**Slide 1: Pembukaan**
Selamat malam Bapak/Ibu dosen pembimbing dan penguji. Terima kasih saya ucapkan kepada:

- **Bapak Sukarsa dan Ibu Tari** selaku dosen pembimbing
- **Bapak Putu dan Ibu Suciptawati** selaku dosen penguji

yang telah menyempatkan waktunya untuk hadir pada ujian telaah pustaka saya. Perkenalkan, saya **Luh Sukma Mulyani**, NIM **2108541027**. Pada kesempatan ini, saya akan memaparkan **Telaah Pustaka** dari tugas akhir saya yang berjudul: **"Perhitungan Matriks Jarak pada Dokumen Teks."**

**Slide 4: Latar Belakang**
Perhitungan matriks jarak pada dokumen teks merupakan salah satu tahap penting dalam analisis data berbasis teks. Matriks ini digunakan untuk menggambarkan hubungan antar dokumen dalam bentuk numerik, sehingga memudahkan dalam proses pengelompokan, pencarian informasi, maupun klasifikasi, termasuk dalam algoritma seperti K-Nearest Neighbors (KNN).

Sebelum dokumen dapat direpresentasikan secara numerik dan dihitung tingkat kemiripannya, pemahaman terhadap konsep dasar seperti **Text Mining** dan **Document Similarity** diperlukan. Kedua konsep ini menjadi fondasi dalam membangun pemodelan berbasis teks secara sistematis dan terukur.

**Slide 6: Text Mining dan Document Similarity**

- **Text Mining** adalah cabang dari *data mining* yang berfokus pada ekstraksi informasi dari data teks yang tidak terstruktur. Tujuan utamanya adalah menemukan pola, struktur, dan makna tersembunyi dalam kumpulan teks.
- Salah satu konsep penting dalam text mining adalah **Document Similarity**, yaitu cara untuk mengukur **tingkat kemiripan** antar dua dokumen secara matematis.

Ukuran similarity ini menjadi dasar dalam penyusunan **matriks jarak**, yang sangat penting dalam analisis klasifikasi teks seperti K-Nearest Neighbors (KNN).

Sebelum nilai similarity dapat dihitung secara akurat, dokumen perlu melalui tahapan **text preprocessing** untuk memastikan bahwa representasi teks bersifat bersih, konsisten, dan relevan secara statistik.

**Slide 8: Text Preprocessing**
Tahapan preprocessing yang digunakan dalam penelitian ini mengacu pada Aggarwal (2018) dan mencakup enam langkah utama:

1. **Case folding**: mengubah seluruh huruf menjadi huruf kecil.
2. **Cleaning**: menghapus elemen-elemen non-informatif seperti tag HTML, tanda baca, dan karakter non-ASCII, agar teks menjadi bersih sebelum diproses lebih lanjut.
3. **Tokenisasi**: memecah teks menjadi unit kata atau token.

4. **Normalisasi**: mengubah kata tidak baku menjadi bentuk standar, seperti “luv” menjadi “love”.
5. **Stopwords removal**: menghilangkan kata umum yang tidak memberikan informasi penting, seperti “yang”, “dari”, atau “ke”.
6. **Stemming**: mengembalikan kata ke bentuk dasarnya, misalnya “bermain”, “dimainkan”, dan “memainkan” menjadi “main”.

Setelah keenam langkah ini, teks menjadi lebih bersih dan konsisten, sehingga siap direpresentasikan secara numerik untuk analisis lanjutan.

Setelah teks melalui tahap preprocessing, langkah selanjutnya adalah merepresentasikan dokumen ke dalam bentuk numerik.

**Slide 10: Representasi BoW**

BoW merepresentasikan dokumen sebagai kumpulan kata-kata, tanpa memperhatikan tata urutan atau struktur gramatikalnya. Informasi yang dipertahankan hanyalah frekuensi kemunculan kata dalam dokumen. Untuk keperluan penelitian ini, khususnya untuk perhitungan jarak Jaccard, kita akan menggunakan BoW biner, di mana nilai 1 diberikan jika sebuah kata muncul dalam dokumen, dan 0 jika tidak muncul.

Pendekatan representasi dokumen selanjutnya yang saya gunakan adalah Term Frequency-Inverse Document Frequency (TF-IDF).

**Slide 11: TF-IDF**

TF-IDF mengukur tingkat kepentingan suatu kata terhadap sebuah dokumen dalam suatu kumpulan dokumen (korpus). Bobot kata dihitung berdasarkan dua komponen utama:

- **Term Frequency (TF)**, sesuai Persamaan **(2.1)**, mengukur frekuensi kemunculan kata dalam sebuah dokumen.
- **Inverse Document Frequency (IDF)**, pada Persamaan **(2.2)**, mengukur seberapa jarang kata tersebut muncul di seluruh korpus.

