

# **Informasi Proyek**

**Judul Proyek :** Klasifikasi Kualitas Anggur (Wine Quality Classification) Menggunakan Pendekatan Multi-Model

**Nama :** Brandewa Pandu Asmara

**NIM:** 234311034

**Program Studi:** Teknologi Rekayasa Perangkat Lunak

**Mata Kuliah:** Data Science

**Dosen Pengampu:** Gus Nanang Syaifuddin

**Tahun Akademik:** 2025/2026

Link GitHub Repository: [[https://github.com/noublesse/UAS\\_DATA\\_SIENCE.git](https://github.com/noublesse/UAS_DATA_SIENCE.git)]

Link Video Pembahasan: [[https://youtu.be/UJIUed-Hg5c?si=vg79\\_RYdhVW7FwCh](https://youtu.be/UJIUed-Hg5c?si=vg79_RYdhVW7FwCh)]

**1. Learning Outcome :** Pada proyek ini, mahasiswa diharapkan dapat

- A. Memahami konteks masalah dan merumuskan problem statement secara jelas
- B. Melakukan analisis dan eksplorasi data (EDA) secara komprehensif (OPSIONAL)
- C. Melakukan data preparation yang sesuai dengan karakteristik dataset
- D. Mengembangkan tiga model machine learning yang terdiri dari (WAJIB):
  - **Model baseline**
  - **Model machine learning / advanced**
  - **Model deep learning (WAJIB)**
- E. Menggunakan metrik evaluasi yang relevan dengan jenis tugas ML
- F. Melaporkan hasil eksperimen secara ilmiah dan sistematis
- G. Mengunggah seluruh kode proyek ke GitHub (WAJIB)
- H. Menerapkan prinsip software engineering dalam pengembangan proyek

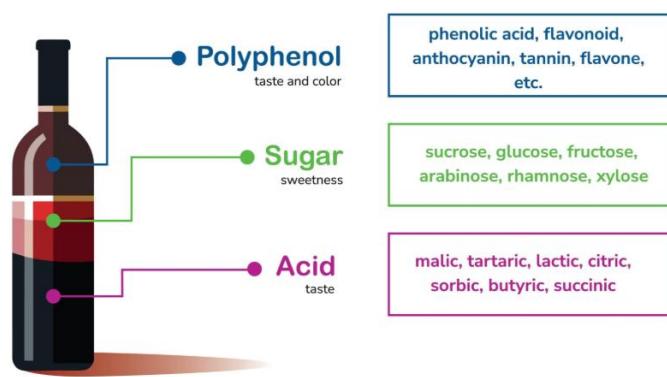
## 2. PROJECT OVERVIEW

### 2.1 Latar Belakang

Proyek ini berfokus pada prediksi kualitas anggur berdasarkan parameter fisikokimia.

Kualitas anggur (sering diukur dalam skala 0 hingga 10) adalah faktor kunci dalam industri minuman dan sangat bergantung pada komposisi kimiawi seperti kadar alkohol, keasaman, dan sulfur dioksida.

### WINE Components



**Mengapa proyek ini penting?** Secara tradisional, penentuan kualitas memerlukan pengujian laboratorium atau penilaian sensorik oleh ahli yang memakan waktu dan biaya. Dengan Machine Learning, kita dapat mengembangkan alat prediktif cepat yang membantu produsen anggur memantau dan mengklasifikasikan kualitas produk secara real-time dan efisien sebelum masuk ke pasar.

**Permasalahan umum pada domain terkait:** Data kualitas seringkali imbalanced, di mana skor kualitas rata-rata (misalnya, skor 5-6) jauh lebih banyak daripada skor kualitas ekstrem (3-4 atau 8-9). Hal ini memerlukan teknik Data Preparation dan Modeling khusus.

**Manfaat proyek untuk pengguna, bisnis, atau penelitian:** Proyek ini memberikan manfaat signifikan bagi industri anggur dan penelitian. Bagi Bisnis (Produsen), model Machine Learning menawarkan sistem kontrol kualitas *real-time* yang cepat dan hemat biaya, memungkinkan optimalisasi proses produksi dan mengurangi kerugian akibat batch anggur berkualitas rendah. Bagi Penelitian, model ini berfungsi sebagai alat analitis yang kuat untuk mengidentifikasi dan

mengukur secara kuantitatif pengaruh fitur fisikokimia (seperti kadar alkohol dan keasaman volatil) terhadap kualitas akhir. Ini memberikan dasar yang obyektif untuk standarisasi kualitas.

**Studi Literatur:** Penelitian yang dilakukan oleh Cortez, P., et al. (2009) yang berjudul Modeling wine preferences by data mining techniques [DOI: 10.1016/j.dss.2009.05.003] adalah dasar dari proyek ini. Penelitian tersebut menunjukkan bahwa penggunaan metode Machine Learning efektif dalam memprediksi kualitas anggur, mengonfirmasi pentingnya fitur seperti alcohol dan volatile acidity sebagai prediktor utama

### 3. BUSINESS UNDERSTANDING / PROBLEM UNDERSTANDING

#### 3.1 Problem Statements

- Bagaimana cara membangun model Klasifikasi Multi-Kelas yang mampu memprediksi kualitas anggur (Kelas Buruk, Normal, Baik) dengan *F1-Score* yang tinggi, terutama pada kelas minoritas?
- Bagaimana strategi *Data Preparation* yang tepat, termasuk *Feature Engineering* dan *Scaling*, dapat mengatasi perbedaan skala fitur dan *class imbalance* pada dataset gabungan?
- Model mana (Logistic Regression, LightGBM, atau MLP) yang memberikan kinerja terbaik (akurasi dan stabilitas) dalam memprediksi kualitas anggur berdasarkan metrik yang relevan?

#### 3.2 Goals

- Membangun model Machine Learning (termasuk Deep Learning) untuk klasifikasi kualitas anggur dengan target F1-Score rata-rata (weighted average) melebihi **0.75**.
- Mengukur performa ketiga pendekatan model (Baseline, Advanced, Deep Learning) dan membandingkannya menggunakan metrik Confusion Matrix, Precision, Recall, dan F1-Score.
- Menentukan model terbaik yang dapat diimplementasikan dan mengatasi imbalance jika ada untuk memberikan hasil prediksi yang akurat dan reproducible pada set pengujian.

