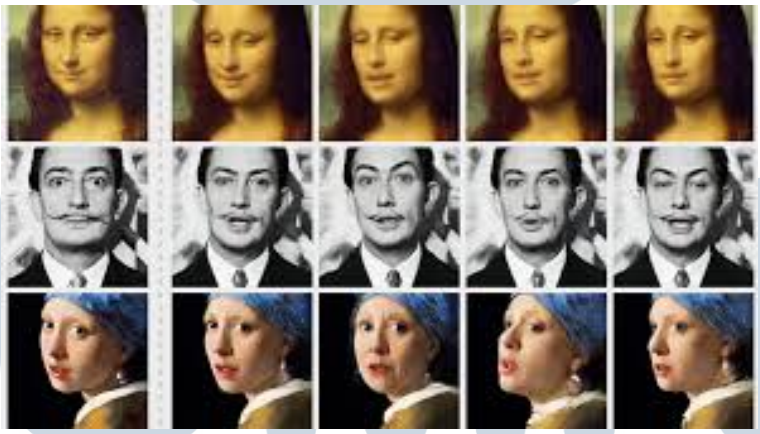
BAB 2
LANDASAN TEORI

2.1 Deepfake

Deepfake adalah istilah yang digunakan untuk menggambarkan media yang dihasilkan menggunakan teknik kecerdasan buatan (AI) yang mampu meniru wajah, suara, atau tindakan seseorang secara sangat realistis. Teknologi ini terutama menggunakan metode yang dikenal sebagai *Generative Adversarial Networks* (GANs), di mana dua jaringan neural, yaitu generator dan *discriminator*, dilatih secara bersamaan. Generator berusaha menciptakan gambar atau video palsu yang meyakinkan, sementara *discriminator* berusaha membedakan mana yang asli dan mana yang palsu. Melalui proses iteratif ini, *deepfake* yang dihasilkan menjadi semakin realistis. Selain mampu meniru wajah, suara, dan video, *deepfake* ini dapat menyebarkan misinformasi yang menyebabkan ketidakstabilan politik atau berbagai kejahatan dunia maya Gambar 2.1 [13].



Gambar 2.1. Contoh gambar wajah yang dihasilkan menggunakan teknologi deepfake.

2.2 Support Vector Machine

Support Vector Machines (SVM) adalah salah satu algoritma *Machine Learning* yang paling kuat dan paling banyak digunakan untuk klasifikasi data. Pada dasarnya tujuan utama (SVM) adalah menghasilkan margin *hyperplane* maksimum dalam ruang multidimensi yang memisahkan kelas-kelas yang berbeda. Kesalahan dapat diminimalkan dengan menghasilkan *hyperplane* secara iteratif [14] [15].

Kelebihan SVM:

1. Efektif untuk dataset berdimensi tinggi: SVM mampu menangani dataset dengan jumlah fitur yang besar.
2. Kemampuan generalisasi yang baik: Dengan margin maksimum, SVM cenderung menghindari overfitting.
3. Fleksibilitas kernel: SVM mendukung berbagai kernel (linear, polynomial, RBF) untuk menangani masalah klasifikasi non-linear.

Kekurangan SVM:

1. Kurang efisien untuk dataset besar.
2. Sulit untuk menafsirkan hasil model jika kernel kompleks digunakan.
3. Memerlukan tuning parameter (seperti kernel, C, dan gamma) yang tepat untuk mendapatkan kinerja optimal.

Konsep penting dari SVM:

1. Support Vectors adalah titik data yang paling dekat dengan *hyperplane* dan yang memengaruhi posisi *hyperplane* disebut sebagai *Support Vectors*.
2. *Hyperplane* adalah batas keputusan terbaik yang digunakan untuk memisahkan atau membagi kelas dalam ruangan n-dimensi. *Hyperplane* dibuat sedemikian rupa sehingga memiliki margin maksimum.

Misalkan terdapat tiga data contoh sebagai berikut:

- $(1,1)$ dengan label $+1$
- $(2,2)$ dengan label $+1$
- $(2,0)$ dengan label -1

