MODEL PENILAIAN JAWABAN ISIAN PENDEK MENGGUNAKAN UKURAN KEMIRIPAN TEKS DAN PEMBELAJARAN MESIN

TUGAS AKHIR

Karya tulis sebagai salah satu syarat
untuk memperoleh gelar Sarjana dari
Institut Teknologi Bandung

Oleh

SARAH SHABRINA

NIM: 10117010

(Program Studi Sarjana Matematika)

INSTITUT TEKNOLOGI BANDUNG
2021

ABSTRAK

MODEL PENILAIAN JAWABAN ISIAN PENDEK MENGGUNAKAN UKURAN KEMIRIPAN TEKS DAN PEMBELAJARAN MESIN

Oleh

Sarah Shabrina

NIM: 10117010

Penilaian jawaban isian pendek oleh manusia seringkali menemui beberapa permasalahan. Diantaranya yaitu kesalahan dan ketidakkonsistenan manusia dalam menilai jawaban karena bias, kelelahan atau yang lainnya dan kurangnya efektifitas, efisiensi penilai ketika ujian/tes/*assessment* berada dalam skala besar. Penilaian jawaban isian pendek dengan menggunakan mesin komputerpun menjadi alternatif solusi dalam menangani masalah tersebut.

Penelitian ini menggunakan model pembelajaran mesin dengan jenis pohon keputusan untuk melakukan penilaian. Fitur yang digunakan dalam pembangunan model adalah ukuran kemiripan teks *cosine similarity* berdasarkan *Bag Of Word* (BoW) dan *Latent Semantic Indexing* (LSI). *Cosine similarity* dengan BoW mengukur kemiripan dari segi leksikal (berkaitan dengan kata). Sedangkan *cosine similarity* dengan LSI mengukur kemiripan dari segi semantik (makna dari kata) tersebut. Ukuran kemiripan teks menghitung tingkat kemiripan antara jawaban murid dengan kunci jawaban. Dari dua fitur tersebut, dilakukan prediksi skor jawaban murid. Performansi model dievaluasi dengan ukuran akurasi. Akurasi dari model terbaik yang didapatkan dalam penelitian ini adalah sebesar 83%.

Kata kunci: penilaian, isian pendek, pembelajaran mesin, ukuran kemiripan teks, pohon keputusan

ABSTRACT

SHORT ANSWER GRADING MODEL USING TEXT SIMILARITY AND MACHINE LEARNING

by

Sarah Shabrina

NIM: 10117010

The assessment of short answer answers by humans often encounters several problems. There are human error and inconsistency in assessing answers due to bias, fatigue or the other and lack of effectiveness, efficiency of assessors when exams/tests/assessments are on a large scale. Assessment of short answers using a computer machine is an alternative solution in dealing with this problem.

This study uses a machine learning model with decision tree to conduct an assessment. The feature used in the building model is the text similarity by cosine similarity based on Bag Of Word (BoW) and Latent Semantic Indexing (LSI). Cosine similarity with BoW measures similarity in terms of lexical (related to words). Meanwhile, cosine similarity with LSI measures the similarity in terms of semantics (meaning of the word). The text similarity measure calculates the level of similarity between student's answers and the answer key. From these two features, the student's answer scores will be predicted. The performance of the model is evaluated with a measure of accuracy. The accuracy of the best model obtained in this study is 83%.

Keywords: grading, short answer, machine learning, text similarity, decision tree

MODEL PENILAIAN JAWABAN ISIAN PENDEK MENGGUNAKAN UKURAN KEMIRIPAN TEKS DAN PEMBELAJARAN MESIN

Oleh

Sarah Shabrina

NIM: 10117010

Program Studi Sarjana Matematika

Institut Teknologi Bandung

Menyetujui

Pembimbing

Tanggal Mei 2021

Ketua	Anggota
(Prof. Marcus Wono Setya Budhi, Ph.D.)	(Hilda Assiyatun M.Si.,Ph.D.)

Dipersembahkan untuk Tuhan, Bangsa, dan Almamater.

KATA PENGANTAR

Assalamu'alaikum Warahmatullahi Wabarakatuh

Puji syukur kehadirat Allah SWT yang telah memberikan nikmat, rahmat dan kesempatan penulis untuk dapat membuka mata, menghirup udara segar, sampai menggerakkan jari-jemari sehingga penulis telah berhasil menyelesaikan tulisan tugas akhir ini dengan tepat waktu.