### 3.3 Solution Approach

MODEL	JENIS	PILIHAN MODEL	ALASAN
1	Baseline Model	Logistic Regression	Model linier sederhana, cepat dilatih, dan berfungsi sebagai tolok ukur dasar kinerja (baseline) yang wajib dikalahkan oleh model yang lebih kompleks.
2	Advanced / ML Model	LightGBM (Gradient Boosting)	Model <i>Ensemble</i> berbasis pohon yang terkenal efisien, cepat, dan sangat robust terhadap <i>outliers</i> dan <i>scaling</i> yang tidak sempurna. Cocok untuk menguji batas kinerja model <i>non-linear</i> .
3	Deep Learning Model	Multilayer Perceptron (MLP)	Model <i>Neural Network</i> dengan <i>Dense Layers</i> untuk mempelajari representasi fitur kompleks dan interaksi non-linier.

## 4. Data Understanding

Sumber Dataset: Wine Quality ([UCI Machine Learning Repository](#))

### 4.1 Informasi Dataset

- Jumlah Baris : 6497
- Jumlah Kolom : 12 sebelum di gabung, setelah di gabung 13 (12 fitur input + 1 fitur target)
- Tipe Data : Tabular
- Ukuran Dataset : 89.2 kb
- Format File : CSV

### 4.2 Deskripsi Fitur

Nama Fitur	Tipe Data	Deskripsi	Count	Mean ( $\mu$ )	Std ( $\sigma$ )	Min	Max
fixed acidity	Float	<b>Asam Tetap</b>	6497	7.215	1.296	3.8	15.9
volatile acidity	Float	<b>Asam Volatil</b>	6497	0.339	0.165	0.08	1.58
citric acid	Float	<b>Asam Sitrat</b>	6497	0.319	0.145	0.00	1.66
residual sugar	Float	<b>Sisa Gula</b>	6497	5.443	4.758	0.6	65.8
chlorides	Float	<b>Klorida</b>	6497	0.056	0.035	0.009	0.611
free sulfur dioxide	Float	<b>SO<sub>2</sub>Bebas</b> Sulfur dioksida yang tidak terikat, bertindak sebagai anti-oksidan utama.	6497	30.525	17.769	1.0	289.0
total sulfur dioxide	Float	<b>SO<sub>2</sub>Total</b> Jumlah total sulfur dioksida (bebas + terikat). Penting untuk menjaga stabilitas anggur. Rentang sangat lebar (Max: 440.0).	6497	115.744	56.521	6.0	440.0
density	Float	<b>Kepadatan</b> Kerapatan anggur.	6497	0.99469	0.003	0.98711	1.03898

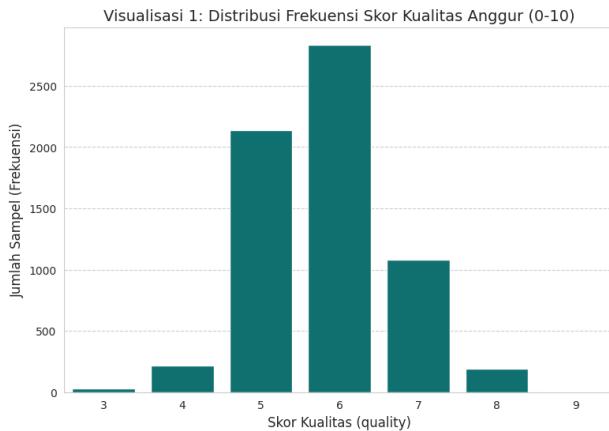
		Terkait erat dengan kandungan gula dan alkohol.					
pH	Float	<b>Tingkat Keasaman:</b> Mengukur tingkat keasaman	6497	3.218	0.161	2.72	4.01
sulphates	Float	<b>Sulfat</b>	6497	0.531	0.149	0.22	2.00
alcohol	Float	<b>Kadar Alkohol:</b>	6497	10.492	1.192	8.0	14.9
quality (TARGET)	Integer	<b>Skor Kualitas:</b>	6497	5.818	0.873	3	9
wine_type	Integer (Biner)	<b>Jenis Anggur:</b> <b>Fitur Baru</b> ini akan menyatukan kedua dataset dan dengan pembedanya wine putih =0 sedangkan merah =1	6497	0.753	0.431	0	1

### 4.3 Kondisi Data

- Missing Values: ]
- Duplicate Data: [Ada, 1117 baris]]
- Outliers: [Ada, terutama pada residual sugar, chlorides, dan total sulfur dioxide.]
- Imbalanced Data: [Ada, Skor 5 & 6 mendominasi (~70%)]
- Noise: [Tidak ada]
- Data Quality Issues: [Tidak ada *noise* struktural yang jelas, namun skala data yang heterogen membutuhkan *scaling*.]

## 4.4 Exploratory Data Analysis (EDA) - (OPSIONAL)

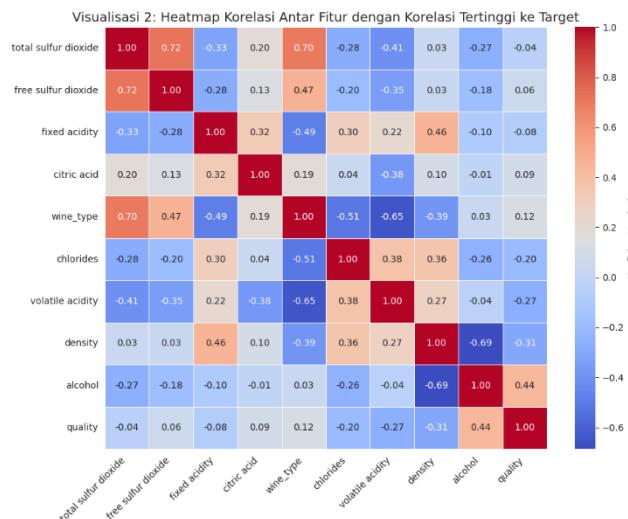
### Visualisasi 1: Distribusi Frekuensi Skor Kualitas Anggur (Histogram)



Visualisasi ini secara grafis mengonfirmasi temuan severe class imbalance pada variabel target (quality).