Bobot akhir ditentukan melalui Persamaan **(2.3)**, yaitu hasil kali antara TF dan IDF.
Dalam penelitian ini, pembobotan TF-IDF diimplementasikan menggunakan pustaka **Scikit-learn**.
Berbeda dari rumus IDF standar, Scikit-learn menambahkan **konstanta satu** pada pembilang dan penyebut untuk menghindari pembagian nol dan menjaga kestabilan numerik. Rumus yang digunakan tercantum dalam Persamaan **(2.4)**.
Setelah dihitung, bobot TF-IDF dinormalisasi dengan **L2 normalization** (Persamaan **2.6**) agar perbandingan antar dokumen tidak dipengaruhi oleh panjang dokumen. Hasil akhirnya berupa vektor berdimensi tinggi yang menjadi input utama dalam perhitungan **matriks jarak** dan algoritma klasifikasi seperti **KNN**.

Setelah representasi numerik terbentuk, dokumen dirangkum ke dalam **Document Term Matrix (DTM)**

**Slide 12: Document Term Matriks**
**Document Term Matrix** atau **DTM** adalah sebuah matriks berukuran $N \times n$, berfungsi sebagai representasi numerik dari kumpulan dokumen, dan memungkinkan analisis lebih lanjut terhadap **pola kemunculan kata**, serta **hubungan antar dokumen**.

Sebagai contoh, pada matriks dengan 3 dokumen dan 4 kata unik yang ditampilkan di slide, baris pertama [2,0,1,0] menunjukkan bahwa **dokumen pertama** mengandung **kata pertama sebanyak dua kali**, dan **kata ketiga satu kali**, sedangkan kata kedua dan keempat tidak muncul. Representasi ini menjadi dasar untuk menghitung **matriks jarak** antar dokumen..

Dengan dokumen yang sudah direpresentasikan secara numerik melalui DTM, tahap selanjutnya adalah membahas **metrik jarak** yang digunakan untuk mengukur tingkat kemiripan antar dokumen.

**Slide 14: Metrik jarak**
Secara matematis, konsep kedekatan antar objek dalam analisis data dinyatakan melalui fungsi yang disebut **metrik jarak**. Fungsi ini digunakan untuk menghitung jarak antara dua titik dalam ruang berdimensi tinggi, dan harus memenuhi empat sifat dasar, yaitu: **tidak bernilai negatif**, **bernilai nol hanya jika objek identik**, **bersifat simetris**, serta **memenuhi sifat segitiga.**

Dalam konteks analisis dokumen teks, setiap dokumen direpresentasikan sebagai vektor dalam ruang multidimensi. Oleh karena itu, pemilihan metrik jarak sangat krusial. Jenis metrik yang digunakan harus disesuaikan dengan karakteristik data dan tujuan analisis, karena penggunaan metrik yang kurang tepat dapat memengaruhi akurasi hasil klasifikasi maupun pengelompokan. Pada penelitian ini saya menggunggunakan empat metrik, yaitu Euclidean, Manhattan, Jaccard, dan Cosine distance.

**Slide 15: Jarak Euclidean**
**Jarak Euclidean**, merupakan salah satu metrik jarak yang paling umum digunakan. Jarak ini mengukur **jarak lurus** atau **garis terpendek** antara dua titik, dan perhitungannya didasarkan pada **teorema Pythagoras**, sebagaimana ditunjukkan dalam **Persamaan (2.11)**.

**Slide 16: Jarak Manhattan**
**Jarak Manhattan** mengukur jarak antar dua titik berdasarkan **total perbedaan absolut** pada tiap dimensi. Metrik ini juga dikenal sebagai **city block** atau **taxicab distance**, karena menggambarkan pola gerak kendaraan yang hanya bisa bergerak lurus menyusuri jalan kota. Perhitungannya ditunjukkan pada **Persamaan (2.12)**

**Slide 17: Jarak Jaccard**
**Jarak Jaccard** digunakan untuk mengukur tingkat **ketidaksamaan** antara dua himpunan. Dalam konteks teks, setiap dokumen dianggap sebagai himpunan kata. Metrik ini menghitung **rasio antara irisan dan gabungan** dua himpunan kata. Semakin banyak kata yang sama, maka nilai **kesamaan Jaccard** semakin tinggi. **Kesamaan Jaccard** dihitung menggunakan **Persamaan (2.13)**, sedangkan **Jarak Jaccard** adalah kebalikannya, sesuai

**Persamaan (2.14)**. Nilainya berada antara **0 dan 1**, di mana 0 berarti dua dokumen sepenuhnya mirip, dan 1 berarti tidak ada kesamaan sama sekali.

