. Vektorisasi Teks menggunakan TF-IDF

Teks diubah menjadi representasi numerik menggunakan TfidfVectorizer dari sklearn. Teknik ini mengubah teks menjadi fitur-fitur numerik yang merepresentasikan frekuensi kata-kata yang penting dalam dokumen, dengan mempertimbangkan frekuensi kata di seluruh dokumen.

Parameter yang digunakan dalam TfidfVectorizer:

* + max\_features=10000 membatasi jumlah fitur yang digunakan hingga 10.000 kata.
  + stop\_words='english' menghilangkan kata-kata umum dalam bahasa Inggris yang tidak membawa banyak informasi.
  + ngram\_range=(1, 3) menangkap unigram, bigram, dan trigram untuk menangkap konteks kata yang lebih luas.
  1. Pencarian Model Optimal menggunakan GridSearchCV

Sebuah model regresi logistik (Logistic Regression) dengan parameter class\_weight='balanced' (untuk menangani ketidakseimbangan kelas) dan random\_state=42 untuk memastikan hasil yang konsisten digunakan. Pencarian Hyperparameter: GridSearchCV digunakan untuk mencari kombinasi terbaik dari beberapa parameter:

* + C (parameter regulasi),
  + solver (metode optimasi),
  + max\_iter (jumlah iterasi maksimum).

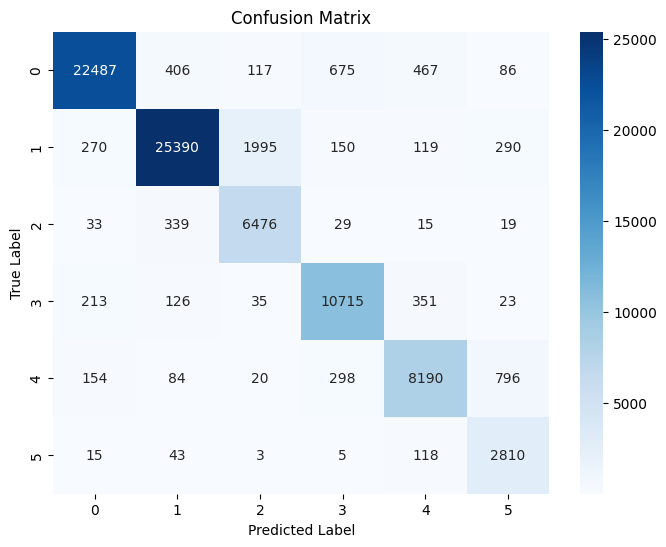
GridSearchCV menggunakan cross-validation dengan 3 lipatan untuk memastikan bahwa model yang terpilih tidak overfitting dan memberikan hasil yang terbaik.

* 1. Evaluasi Model

Setelah menemukan model terbaik melalui GridSearchCV, model dievaluasi menggunakan data uji (X\_test\_vec).



Metrik Evaluasi: Akurasi, laporan klasifikasi (precision, recall, f1-score), dan matriks kebingunguan (confusion matrix) digunakan untuk mengukur kinerja model.

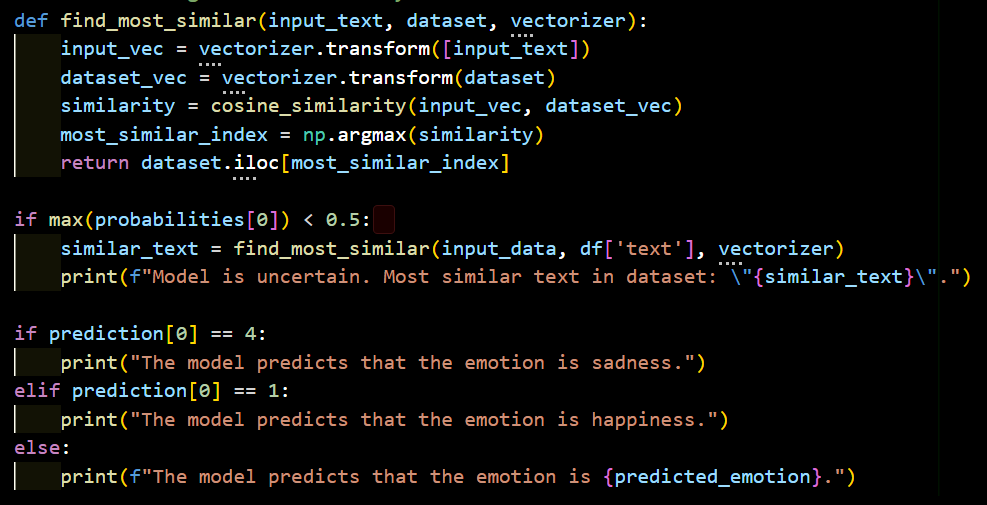


Matriks kebingunguan divisualisasikan menggunakan heatmap untuk menunjukkan sejauh mana prediksi model sesuai dengan label yang sebenarnya.