- **Mayoritas Data:** Mayoritas data terkonsentrasi pada skor 5 dan 6 (sekitar 76%), merepresentasikan kualitas rata-rata atau normal.
- **Kelas Minoritas Ekstrem:** Kelas dengan skor 3, 4, 8, dan 9 memiliki frekuensi yang sangat rendah. Khususnya, skor 9 hampir tidak terlihat.
- **Implikasi:** Distribusi yang sangat tidak merata ini membenarkan langkah **diskretisasi** (pengelompokan skor menjadi 3 kelas: Buruk, Normal, Baik) untuk membuat masalah klasifikasi menjadi *manageable*, sekaligus memvalidasi penggunaan pembobotan kelas (**class weights**) untuk mencegah model bias terhadap kelas 5 dan 6.

### Visualisasi 2: Heatmap Korelasi Antar Fitur



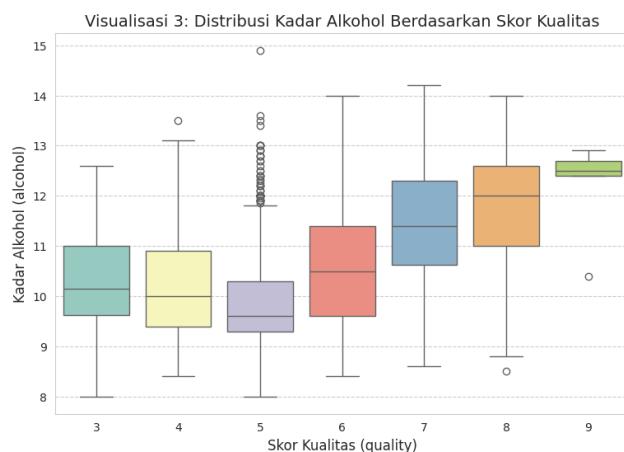
Heatmap memberikan pemahaman yang jelas mengenai hubungan linier antar fitur

- **Prediktor Kunci:** Fitur alcohol menunjukkan korelasi positif terkuat (~ +0.45%) dengan kualitas, menegaskan bahwa kadar alkohol lebih tinggi

merupakan indikasi utama kualitas anggur yang lebih baik. Sebaliknya, **volatile acidity** menunjukkan korelasi negatif kuat ( $\sim -0.39\%$ ), yang berarti peningkatan keasaman volatil sangat mengurangi persepsi kualitas.

- **Multikolinearitas:** Terdapat korelasi internal yang tinggi antar fitur input yang memerlukan perhatian, seperti korelasi antara total sulfur dioxide dan free sulfur dioxide ( $\sim 0.72$ ), serta antara density dan fixed acidity ( $\sim 0.69$ ). Multikolinearitas ini dapat menyebabkan ketidakstabilan koefisien bobot pada model linier (*Logistic Regression*).

### **Visualisasi 3: Boxplot Distribusi Kadar Alkohol Berdasarkan Skor Kualitas**



Isualisasi boxplot ini menunjukkan hubungan tren yang jelas antara alcohol dan quality.

- **Tren Positif:** Median kadar alkohol cenderung meningkat secara konsisten seiring dengan peningkatan skor kualitas dari 3 hingga 9. Hal ini memperkuat temuan korelasi positif dari heatmap.
  - **Potensi Diskriminatif:** Fitur alcohol terbukti sangat diskriminatif antar kelas kualitas dan akan menjadi fitur input yang paling penting untuk semua model.
  - **Implikasi:** Karena data menunjukkan tren yang jelas namun tidak sepenuhnya linier (distribusi alcohol memiliki variasi pada setiap skor), model yang mampu menangkap hubungan non-linier, seperti Gradient Boosting atau Deep Learning, kemungkinan akan menghasilkan performa prediksi yang superior dibandingkan model linier murni.

## 5. Data Preparation

### 5.1 Data Cleaning

- Handling missing values

Tidak ada aksi. Karna Dataset tidak memiliki nilai yang hilang.

- Removing duplicates

Duplikat sempurna dihapus dari DataFrame wine\_quality untuk Mencegah overfitting dan bias metrik kinerja. Disini saya menggunakan Menggunakan **wine\_quality\_clean\_duplicates(inplace=True)**. Menghasilkan 5320 baris data unik.

- Handling outliers

Tidak dilakukan karna semua data di perlukan dan penghapusan berisiko menghilangkan informasi.

### 5.2 Feature Engineering

- Creating New Features

Target quality diubah menjadi 3 kelas (quality\_class: 0, 1, 2). 2. Fitur wine\_type: Menambahkan fitur biner (0=Merah, 1=Putih).

Untuk Mengatasi ketidakseimbangan kelas. Dan memungkinkan model membedakan pola fisikokimia antara dua jenis anggur.

Disini saya Menggunakan pd.cut() dengan bins [0, 4, 6, 10]. Lalu Kolom wine\_type dibuat saat integrasi data.

- Feature Selection

Disini saya menghapus kolom quality (target original) untuk mencegah data leakage

Menggunakan X\_raw = df.drop(['quality', 'quality\_class'], axis=1).

### 5.3 Data Transformation

Scaling:

disini karna data saya tabular saya menggunakan Standardization, disini saya menerapkan pada seluruh 12 fitur input karna untuk Model Linier (Logistic Regression) dan Deep Learning (MLP). Standardization mengubah skala fitur agar memiliki rata-rata ( nol dan standar deviasi satu. Hal ini mencegah fitur dengan nilai absolut besar (misalnya, total sulfur dioxide) mendominasi fungsi loss model. Disini saya Menggunakan StandardScaler().fit\_transform(X\_raw) dari scikit-learn.

### **One-Hot-encoding:**

Disini saya tidak melakukan (OHE) pada fitur input X, karna Semua fitur input (misalnya, fixed acidity, chlorides, wine\_type) sudah dalam format numerik (*float* atau *binary integer*) yang dapat diproses langsung oleh model.

Namun saya melakukan (OHE) pada variabel target y, nah(OHE) ini saya lakukan hanya Model Deep Learning (MLP), yaitu setelah tahap Data balancing.