Langkah 1: Tentukan hyperplane awal:

$$w_1 = 0, \quad w_2 = 0, \quad b = 0$$

Langkah 2: Perbarui hyperplane dengan iterasi (asumsi learning rate $\eta = 1$):

$$y_i(w \cdot x_i + b) \tag{2.1}$$

Iterasi pertama:

$$y_i(w \cdot x_i + b) = 1(0 \cdot 1 + 0 \cdot 1 + 0) = 0 < 1$$

$$w_1 = w_1 + \eta \cdot y_i \cdot x_{i1} = 0 + 1 \cdot 1 \cdot 1 = 1$$

$$w_2 = w_2 + \eta \cdot y_i \cdot x_{i2} = 0 + 1 \cdot 1 \cdot 1 = 1$$

$$b = b + \eta \cdot y_i = 0 + 1 \cdot 1 = 1$$

Hyperplane setelah iterasi pertama:

$$w_1 = 1, \quad w_2 = 1, \quad b = 1$$

Iterasi Kedua

$$y_i(w \cdot x_i + b) = 1(1 \cdot 2 + 1 \cdot 2 + 1) = 5 > 1$$

Karena $y_i(w \cdot x_i + b) > 1$, tidak perlu dilakukan pembaruan.
Iterasi Ketiga

$$y_i(w \cdot x_i + b) = -1(1 \cdot 2 + 1 \cdot 0 + 1) = -3 < -1$$

$$w_1 = w_1 + \eta \cdot y_i \cdot x_{i1} = 1 + 1 \cdot (-1) \cdot 2 = -1$$

$$w_2 = w_2 + \eta \cdot y_i \cdot x_{i2} = 1 + 1 \cdot (-1) \cdot 0 = 1$$

$$b = b + \eta \cdot y_i = 1 + 1 \cdot (-1) = 0$$

Hyperplane setelah iterasi ketiga:

$$w_1 = -1, \quad w_2 = 1, \quad b = 0$$

Keterangan

- y_i: Label dari data ke-i
- w_1, w_2: Parameter dari hyperplane
- b: Bias dari hyperplane
- η: Learning rate
- x_i: Data ke-i

- $y_i(w \cdot x_i + b)$: Persamaan hyperplane untuk data ke-i

2.3 Least Absolute Shrinkage and Selection Operator (LASSO)

Lasso adalah teknik regresi yang mengintegrasikan proses regularisasi dan seleksi fitur secara otomatis. Metode ini menambahkan penalti berupa jumlah nilai absolut dari koefisien regresi ($L1$-norm) ke fungsi biaya (*loss function*). Fungsi biaya untuk regresi Lasso dapat dituliskan sebagai:

$$\text{Loss Function} = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda \sum_{j=1}^{p}|w_j| \tag{2.2}$$

Keterangan:

- n: Jumlah data
- p: Jumlah fitur
- y_i: Nilai sebenarnya dari data ke-i
- $\hat{y}_i$: Prediksi model untuk data ke-i
- w_j: Koefisien regresi untuk fitur ke-j
- λ: Parameter regularisasi ($\lambda \geq 0$)

Penalti $\lambda \sum_{j=1}^{p}|w_j|$ menyebabkan beberapa koefisien regresi menjadi nol, sehingga Lasso secara otomatis melakukan seleksi fitur.

Keunggulan Lasso

- Seleksi Fitur Otomatis: Memilih fitur yang relevan dengan menghilangkan fitur yang kurang penting.
- Model yang Sederhana: Menghasilkan model yang lebih interpretable karena hanya mempertahankan fitur yang signifikan.
- Meningkatkan Generalisasi: Mengurangi risiko *overfitting* dengan menyederhanakan model.

Dalam konteks deteksi *deepfake*, Lasso dapat membantu mengidentifikasi fitur-fitur penting, seperti pola-pola kecil yang mungkin tidak terlihat secara langsung, sambil menghilangkan fitur yang tidak relevan [16].

2.4 SelectKBest

SelectKBest adalah metode seleksi fitur berbasis skor statistik yang digunakan untuk memilih sejumlah fitur (k) terbaik yang memiliki hubungan paling signifikan dengan variabel target. Proses ini melibatkan penerapan fungsi statistik tertentu, seperti:

1. Chi-square (χ^2) mengukur ketergantungan antara fitur dan label, digunakan untuk data kategorikal. Rumus untuk statistik χ^2 adalah:

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i} \tag{2.3}$$

dengan:

- O_i: Frekuensi observasi untuk kategori ke-i
- E_i: Frekuensi yang diharapkan untuk kategori ke-i

Chi-square digunakan ketika fitur dan target bersifat kategorikal, misalnya dalam pengelompokan warna atau jenis pola dalam deteksi *deepfake*.

2. ANOVA mengukur perbedaan rata-rata antar kelompok. Untuk fitur kontinu dan target kategorikal, statistik F-test dihitung sebagai:

$$F = \frac{\text{Variansi Antara Kelompok}}{\text{Variansi Dalam Kelompok}} \tag{2.4}$$

dengan:

- Variansi Antara Kelompok: Variasi rata-rata antar kategori
- Variansi Dalam Kelompok: Variasi di dalam kategori yang sama

ANOVA digunakan untuk fitur kontinu yang perlu dianalisis berdasarkan kategori target, misalnya membandingkan distribusi intensitas cahaya antara gambar *deepfake* dan asli.