**Slide 18: Jarak Cosine**
**Jarak Cosine** digunakan untuk mengukur **perbedaan arah** antara dua vektor dalam ruang berdimensi tinggi, tanpa mempertimbangkan panjang vektornya. Dalam konteks dokumen teks, setiap dokumen direpresentasikan sebagai vektor bobot kata, lalu dihitung sudut di antara vektor tersebut. **Kesamaan Cosine** dijelaskan dalam **Persamaan (2.15)**, yaitu berdasarkan hasil dot product yang dibagi dengan panjang masing-masing vektor. **Jarak Cosine** adalah kebalikannya, dihitung dengan **Persamaan (2.16)**, yaitu satu dikurang nilai kesamaan.

Setelah menghitung jarak antar setiap pasangan dokumen menggunakan metrik-metrik tersebut, hasilnya akan disimpan dalam sebuah Matriks Jarak.

**Slide 20: Matriks Jarak**
**Matriks jarak** adalah matriks persegi $n \times n$ yang menyimpan nilai jarak antar dokumen. Setiap elemen Dij menunjukkan jarak antara dokumen ke-i dan ke-j. Elemen diagonal selalu nol, dan matriks bersifat simetris jika metrik yang digunakan memenuhi sifat simetri. Struktur lengkapnya ditampilkan pada slide berikut.

Untuk mendemonstrasikan proses perhitungan matriks jarak, berikut di sajikan sebuah studi kasus sederhana.

**Slide 22: Studi kasus**
Untuk mengilustrasikan konsep yang telah dibahas, digunakan sebuah studi kasus sederhana. Dataset terdiri dari **10 dokumen teks acak** yang diambil dari korpus tugas akhir, dan setelah melalui proses preprocessing, diperoleh **20 kata unik** sebagai kosakata utama.
Setiap dokumen direpresentasikan dalam bentuk numerik menggunakan **Bag of Words** dan **TF-IDF**, lalu dihitung **matriks jaraknya** dengan beberapa metrik yang telah dijelaskan sebelumnya.
Hasil akhirnya adalah **matriks 10 × 10** yang menunjukkan tingkat kemiripan antar dokumen berdasarkan representasi dan metrik yang digunakan.

**Slide 23: Tahapan Preprocessing**
Seperti yang telah dijelaskan sebelumnya, dokumen melalui enam tahap preprocessing: case folding, cleaning, tokenisasi, normalisasi, stopwords removal, dan stemming.
Slide ini menunjukkan contoh hasil transformasi teks sebelum dan sesudah preprocessing, di mana teks mentah diubah menjadi token yang lebih bersih dan siap dianalisis.

Setelah proses prapemrosesan teks selesai, langkah berikutnya dalam penelitian ini adalah merepresentasikan dokumen ke dalam bentuk numerik, agar bisa dihitung dan dianalisis secara matematis.

**Slide 24: representasi Bag of Word**
Dokumen yang telah dipreproses kemudian direpresentasikan menggunakan metode **Bag of Words**. Pada metode ini, setiap dokumen diubah menjadi **vektor biner** berdasarkan kemunculan kata.
Contohnya dapat dilihat pada slide, di mana masing-masing dokumen direpresentasikan dalam bentuk vektor sepanjang 20 fitur kata unik.

**Slide 25-26: TF-IDF**
Representasi dokumen juga dihitung menggunakan **TF-IDF**, yaitu bobot yang menggabungkan **frekuensi kemunculan kata dalam dokumen** dan **tingkat kelangkaannya di seluruh korpus**. Nilai **TF** menunjukkan seberapa sering kata muncul dalam satu dokumen. Contohnya, kata *"url"* muncul satu kali di Dokumen 8, sehingga tf('url', D8) = 1
sedangkan 'happy_emoji' muncul tiga kali di D10, sehingga
tf('happy_emoji', D10) = 3

Kemudian **IDF** dihitung menggunakan **Persamaan (2.4)** dari Scikit-learn
Hasil perkalian keduanya menghasilkan bobot TF-IDF sesuai **Persamaan (2.3)**. Contoh pada slide menunjukkan kata *"url"* di Dokumen 8 memiliki bobot sebesar **1,788**.