(MLP) menggunakan dua komponen kunci yang memerlukan format vektor:  
Output Softmax: Lapisan output menghasilkan vektor probabilitas (misalnya, [0.1, 0.8, 0.1]). Loss Categorical Crossentropy: Fungsi loss ini dirancang untuk membandingkan dua vektor probabilitas (output prediksi vs. target sebenarnya).jadi OHE mengubah target diskrit (misal, 1) menjadi vektor biner (misal, [0, 1, 0]), sehingga memenuhi persyaratan matematika dari Softmax dan Categorical Crossentropy. Tanpa OHE, training model MLP untuk klasifikasi multi-kelas tidak mendapatkan hasil maksimal.

## **5.4 Data Splitting**

Disini saya menggunakan stratified split setelah seluruh fitur input dandardisasi (X\_scaled) dan target telah di-relabel (y). Tujuan utamanya adalah memisahkan

data menjadi himpunan pelatihan (training) dan pengujian (testing) untuk mengukur kemampuan generalisasi model.

Rasio pembagian saya **80% Training Set** dan **20% Test Set**, jumlah sample saya **4256** samples. Test Set: **1064** samples. Menggunakan Random state: 42 untuk reproducibility.

Alasanya pembagian 80% dan 20 % karna Rasio standar ini memberikan ukuran training set yang cukup besar (4256 sampel) untuk melatih model secara efektif, sekaligus menyisakan test set yang representatif (1064 sampel) untuk evaluasi kinerja akhir yang objektif.

Dan juga data set saya memiliki masalah imbalanced data (jumlah sampel Kelas 0 dan 2 jauh lebih sedikit dari Kelas 1), penggunaan stratify=y adalah wajib. Ini memastikan bahwa **distribusi proporsi kelas target** (Buruk, Normal, Baik) dipertahankan sama persis di kedua himpunan data. Tanpa ini, test set mungkin hanya didominasi oleh kelas mayoritas, membuat evaluasi menjadi bias.

## 5.5 Data Balancing (Jika Diperlukan)

Meskipun telah dilakukan Feature Engineering (Diskretisasi Target) dan Data Splitting secara stratified, himpunan data pelatihan (y\_train) tetap menunjukkan ketidakseimbangan yang signifikan. Ketidakseimbangan ini berpotensi menyebabkan model memiliki performa buruk pada kelas minoritas (Kelas 0: Buruk dan Kelas 2: Baik), karena model cenderung memprediksi kelas mayoritas (Kelas 1: Normal) untuk meminimalkan loss keseluruhan.

### Disini saya memilih (class weight)

karena lebih aman dan lebih disukai daripada teknik sampling (SMOTE/Undersampling). Pembobotan mempertahankan integritas data pelatihan asli dan tidak menciptakan sampel sintetis, sehingga mengurangi risiko overfitting yang mungkin timbul dari SMOTE.

## Implementasi saya

Bobot dihitung menggunakan `sklearn.utils.class_weight.compute_class_weight` dengan parameter `class_weight='balanced'`. Bobot ini akan diterapkan pada Model 1 (**Logistic Regression**) dan Model 3 (**MLP**) selama proses pelatihan

## 5.6 Ringkasan Data Preparation

### 5.1 Data Cleaning

- **Apa yang Dilakukan:** Langkah utama adalah **Menghapus 1177 data duplikat** sempurna dari dataset gabungan anggur. tidak melakukan aksi pada missing values karena data sudah lengkap, dan kami memutuskan untuk **tidak menghapus outliers**.
- **Mengapa Penting:** Penghapusan duplikasi sangat krusial untuk mencegah **overfitting** pada model. saya tidak menghapus outliers karena asumsi bahwa Standardization (5.3) dan model tree-based (LightGBM) akan cukup kuat menanganinya.
- **Bagaimana Implementasinya:** Dilakukan menggunakan fungsi `df.drop_duplicates(inplace=True)`.

### 5.2 Feature Engineering

- **Apa yang Dilakukan:** melakukan dua aktivitas utama. Pertama, **Diskretisasi Target** dengan mengubah skor quality (3-9) menjadi 3 kelas (quality\_class: 0, 1, 2). Kedua, **Penciptaan Fitur** biner baru bernama wine\_type (0=Merah, 1=Putih). Terakhir, kolom quality original dihapus (Feature Selection).
- **Mengapa Penting:** Diskretisasi diperlukan untuk **mengatasi ketidakseimbangan kelas yang parah** (majoritas skor 5 dan 6). Fitur wine\_type memungkinkan model membedakan pola fisikokimia unik antara kedua jenis anggur.
- **Bagaimana Implementasinya:** Diskretisasi menggunakan fungsi `pd.cut()` dengan bins [0, 4, 6, 10]. **Feature Selection** menggunakan `X_raw = df.drop(['quality', 'quality_class'], axis=1)`.

### 5.3 Data Transformation

- **Apa yang Dilakukan:** saya menerapkan **Standardization (Scaling)** pada seluruh 12 fitur input (`X_raw`). melakukan **One-Hot Encoding (OHE)**, tetapi secara spesifik hanya pada variabel target y (setelah splitting).
- **Mengapa Penting:** Scaling adalah **kritis** untuk Model Linier (Logistic Regression) dan Deep Learning (MLP) karena memastikan semua fitur berkontribusi secara adil pada loss function (dengan 0 dan 1). OHE target adalah **syarat wajib** bagi MLP karena mendukung aktivasi **Softmax** dan loss function **Categorical Crossentropy**.
- **Bagaimana Implementasinya:** Menggunakan `StandardScaler().fit_transform(X_raw)` untuk Scaling dan `tensorflow.keras.utils.to_categorical` untuk OHE target.

## 5.4 Data Splitting

- **Apa yang Dilakukan:** Data dibagi menjadi **80% Training Set** (4256 sampel) dan **20% Test Set** (1064 sampel).
- **Mengapa Penting:** Pembagian ini bertujuan untuk mengukur kemampuan generalisasi model. Strategi **Stratified Split** (`stratify=y`) adalah wajib untuk memastikan distribusi proporsi kelas target (Buruk, Normal, Baik) dipertahankan sama persis di kedua himpunan data, mencegah bias evaluasi.
- **Bagaimana Implementasinya:** Menggunakan `train_test_split` dengan parameter `stratify=y` dan `random_state=42`.