* 1. Prediksi dan Fallback menggunakan Cosine Similarity

Prediksi Teks: Model digunakan untuk memprediksi emosi berdasarkan input teks.

Cosine Similarity: Jika model tidak cukup yakin (confidence < 0.5), sistem melakukan fallback dengan mencari teks yang paling mirip dalam dataset menggunakan cosine similarity. Teks yang paling mirip ditampilkan sebagai fallback untuk memberikan informasi tambahan kepada pengguna.



* 1. Penyimpanan Model dan Vektorisasi

Model yang telah dilatih dan vektorisasi disimpan menggunakan pickle.dump() untuk digunakan pada file emotion\_predict.py tanpa perlu melatih ulang.

# Dataset

1. Penjelasan Dataset

Dataset diunduh melalui halaman proyek Kaggle yang relevan. Pengunduhan ini dilakukan dalam bentuk file ZIP yang berisi file CSV. Format data yang digunakan adalah CSV, format yang umum digunakan untuk menyimpan data dalam bentuk table. Data disimpan dalam format teks dengan setiap nilai dipisahkan oleh koma, sehingga mudah untuk dibaca dan diproses oleh Python.

Repository : <https://www.kaggle.com/datasets/nelgiriyewithana/emotions/data>

Dataset ini berisi data teks dengan kolom:

* Text: Berisi kalimat yang mencerminkan emosi pengguna.
* Label: Kode numerik yang merepresentasikan kategori emosi dari teks tersebut. Dari dataset di atas, label terdiri dari angka 0 hingga 5.

Dalam pembuatan model deteksi emosi, data yang digunakan terdiri dari teks dalam bahasa Inggris yang berisi kalimat atau frasa pendek yang mengungkapkan emosi, dilengkapi dengan label emosi dalam bentuk numerik.

Setiap baris teks mengandung ekspresi perasaan atau emosi tertentu, dan tujuan utamanya adalah memetakan teks tersebut pada kategori emosi yang tepat berdasarkan konteks kalimat.

1. EDA
2. Analisis Kolom text (Fitur Utama)

* Jenis Data: Kolom text berisi teks yang mencerminkan berbagai emosi. Kolom ini merupakan fitur utama yang diubah menjadi representasi numerik menggunakan teknik TF-IDF.
* Panjang Teks: Panjang teks dalam setiap baris sangat bervariasi. Misalnya:
  + Beberapa teks sangat pendek ("I feel lost"), sedangkan yang lain panjang dengan detail tambahan.
  + Distribusi panjang teks dapat dianalisis untuk memahami karakteristik data.
* Konten Teks: Teks terdiri dari bahasa informal dan ekspresi sehari-hari, sehingga model harus mampu menangani variasi seperti berikut ini.
  + Terdapat pola emosi eksplisit seperti "helpless", "happy", atau "disheartened", yang membantu klasifikasi.
  + Frase ambigu, seperti "I feel weird", akan lebih sulit diklasifikasikan karena konteks tidak eksplisit.

1. Analisis Kolom (Target)

Merupakan kolom numerik yang menunjukkan kelas atau kategori emosi yang sesuai dengan teks tersebut. Dalam data ini, label emosi berupa angka, misalnya:

0: Netral – Kondisi tanpa emosi tertentu atau perasaan yang tidak kuat.

1: Bahagia atau Senang – Menunjukkan perasaan positif atau kebahagiaan.

2: Bersemangat atau Antusias – Menunjukkan perasaan penuh energi.

3: Tenang atau Damai – Menunjukkan perasaan yang damai, santai, atau tenang.

4: Sedih atau Negatif – Menunjukkan perasaan sedih, kecewa, atau putus asa.