Setelah bobot TF-IDF dihitung, vektor hasilnya dinormalisasi menggunakan **L2 normalization**. Proses ini dilakukan sesuai **Persamaan (2.6)**, yaitu dengan membagi setiap komponen vektor dengan panjang total vektornya.
Contoh pada slide menunjukkan vektor bobot Dokumen 8 dinormalisasi hingga panjangnya menjadi satu, menghasilkan nilai akhir seperti [0.474, 0.474, ..., 0.313].
Normalisasi ini penting agar panjang dokumen tidak memengaruhi hasil perhitungan jarak.

Setelah dokumen direpresentasikan dalam bentuk vektor TF-IDF yang telah dinormalisasi, dilakukan perhitungan jarak antar dokumen.

**Slide 27: Perhitungan manhattan**
Pada slide ini, digunakan **Manhattan Distance** untuk menghitung jarak antara **Dokumen 1 dan Dokumen 6**, sesuai **Persamaan (2.12)**. Jarak dihitung dengan menjumlahkan selisih absolut di setiap fitur. Hasil perhitungannya menghasilkan nilai akhir sebesar **2,663**, yang menunjukkan seberapa jauh kedua dokumen berdasarkan distribusi kata-katanya.

Semakin kecil nilai jarak Manhattan antar dokumen, semakin besar tingkat kemiripan antara kedua dokumen tersebut. Sebaliknya, nilai jarak yang besar menunjukkan perbedaan yang lebih signifikan.

**Slide 28: Perhitungan Jaccard**
Untuk representasi biner, digunakan **Jaccard Similarity** untuk mengukur kemiripan antar dokumen, sesuai **Persamaan (2.13)**. Nilai similarity dihitung dari rasio antara jumlah kata yang sama dengan total kata unik dari kedua dokumen. Selanjutnya, **Jaccard Distance** diperoleh dari **Persamaan (2.14)**, yaitu satu dikurangi nilai similarity. Pada contoh ini, hasil similarity adalah **0,2**, sehingga Jaccard Distance-nya adalah **0,8**.

Nilai 0.8 ini mengindikasikan bahwa 80% dari seluruh kata unik di kedua dokumen tersebut tidak dimiliki secara bersama, atau dengan kata lain, hanya 20% kata yang sama. Kondisi ini mencerminkan bahwa Dokumen 1 dan Dokumen 6 memiliki perbedaan yang cukup besar dalam hal konten kata yang digunakan.

**Slide 29: Perhitungan Euclidean**
Selanjutnya dilakukan perhitungan **Jarak Euclidean** antara Dokumen 1 dan Dokumen 6, menggunakan representasi TF-IDF yang telah dinormalisasi. Jarak ini dihitung berdasarkan **Persamaan (2.11)**, yang menjumlahkan kuadrat selisih antar komponen vektor, kemudian diakar. Hasil perhitungan memberikan nilai jarak sebesar **1,249**, yang menggambarkan tingkat perbedaan antara dua dokumen dalam ruang vektor. Semakin kecil nilai ini, semakin mirip kedua dokumen secara distribusi bobot kata.

**Slide 30: Perhitungan cosine**
Perhitungan berikutnya menggunakan **Cosine Similarity**, yang mengukur kemiripan arah antara dua vektor dokumen, tanpa memperhitungkan panjangnya. Perhitungan ini mengacu pada **Persamaan (2.15)**, yaitu dot product antara dua vektor dibagi hasil kali panjang vektor masing-masing.
Nilai Cosine Similarity antara Dokumen 1 dan Dokumen 6 adalah **0,219**, yang kemudian digunakan untuk menghitung **Cosine Distance** sesuai **Persamaan (2.16)**, yaitu satu dikurangi nilai similarity. Hasil akhir Cosine Distance sebesar **0,781** menunjukkan bahwa kedua dokumen cukup berbeda dalam hal arah distribusi kata.

**Slide 32: Kesimpulan**
Sebagai kesimpulan, perhitungan matriks jarak merupakan langkah penting dalam menganalisis dokumen teks karena dapat mengukur kemiripan atau perbedaan antar dokumen secara numerik. Proses ini dimulai dari tahap prapemrosesan teks, kemudian representasi vektor seperti Bag of Words atau TF-IDF, hingga perhitungan jarak menggunakan metrik tertentu seperti Manhattan dan Jaccard. Pemilihan metrik jarak harus disesuaikan dengan tujuan analisis dan karakteristik data. Dengan memahami proses ini secara menyeluruh, kita dapat menggali struktur dan pola dalam kumpulan teks secara lebih sistematis.