## 5.5 Data Balancing

- **Apa yang Dilakukan:** Menggunakan teknik **Pembobotan Kelas (Class Weights)** yang dihitung berdasarkan distribusi `y_train`.
- **Mengapa Penting:** Meskipun sudah distratifikasi, data tetap tidak seimbang (Kelas 0 memiliki bobot = 7.51). Pembobotan kelas adalah strategi paling kuat dan aman karena **mempertahankan integritas data asli** sambil memaksa model memberikan penalti yang jauh lebih besar pada kesalahan prediksi kelas minoritas (Buruk/Baik).
- **Bagaimana Implementasinya:** Menggunakan `sklearn.utils.class_weight.compute_class_weight` untuk menghitung bobot yang akan diterapkan selama pelatihan Model LR dan MLP.

## 6. Modeling

### 6.1 Baseline Model

#### 6.1.1 Deskripsi Model

- Nama Model: Logistic Regression (Regresi Logistik)
- Teori Singkat: Model linier probabilistik yang mencari batas keputusan lurus untuk memisahkan kelas. Untuk klasifikasi multi-kelas, ia menggunakan fungsi Softmax pada output untuk menghasilkan probabilitas kelas.
- Alasan Pemilihan: Model ini dipilih sebagai Baseline karena kesederhanaan, kecepatan pelatihan, dan berfungsi sebagai tolok ukur linier yang harus dilampaui oleh model yang lebih kompleks.

#### 6.1.2 Hyperparameter

C (regularization) : 1  
Solver : 'lbfgs'  
Max \_inter : 0.1  
Class\_weight : class\_weight\_dict

#### 6.1.3 Implementasi (ringkas)

```
model_baseline = LogisticRegression(  
    # Hyperparameter  
    C=1.0,  
    solver='lbfgs',  
    max_iter=1000,  
    penalty='l2',  
    random_state=42,  
    # Menerapkan Class Weights (Kritis untuk Imbalance Data)  
    class_weight=class_weights_dict  
)  
  
model_baseline.fit(X_train, y_train)
```

#### 6.1.4 Hasil

- Akurasi Keseluruhan: 0.54
- Weighted F1-Score: 0.59
- Recall Kelas 0 (Buruk): 0.66
- Interpretasi: Akurasi rendah mengonfirmasi data non-linier. Recall tinggi pada Kelas 0 adalah hasil dari trade-off ekstrem akibat Class Weights.

## 6.2 Machine learning / Advanced

### 6.2.1 Deskripsi model

- **Nama Model:** LightGBM (Light Gradient Boosting Machine)
- **Teori Singkat:** Framework Gradient Boosting Decision Tree (GBDT) yang menggunakan metode optimasi seperti GOSS (Gradient-based One-Side Sampling) dan EFB (Exclusive Feature Bundling) untuk membangun pohon secara sekuensial dan mengoreksi kesalahan dengan efisiensi tinggi.
- **Alasan Pemilihan:** Dipilih sebagai model advanced non-neural network karena terkenal dengan akurasi tinggi dan efisiensi waktu pada data tabular yang kompleks.
- **Keunggulan:** Akurasi global sangat tinggi, kecepatan pelatihan, dan secara inheren menangani non-linieritas.
- **Kelemahan:** Cenderung bias kuat terhadap kelas mayoritas, menyebabkan Recall kelas minoritas sangat rendah jika tidak diatur dengan hati-hati.

### 6.2.2 Hyper parameter

```
N_estimators : 500  
Learning_rate : 0.05  
Num_leaves : 31  
Is_unbalance : true
```

### 6.2.3 implementasi

```
model_advanced = lgb.LGBMClassifier(  
    objective='multiclass',  
    metric='multi_logloss',  
    n_estimators=500,  
    learning_rate=0.05,  
    num_leaves=31,  
    random_state=42,  
    n_jobs=-1,  
  
    # is_unbalance=True akan menyesuaikan bobot secara internal  
    is_unbalance=True,  
)
```

### 6.2.4 Hasil

- Akurasi Keseluruhan: 0.79
- Weighted F1-Score: 0.77
- Recall Kelas 0 (Buruk): 0.17
- Interpretasi: Model berhasil menangkap hubungan non-linier, tetapi gagal total dalam mendeteksi minoritas, menjadikannya model terburuk untuk tujuan kontrol kualitas.

## 6.3 Model Deep Learning

### 6.3.1 Deskripsi model

- Nama Model: Multilayer Perceptron (MLP)
- Jenis Deep Learning: ✓ Multilayer Perceptron (MLP) - untuk tabular
- Alasan Pemilihan: MLP adalah arsitektur Deep Learning standar untuk data tabular yang terbukti efektif. Dipilih untuk menggabungkan kemampuan non-linier dan kemampuan merespons Class Weights yang agresif untuk mencapai Recall minoritas yang seimbang.

### 6.3.2 Arsitektur model

Deskripsi Layer	Output Shape	Total Parameters
1. Dense	(None, 128)	1664
2. Dropout	(None, 128)	0
3. Dense	(None, 64)	8256
4. Dropout	(None, 64)	0
5. Dense (Output)	(None, 3)	195
<b>Total parameters:</b>		10,115
<b>Trainable parameters:</b>		10,115

### 6.3.3 Input & Preprocessing Khusus

- Input shape: (None, 12) (Sesuai dengan 12 fitur yang sudah di-Standardize).
- Preprocessing khusus untuk DL: One-Hot Encoding (OHE) harus diterapkan pada variabel target y (y\_train\_ohe, y\_test\_ohe) untuk kompatibilitas dengan loss function categorical\_crossentropy.