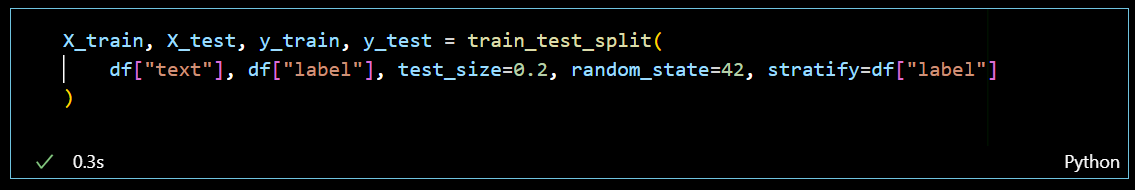
5: Lelah atau Lesu – Menunjukkan perasaan lelah, capek, atau kurang energi.

Label ini berfungsi sebagai *target* atau *output* dari model klasifikasi, yang akan digunakan untuk melatih model agar mampu mengklasifikasikan teks sesuai dengan emosi yang terkandung di dalamnya

Distribusi Label: Dengan menggunakan parameter stratify saat pembagian data latih dan uji, model mempertahankan proporsi label. Namun, perlu dipastikan apakah label dalam dataset asli seimbang. Jika tidak:

* + Ketidakseimbangan label (misalnya, label tertentu jauh lebih sedikit) dapat memengaruhi akurasi model.
  + class\_weight='balanced' telah digunakan untuk menangani potensi ketidakseimbangan.

1. Pembagian Data (Train-Test Split)

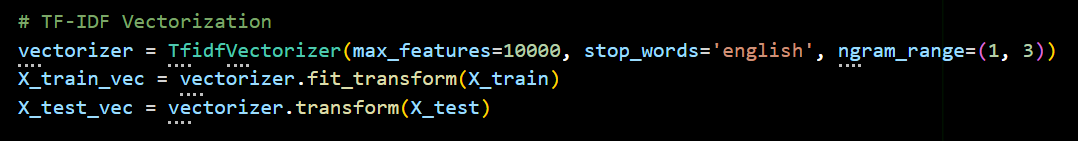


* Proporsi Data: Data dibagi menjadi 80% data latih dan 20% data uji dengan parameter stratify untuk memastikan distribusi label serupa di kedua subset.
* Stratifikasi Penting: Mencegah model dilatih atau diuji pada distribusi label yang tidak representatif.

# Modelling

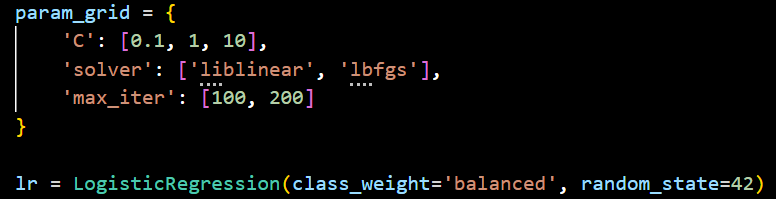
Pada tahap modelling, tujuan utama adalah untuk melatih model pembelajaran mesin (machine learning) yang dapat memprediksi label emosi berdasarkan teks input. Dalam proyek ini, digunakan model Logistic Regression, yang merupakan algoritma klasifikasi yang sering digunakan untuk tugas klasifikasi biner maupun multi-kelas. Logistic Regression bekerja dengan memodelkan hubungan antara fitur input (dalam hal ini, representasi numerik dari teks) dan probabilitas masing-masing kelas label (emosi). Kelebihan Logistic Regression adalah kesederhanaan, efisiensi komputasi, serta kemampuannya untuk menghasilkan probabilitas sebagai output, yang sangat berguna dalam pengklasifikasian emosi.

Untuk mengubah teks menjadi format yang dapat diproses oleh model, TF-IDF (Term Frequency-Inverse Document Frequency) digunakan sebagai teknik vektorisasi. TF-IDF mengukur seberapa penting suatu kata dalam sebuah dokumen relatif terhadap seluruh koleksi dokumen.



Dalam kode ini, jumlah fitur dibatasi hingga 10.000 dan n-gram (1 hingga 3 kata) digunakan untuk menangkap lebih banyak konteks dalam teks.

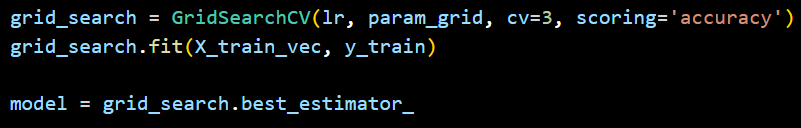
* + 1. Pembuatan Model dan Hyperparameter



Membuat model Logistic Regression (lr) dengan parameter default dan menyiapkan grid hyperparameter (param\_grid) untuk proses tuning.