3. Mutual Information ($I(X;Y)$) mengukur hubungan informasi antara fitur (X) dan target (Y). Rumusnya adalah:

$$I(X;Y) = \sum_{x \in X} \sum_{y \in Y} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} \tag{2.5}$$

dengan:

- $p(x,y)$: Probabilitas bersama antara X dan Y
- $p(x)$: Probabilitas marginal dari X
- $p(y)$: Probabilitas marginal dari Y

Penggunaan Mutual Information cocok untuk hubungan non-linear antara fitur dan target, seperti pola-pola kompleks pada tekstur kulit dalam gambar *deepfake*.

Keunggulan *SelectKBest* Setelah skor dihitung menggunakan salah satu fungsi statistik di atas, fitur-fitur dengan skor tertinggi dipilih untuk digunakan dalam model. Keunggulan utama *SelectKBest* meliputi:

- Meningkatkan Efisiensi: Dengan hanya menggunakan fitur paling relevan, waktu komputasi model berkurang.
- Mengurangi *Overfitting:* Dengan menghapus fitur-fitur yang tidak informatif, model menjadi lebih general.
- Meningkatkan Akurasi: Fokus pada fitur yang signifikan membantu model mempelajari pola yang lebih akurat.

Dalam konteks deteksi *deepfake*, *SelectKBest* dapat digunakan untuk memilih fitur penting, seperti pola pencahayaan, perbedaan tekstur kulit, atau artefak visual lainnya yang mencerminkan manipulasi [17].

2.5 Principal Component Analysis (PCA)

Principal Component Analysis (PCA) adalah teknik reduksi dimensi yang bertujuan untuk menyederhanakan data dengan mentransformasikan fitur-fitur asli ke dalam serangkaian komponen utama (*principal components*). Komponen-komponen ini disusun berdasarkan variansi, di mana komponen pertama memiliki variansi terbesar, diikuti oleh komponen kedua, dan seterusnya. Proses PCA melibatkan beberapa langkah utama berikut:

1. Langkah awal PCA adalah memastikan bahwa semua fitur memiliki skala yang sama. Hal ini dilakukan dengan menghitung nilai z-score:

$$z_{ij} = \frac{x_{ij} - \bar{x}_j}{s_j} \tag{2.6}$$

Keterangan:

- x_{ij}: Nilai fitur j untuk data ke-i
- $\bar{x}_j$: Rata-rata fitur j
- s_j: Standar deviasi fitur j

2. Matriks kovarians menghitung hubungan antara setiap pasangan fitur:

$$\mathbf{C} = \frac{1}{n-1} \sum_{i=1}^{n} (\mathbf{x}_i - \bar{\mathbf{x}})(\mathbf{x}_i - \bar{\mathbf{x}})^T \tag{2.7}$$

Keterangan:

- $\mathbf{x}_i$: Vektor fitur data ke-i
- $\bar{\mathbf{x}}$: Vektor rata-rata semua data
- $\mathbf{C}$: Matriks kovarians

3. PCA mengidentifikasi eigenvektor ($\mathbf{v}_i$) dan eigenvalue (λ_i) dari matriks kovarians:

$$\mathbf{C}\mathbf{v}_i = \lambda_i \mathbf{v}_i \tag{2.8}$$

Keterangan:

- $\mathbf{v}_i$: Eigenvektor yang menunjukkan arah variansi maksimum
- λ_i: Eigenvalue yang menunjukkan besar variansi sepanjang $\mathbf{v}_i$

Eigenvektor diurutkan berdasarkan eigenvalue terbesar untuk menentukan komponen utama.

4. Data asli ditransformasikan ke ruang baru menggunakan eigenvektor:

$$\mathbf{z}_i = \mathbf{X}\mathbf{V} \tag{2.9}$$

Keterangan:

- $\mathbf{X}$: Matriks data asli
- $\mathbf{V}$: Matriks eigenvektor yang terurut
- $\mathbf{z}_i$: Data dalam ruang komponen utama

PCA berguna untuk:

- Mengurangi Dimensi Data: Mengurangi jumlah fitur sambil tetap mempertahankan informasi penting, misalnya pada dataset *deepfake* dengan banyak fitur visual.
- Menghilangkan Korelasi: PCA mengubah fitur-fitur yang berkorelasi menjadi komponen utama yang ortogonal satu sama lain.
- Mempercepat Komputasi: Dengan data berdimensi lebih rendah, proses pelatihan model menjadi lebih cepat.

Dalam konteks deteksi *deepfake*, PCA membantu menyaring fitur penting seperti variansi dalam warna atau pola pada gambar yang mencerminkan manipulasi. Misalnya, PCA dapat digunakan untuk mereduksi data tekstur dan pola pencahayaan menjadi beberapa komponen utama yang mencerminkan perbedaan signifikan antara gambar asli dan *deepfake* [18].