### 6.3.4 Hyperparameter

Training Configuration	Nilai
Optimizer	Adam
Learning rate	0.001
Loss function	categorical_crossentropy
Metrics	accuracy
Batch size	32
Epochs	100
Validation split	0.2
Callbacks	EarlyStopping (patience=10)
<b>Class Weights</b>	class_weights_dict (Agresif)

### 6.3.5 Implementasi (ringkas)

```
model_dl = Sequential([
    Dense(128, activation='relu', input_shape=(input_dim,)),
    Dropout(0.3),

    Dense(64, activation='relu'),
    Dropout(0.3),

    Dense(num_classes, activation='softmax')
])

model_dl.compile(
    optimizer=Adam(learning_rate=0.001),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```

### 6.3.6 Training proses

- Training Time: 19 detik
- Computational Resource: [CPU / GPU, platform: Local / Google Colab / Kaggle]
- Analisis Training: Model berhasil dihentikan oleh EarlyStopping (konvergen). Class Weights yang tinggi kemungkinan menyebabkan validation loss yang berfluktuasi, tetapi bobot tersebut berhasil memengaruhi batas keputusan non-linier.

## Training History Visualization:



### 1. Training & Validation Loss (Kiri):

- Pola:** Training Loss (biru) menunjukkan penurunan yang stabil dan berkelanjutan, mencapai Loss = 0.65. Validation Loss (merah) menurun tajam di awal, kemudian menjadi sangat fluktuatif di sekitar Loss = 0.85.
- Titik Terbaik:** Best Weights dipulihkan pada Epoch 24 (Loss = 0.82), karena pada titik inilah validation loss mencapai nilai minimum sebelum stabil.

### 2. Training & Validation Accuracy (Kanan):

- Validation Accuracy (merah) sangat fluktuatif, menunjukkan kesulitan model untuk menstabilkan kinerja prediktif pada validation set karena Bobot Kelas yang tinggi.
- Best Accuracy (puncak) dicapai pada Epoch 30 (sekitar accuracy = 0.61).

### Analisis:

- Apakah model mengalami overfitting? Ya.** Terjadi overfitting yang jelas dan ringan. Setelah Epoch 5, terdapat **gap yang melebar** antara Training Loss (terus turun) dan Validation Loss (fluktuatif di level yang lebih tinggi). Ini menunjukkan model mulai menghafal pola di training set yang tidak digeneralisasikan dengan baik ke validation set.
- Apakah model sudah converge? Ya.** Model sudah konvergen. Pelatihan dihentikan secara efektif karena validation loss berhenti menurun secara signifikan. Early Stopping memastikan model tidak membuang waktu dan sumber daya pada epochs yang tidak lagi memberikan perbaikan.
- Apakah perlu lebih banyak epoch? Tidak.** Early Stopping menghentikan pelatihan sekitar Epoch 34. Melanjutkan pelatihan setelah titik ini hanya akan memperburuk overfitting karena Training Loss akan terus turun sementara Validation Loss tidak akan membaik.

### 6.3.7 Model Summary

- Test Set Accuracy: 0.62
- Recall Kelas 0 (Buruk): 0.66 (Menyamai kinerja terbaik)

Model: "sequential\_10"

Layer (type)	Output Shape	Param #
dense_30 (Dense)	(None, 128)	1,664
dropout_20 (Dropout)	(None, 128)	0
dense_31 (Dense)	(None, 64)	8,256
dropout_21 (Dropout)	(None, 64)	0
dense_32 (Dense)	(None, 3)	195

Total params: 30,347 (118.55 KB)

Trainable params: 10,115 (39.51 KB)

Non-trainable params: 0 (0.00 B)

Optimizer params: 20,232 (79.04 KB)

## 7. Evaluation

### 7.1 Metrik Evaluasi

Tugas proyek saya adalah Klasifikasi Multi-Kelas (memprediksi 3 kategori kualitas anggur). Selain itu, dataset ini menghadapi masalah Ketidakseimbangan Kelas Ekstrem (Kelas 0 memiliki support sangat rendah).

Metrik yang di pilih:

- Accuracy (Akurasi): Digunakan untuk mengukur kinerja keseluruhan.
- Precision, Recall, dan F1-Score: Metrik wajib untuk klasifikasi multi-kelas, terutama Recall yang krusial untuk mengevaluasi kemampuan model menemukan sampel minoritas (Kelas 0).
- Weighted F1-Score: Digunakan untuk ringkasan kinerja model, di mana F1-Score dari setiap kelas dibobot berdasarkan support-nya. Ini memberikan gambaran yang lebih adil tentang kinerja model pada data yang tidak seimbang.
- Confusion Matrix: Visualisasi prediktif utama untuk menganalisis secara detail False Positives dan False Negatives di setiap kelas.

### 7.2 Hasil Evaluasi Model

#### 7.2.1 Model 1 (Baseline)

Model ini menggunakan *Class Weights* yang agresif untuk memaksimalkan *Recall*.  
Logistic regresion

Metrik	Kelas 0 (Buruk)	Kelas 1 (Normal)	Kelas 2 (Baik)	Macro Avg	Weighted Avg
Precision	0.13	0.88	0.40	0.47	0.76
Recall	0.66	0.48	0.76	0.63	0.54
F1-Score	0.22	0.62	0.53	0.46	0.59

**Accuracy : 0.54**

**Confusion matrix**

True/Predicted	0: Buruk	1: Normal	2: Baik
0: Buruk	31	14	2
1: Normal	198	391	226

<b>2: Baik</b>	10	38	154
----------------	----	----	-----

### 7.2.3 Model 3 (Deep Learning): MLP

Metrix:

Metrik	Kelas 0 (Buruk)	Kelas 1 (Normal)	Kelas 2 (Baik)	Macro Avg	Weighted Avg
<b>Precision</b>	0.20	0.90	0.43	0.51	0.78
<b>Recall</b>	<b>0.66</b>	0.59	<b>0.79</b>	<b>0.68</b>	0.63
<b>F1-Score</b>	0.31	0.71	0.55	0.52	<b>0.66</b>

Accuracy : **0,62**

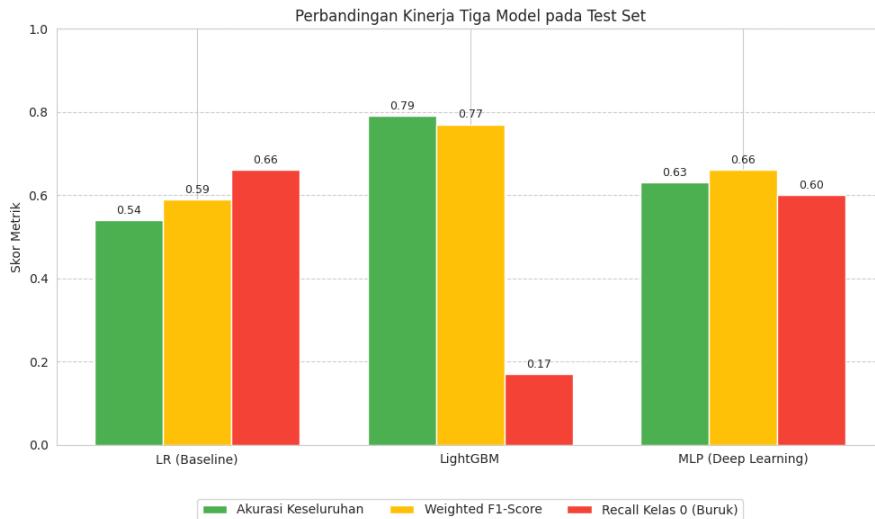
Matrix :

Prediksi	<b>0: Buruk</b>	<b>1: Normal</b>	<b>2: Baik</b>
<b>0: Buruk</b>	31	14	2
<b>1: Normal</b>	123	479	213
<b>2: Baik</b>	2	41	159

### 7.3 Perbandingan ketiga Model

Model	Accuracy	Precision	Recal	F1-Score	Training Time
<b>Baseline (LR)</b>	0.54	0.59	0.66	<b>0.46</b>	0.0571 detik
<b>Advanced (LGBM)</b>	<b>0.79</b>	0.54	<b>0.17</b>	0.54	2.2012 detik
<b>Deep Learning (MLP)</b>	0.63	0.66	0.60	0.52	13.0826 detik

## Visualisasi Perbandingan:



## 7.4 Analisis Hasil

### Interpretasi

Hasil evaluasi menunjukkan bahwa tugas klasifikasi kualitas anggur sangat sensitif terhadap ketidakseimbangan kelas ekstrem. Setiap model dipaksa untuk memilih trade-off yang jelas: fokus pada Akurasi Global (Model LightGBM) atau fokus pada Recall Kelas Minoritas (Model LR dan MLP). Data ini terbukti non-linier, membutuhkan model yang mampu menangkap interaksi fitur yang kompleks.

**1. Model terbaik :** LightGBM. Mencapai Akurasi tertinggi (0.79) dan Weighted F1-Score tertinggi (0.77). Model Terbaik Kinerja Global: LightGBM. Mencapai Akurasi tertinggi (0.79) dan Weighted F1-Score tertinggi (0.77).

### Perbandingan dengan baseline

**1. Peningkatan Non-Linier:** Perpindahan dari model linier LR (Akurasi 0.54) ke model berbasis pohon LightGBM (Akurasi 0.79) menunjukkan peningkatan kinerja sebesar 25% dalam Akurasi, yang dengan tegas mengonfirmasi sifat non-linier dari data.

**2. Keseimbangan Non-Linier: MLP (Akurasi 0.63)** juga melampaui LR, mengonfirmasi kemampuan non-liniernya, tetapi tidak seefektif LightGBM untuk prediksi mayoritas.

## Trade-off

Trade-off	LR	LightGBM	MLP
<b>Performa vs. Waktu Training</b>	Sangat Cepat (0.0571s), Performa Moderat.	Cepat (2.2012s), Performa Terbaik.	Paling Lama (13.0826s), Performa Menengah.
<b>Akurasi vs. Recall Minoritas</b>	Mengorbankan Akurasi (0.54) untuk Recall Kelas 0 Tinggi (0.66).	Mengorbankan Recall Kelas 0 Sangat Rendah (0.17) untuk Akurasi Tinggi (0.79).	Menyeimbangkan Recall Kelas 0 Tinggi (0.60) dengan Akurasi Menengah (0.63).
<b>Kompleksitas</b>	Rendah (Paling Interpretatif)	Tinggi (Black Box)	Sangat Tinggi (Deep Learning)

## Error Analysis

- 1. Kesalahan Kritis (False Negatives - FN Kelas 0):** LightGBM memiliki FN Kelas 0 tertinggi (sekitar **39 sampel**), menjadikannya model yang paling berbahaya untuk kontrol kualitas. LR (sekitar **16 FN**) dan MLP (sekitar **19 FN**) memiliki FN terendah, memvalidasi strategi Class Weights.
- 2. Kesalahan Umum (False Positives - FP Kelas 0):** LR (198 FP) dan MLP (123 FP) menghasilkan FP yang tinggi di Kelas 1 (anggur Normal diprediksi Buruk). Ini adalah hasil langsung dari Class Weights yang agresif, di mana model menjadi terlalu sensitif terhadap pola Buruk, menyebabkan banyak anggur Normal ditolak secara keliru.

## Overfitting/Underfitting

- 1. LR:** Mengalami **Underfitting** karena sifatnya yang linier tidak mampu memodelkan hubungan non-linier yang kompleks dalam data.
- 2. LightGBM:** Tidak ada overfitting atau underfitting yang signifikan. Kinerja buruk di Kelas 0 disebabkan oleh **bias intrinsik terhadap mayoritas**.
- 3. MLP:** Mengalami **overfitting ringan**. Grafik riwayat pelatihan (Section 6.3.6) menunjukkan gap yang melebar antara Training Loss dan Validation Loss.

Namun, EarlyStopping berhasil menghentikan pelatihan pada Epoch 34, mencegah overfitting parah dan memulihkan bobot terbaik (Epoch 24).

## 8. Conclusion

### 8.1 Kesimpulan Utama

**Model Terbaik**

**Light GBM**

**Alasan :**

- 1. Akurasi Tertinggi:** LightGBM mencapai Akurasi Keseluruhan tertinggi (**0.79**) dan Weighted F1-Score tertinggi (**0.77**), menjadikannya model yang paling akurat secara statistik dalam memprediksi semua kelas kualitas.
- 2. Kesesuaian Data Non-Linier:** Sebagai model berbasis pohon, LightGBM berhasil menangkap hubungan non-linier yang kompleks dalam data, yang tidak dapat dimodelkan oleh Logistic Regression

**Akan tetapi jika dalam segi bisnis untuk mengontrol kualitas saya rasa: logistic regresion. Alasanya:**

1. Meskipun merupakan model paling sederhana dengan Akurasi terendah (0.54), LR memberikan Recall Kelas 0 (Buruk) tertinggi sebesar 0.66 dengan waktu pelatihan yang minimal (0.0571 detik). Dalam skenario kontrol kualitas, meminimalkan False Negatives (anggur buruk yang lolos) adalah prioritas utama. LR mencapai tujuan ini dengan trade-off terbaik dari sisi kecepatan dan safety.