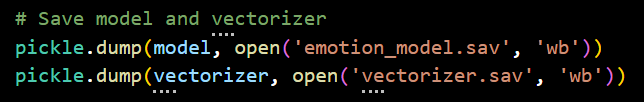
* + 1. Pelatihan Model dengan GridSearchCV

Selanjutnya, model dilatih menggunakan GridSearchCV untuk mencari kombinasi parameter terbaik seperti nilai C (regularization strength), solver (metode optimasi), dan max\_iter (jumlah iterasi maksimum). Dengan menggunakan cross-validation (CV) dalam GridSearchCV, model dilatih secara optimal tanpa overfitting pada data pelatihan.



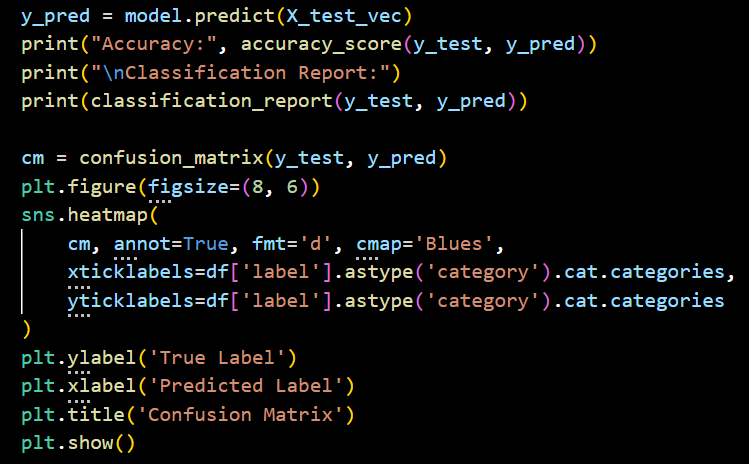
* GridSearchCV melatih model dengan berbagai kombinasi hyperparameter menggunakan data pelatihan (X\_train\_vec, y\_train).
* Model terbaik berdasarkan metrik akurasi dipilih dan disimpan dalam model.
  + 1. Menyimpan Model dan Vectorizer

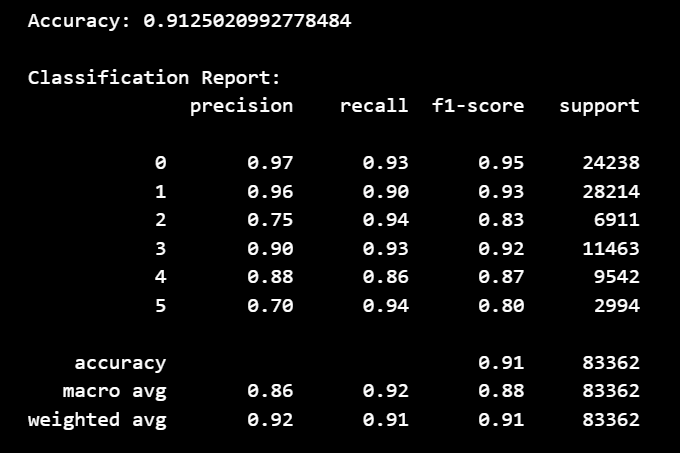
Setelah pemilihan model terbaik, model disimpan untuk digunakan pada prediksi selanjutnya. Selain itu, masalah ketidakseimbangan data ditangani menggunakan parameter class\_weight='balanced', yang memastikan model memberikan bobot yang lebih besar pada kelas-kelas yang kurang terwakili.