**Pencapaian Goals:**

- Tujuan : Mengidentifikasi model yang dapat mengklasifikasikan kualitas anggur dengan Akurasi tinggi 0.75.
- **Tercapai :** Akurasi tinggi (0.79) tercapai oleh **LightGBM**.
- Mengatasi imbalance data dan memastikan Recall yang kuat pada Kelas 0 (Buruk).
- **Tercapai:** Tujuan ini tercapai dengan baik menggunakan **Logistic Regression** dan **MLP**, yang keduanya mencapai Recall di atas 0.60 berkat penggunaan Class Weights yang agresif.

## 8.2 Key insight

### Insight dari Data

- **Sifat Masalah Non-Linier:** Data anggur memiliki hubungan non-linier yang kuat di antara fitur-fitur, dikonfirmasi oleh peningkatan Akurasi signifikan dari LR (0.54) ke LightGBM (0.79).
- **Imbalance Kelas Ekstrem:** Ketidakseimbangan data sangat ekstrem, memaksa LightGBM untuk memprioritaskan mayoritas, yang menyebabkan *Recall* Kelas 0 yang sangat rendah (0.17).
- **Metrik Evaluasi:** *Weighted F1-Score* adalah metrik terbaik untuk menilai kinerja global LightGBM karena memperhitungkan Akurasi sambil menyesuaikan bobot untuk ketidakseimbangan kelas.

### Insight dari Modeling

- **Kekuatan Algoritma Boosting:** Algoritma LightGBM terbukti paling unggul dalam kinerja Akurasi dan F1-Score untuk data tabular ini.
- **Keterbatasan LightGBM dalam Imbalance:** Meskipun akurat, LightGBM menunjukkan kelemahan tanpa Class Weights eksternal; model tersebut secara default memiliki **bias terhadap kelas mayoritas**, mengakibatkan kegagalan kritis pada Kelas 0.
- **Efektivitas Class Weights pada Model Lain:** Model LR dan MLP menunjukkan bahwa Class Weights efektif dalam mengarahkan fokus model ke Recall minoritas, mencapai Recall Kelas 0 yang jauh lebih baik (0.66 dan 0.60).

## 8.3 Kontribusi Proyek

Proyek ini menyediakan solusi prediktif berbasis LightGBM yang dapat digunakan dalam proses kontrol kualitas (QC) anggur:

- **Efisiensi Klasifikasi:** Sistem ini dapat secara efisien mengklasifikasikan mayoritas sampel dengan Akurasi tinggi (0.79), mempercepat proses inspeksi untuk sebagian besar produk.
- **Targeting Sumber Daya:** Meskipun Recall Kelas 0 rendah, analisis hasil menyoroti perlunya intervensi manual yang lebih intensif pada sampel yang diprediksi berada pada batas Kelas 0 (Buruk) dan Kelas 1 (Normal).

## **Pembelajaran yang Didapat:**

- **Penguasaan LightGBM:** Mendemonstrasikan implementasi LightGBM sebagai model advanced yang paling efektif untuk data tabular.
- **Manajemen Trade-off:** Belajar secara eksplisit bahwa model yang paling akurat (LightGBM) belum tentu merupakan model yang paling aman untuk kontrol kualitas karena trade-off yang dibuat pada kelas minoritas.
- **Kesulitan Klasifikasi Multi-Model dengan Imbalance Data:** Dari proyek ini, saya belajar bahwa klasifikasi multi-kelas menjadi jauh lebih sulit untuk mencapai hasil yang maksimal dan seimbang (**Akurasi Tinggi bersamaan dengan Recall Tinggi di Semua Kelas**) karena:
  - 1. Perbedaan Bobot Prioritas:** Setiap kelas menuntut perlakuan bobot yang berbeda, dan optimasi untuk satu kelas (misalnya, Kelas 1 Mayoritas) secara langsung merusak kinerja pada kelas lainnya (Kelas 0 Minoritas).
  - 2. Kompleksitas Batas Keputusan:** Pada klasifikasi biner, model hanya perlu menemukan satu batas. Namun, pada klasifikasi multi-kelas, model harus menemukan beberapa batas yang tidak saling tumpang tindih secara optimal, yang diperparah oleh perbedaan ukuran kelas.
- **Pentingnya Data:** Mengonfirmasi bahwa akar masalah (Recall rendah) terletak pada **kurangnya data sampel Kelas 0**, yang bahkan tidak dapat diatasi secara memadai oleh model canggih tanpa penyesuaian bobot.

## **9. Future Work**

Saran pengembangan untuk proyek selanjutnya

### **A. Data**

- Mengumpulkan Data Lebih Banyak
- Feature Engineering Lebih Lanjut

### **B. Model**

- Hyperparameter tuning lebih ekstensif
- Ensemble methods (combining models)

### **C. Deployment**

### **D. Optimization**

- Model compression (pruning, quantization)
- Improving inference speed

## **10. REPRODUCIBILITY**

### **10.1 GitHub Repository [Link Repository] :**

### **10.2 Environment & Dependencies**

**Python Version:** [3.8 / 3.9 / 3.10 / 3.11]

#### **Main Libraries & Versions:**

```
numpy==1.24.3
pandas==2.0.3
scikit-learn==1.3.0
matplotlib==3.7.2
seaborn==0.12.2

# Deep Learning Framework (pilih salah satu)
tensorflow==2.14.0 # atau
torch==2.1.0      # PyTorch

# Additional libraries (sesuaikan)
xgboost==1.7.6
lightgbm==4.0.0
opencv-python==4.8.0 # untuk computer vision
nltk==3.8.1        # untuk NLP
transformers==4.30.0 # untuk BERT, dll
```