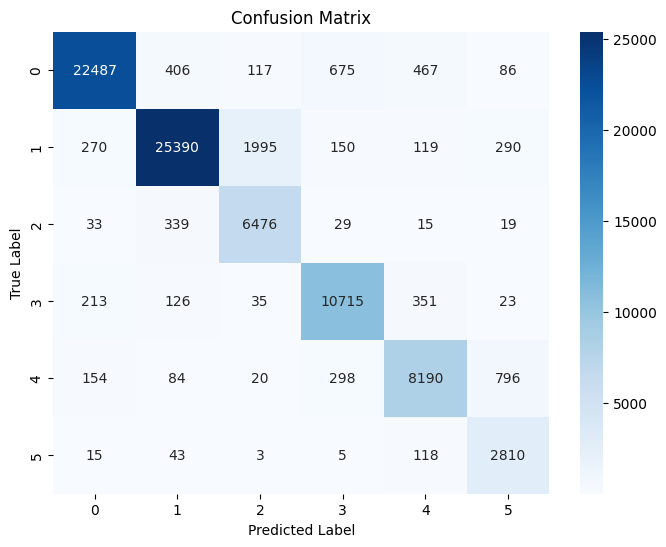


# Performa Model

Model memiliki akurasi sebesar 91.25%, menunjukkan kemampuan yang sangat baik dalam memprediksi sebagian besar data pada dataset pengujian. Kinerja tertinggi ditunjukkan oleh kelas *Neutral* (F1-Score: 0.95) dan *Happiness* (F1-Score: 0.93), dengan precision dan recall yang konsisten tinggi. Di sisi lain, kelas *Fatigue* memiliki performa terendah dengan F1-Score sebesar 0.80, akibat precision yang lebih rendah dibandingkan recall. Hal ini menunjukkan bahwa model sering salah memprediksi *Fatigue* sebagai kelas lain, meskipun cukup mampu mendeteksi saat kelas ini benar-benar ada.

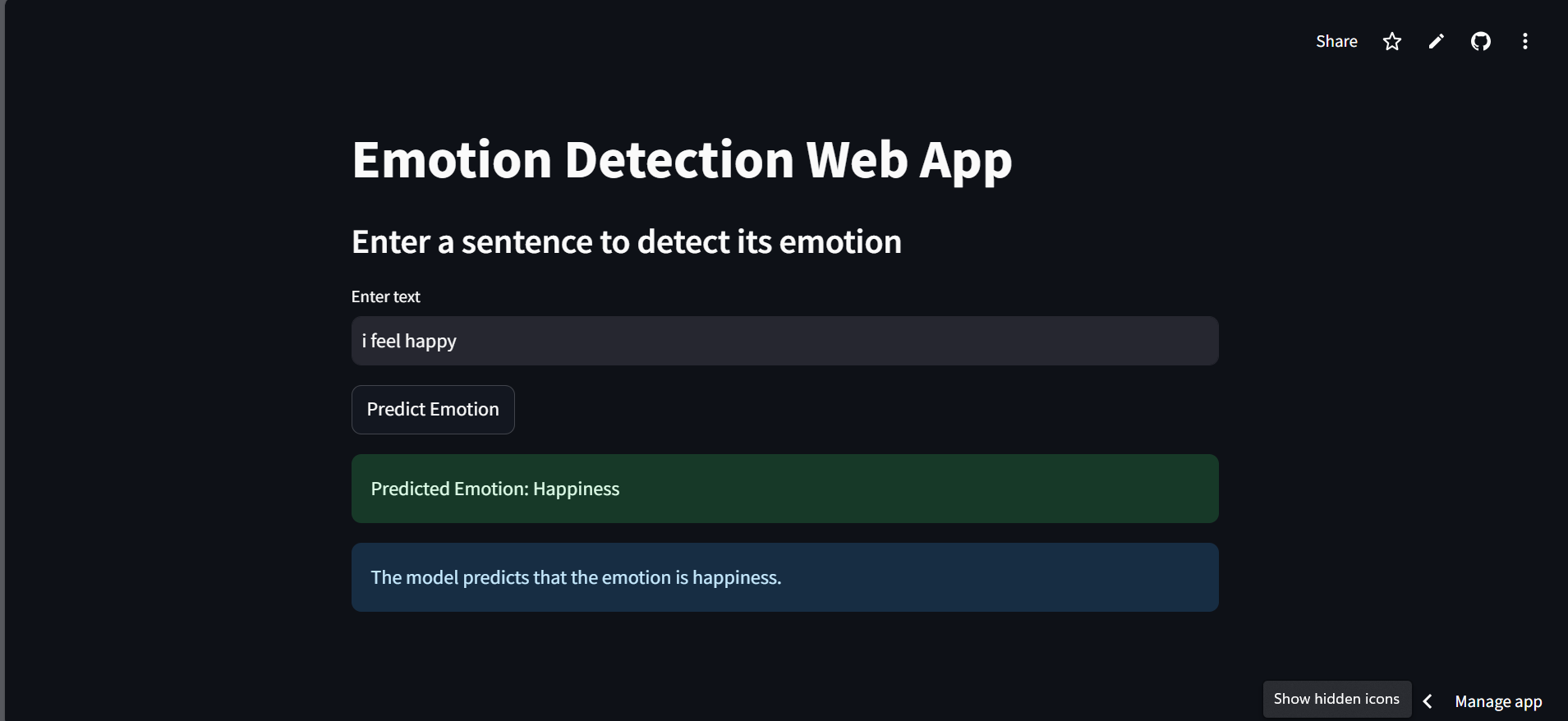






Matriks kebingungan dapat digunakan untuk analisis lebih lanjut, seperti melihat pola kesalahan di mana kelas *Fatigue* sering diprediksi sebagai *Sadness* atau *Neutral*.

# Hasil



Pada aplikasi deteksi emosi berbasis teks yang dibuat, pengguna dapat memasukkan sebuah kalimat yang mencerminkan perasaan mereka. Setelah pengguna memasukkan teks, seperti contoh "i feel happy", aplikasi memproses teks tersebut menggunakan model klasifikasi yang telah dilatih sebelumnya.

Setelah pemrosesan, model menghasilkan prediksi yang menunjukkan emosi yang terkandung dalam teks. Dalam contoh ini, model memprediksi emosi "Happiness" dengan akurasi tinggi, yang kemudian ditampilkan dalam tampilan web sebagai "Predicted Emotion: Happiness". Selain itu, aplikasi juga memberikan penjelasan tambahan, yaitu "The model predicts that the emotion is happiness", yang membantu pengguna untuk lebih memahami alasan di balik prediksi tersebut.

Hasil ini menunjukkan bahwa model klasifikasi yang digunakan dalam aplikasi mampu mendeteksi emosi dengan baik, khususnya untuk kalimat yang cukup eksplisit seperti "i feel happy". Proses prediksi ini didasarkan pada vektorisasi teks menggunakan teknik TF-IDF dan model regresi logistik yang telah dilatih dengan berbagai data teks berlabel. Meskipun demikian, penting untuk dicatat bahwa akurasi model dapat bervariasi tergantung pada kejelasan dan kompleksitas kalimat yang diberikan oleh pengguna. Dalam kasus kalimat yang ambigu atau tidak jelas dalam konteks emosional, model mungkin memerlukan penyesuaian atau fallback menggunakan teknik lain seperti cosine similarity untuk mencari teks yang paling mirip dalam dataset.

Secara keseluruhan, aplikasi ini memberikan hasil yang cukup memuaskan dalam mengidentifikasi emosi dari teks yang diberikan, dan dapat menjadi alat bantu yang berguna dalam analisis sentimen dan deteksi emosi berbasis teks.

# Kesimpulan

Proses pembuatan aplikasi deteksi emosi berbasis teks dimulai dengan pengumpulan dan pra-pemrosesan data. Data yang digunakan berisi teks yang mencerminkan berbagai emosi, yang kemudian diproses untuk menangani masalah ketidakseimbangan kelas dan variabilitas panjang teks. Kolom "text" yang berisi kalimat dengan emosi diekstraksi menjadi fitur numerik menggunakan teknik TF-IDF (Term Frequency-Inverse Document Frequency). Teknik ini memungkinkan model untuk menangkap pentingnya kata-kata dalam konteks setiap dokumen dengan memperhatikan distribusi kata-kata dalam seluruh dataset.

Selanjutnya, model yang digunakan dalam aplikasi ini adalah regresi logistik, yang dilatih menggunakan data teks yang sudah diekstraksi menjadi vektor dengan TF-IDF. GridSearchCV digunakan untuk menemukan kombinasi parameter model yang optimal, yang mencakup nilai C (strength of regularization), solver, dan jumlah iterasi maksimum. Untuk menangani ketidakseimbangan kelas, parameter class\_weight='balanced' diterapkan pada model. Setelah model terlatih, ia dievaluasi menggunakan data uji, menghasilkan metrik seperti akurasi dan laporan klasifikasi yang menunjukkan kinerja model dalam mengklasifikasikan emosi yang terkandung dalam teks.

Aplikasi web yang dibangun memungkinkan pengguna untuk memasukkan kalimat dan mendapatkan prediksi emosi secara real-time. Pengguna dapat melihat prediksi emosi yang dihasilkan oleh model, yang menunjukkan emosi seperti kebahagiaan, ketenangan, atau kesedihan berdasarkan teks yang diberikan. Aplikasi ini menunjukkan kinerja yang memadai dalam mendeteksi emosi pada kalimat yang jelas. Secara keseluruhan, pengembangan model dan aplikasi web ini berhasil menghasilkan sistem deteksi emosi berbasis teks yang efektif dan dapat diandalkan, meskipun perbaikan lebih lanjut diperlukan untuk menangani kasus-kasus ambigu dalam data teks.