Tugas Akhir ini berjudul "Model Penilaian Jawaban Isian Pendek Menggunakan Ukuran Kemiripan Teks dan Pembelajaran Mesin", dituliskan sebagai salah satu syarat untuk memperoleh gelar Sarjana dari Institut Teknologi Bandung. Karya ini bermaksud untuk memudahkan pekerjaan tenaga pengajar-pengajar yang ada dengan membuat model yang dapat menilai suatu jawaban ujian atau soal secara otomatis. Semoga tulisannya bermanfaat untuk perkembangan teknologi dalam ranah pendidikan ke depannya.

Penulis mengucapkan terimakasih kepada Bapak Marcus Wono Setya Budhi dan Bu Hilda Assiyatun, selaku dosen pembimbing, yang telah memberikan kritik, saran dan *feedback* dari awal proses perancangan Tugas Akhir hingga tulisan ini berhasil diselesaikan. Penulis juga mengapresiasi pihak-pihak lain yang berkontribusi membantu penyelesaian tugas ini baik secara langsung maupun tidak langsung.

Penulis menyadari bahwa karya ini jauh dari kata sempurna dan masih terdapat beberapa kekurangan. Oleh karena itu, penulis sangat mengharapkan saran dan kritik yang membangun dari pembaca untuk penyempurnaan tulisan ini.

Wassalamu'alaikum Warahmatullahi Wabarakatuh

Banyuwangi, 24 Mei 2021

Penulis

DAFTAR ISI

DAFTAR LAMPIRAN

DAFTAR GAMBAR

DAFTAR TABEL

Bab I Pendahuluan

Tulisan dalam Tugas Akhir ini diawali dengan pendahuluan yang berisi tentang gambaran secara singkat mengenai isi Tugas Akhir ini sekaligus memberikan hal acuan untuk masuk pada bab-bab berikutnya. Bab ini menjelaskan latar belakang, rumusan dan batasan masalah, tujuan dan manfaat penelitian, dan sistematika penulisan yang menjelaskan struktur pengorganisasian penulisan Tugas Akhir.

I.1 Latar Belakang

Dalam suatu proses pembelajaran, penilaian dari suatu pengetahuan memainkan peranan penting untuk pembelajaran yang efektif (Mohler dkk, 2011). Ada beberapa metode yang dapat digunakan dalam menilai kemampuan murid. Di antaranya yaitu penilaian dengan metode tugas, kuis, ujian yang memberikan pertanyaan sehingga mengharuskan murid untuk memberikan jawaban sesuai dengan pengetahuan yang dimilikinya. Tipe jawaban dari suatu pertanyaan dalam suatu penilaian tersebut juga bervariasi, yaitu tipe jawaban pilihan ganda, esai, dan isian pendek.

Jawaban isian pendek merupakan jawaban yang memenuhi beberapa kriteria, yaitu: Jawabannya berupa bahasa alami (*Natural language*), panjang jawaban di antara 1 frasa sampai 1 paragraf, jawabannya membutuhkan murid untuk menggunakan pengetahuan, di luar dari apa yang tertera di dalam soal, penilaiannya berfokus dengan konten jawaban tidak pada *grammar/style* tulisan, jawabannya dinilai secara objektif (Burrows dkk, 2005).

Kini, penilaian jawaban isian pendek oleh manusia seringkali memiliki beberapa problematika, diantaranya yaitu kesalahan dan ketidakkonsistenan manusia dalam menilai jawaban karena bias, kelelahan atau yang lainnya (Haley dkk, 2007) dan kurangnya efektifitas dan efisiensi penilai (manusia) ketika ujian/tes/*assessment* berada dalam skala besar (Burrows dan D'Souza, 2005).

Penilaian jawaban isian pendek dengan menggunakan teknik komputasipun menjadi alternatif solusi dalam menangani masalah tersebut. Penilaian otomatis

jawaban isian pendek adalah tugas untuk menilai respon atau jawaban yang bertipe isian pendek dengan menggunakan metode komputasi. Penilaian otomatis ini mengevaluasi jawaban murid dengan jawaban yang tertera pada kunci jawaban (Burrows dkk, 2005).

Oleh karena itu, dalam tugas akhir ini akan dikembangkan model penilaian jawaban isian pendek otomatis dengan menggunakan metode pemeriksaan ukuran kemiripan teks (*Text Similarity*) jawaban murid dengan jawaban yang benar dan pembangunan model pembelajaran mesin (*Machine Learning*). Metode ini berfokus pada pembuatan mesin yang dapat belajar tanpa diprogram secara eksplisit, untuk memprediksi skor dari jawaban murid secara otomatis.

I.2 Rumusan Masalah

Rumusan Masalah dalam tugas akhir ini adalah sebagai berikut:

1. Bagaimana ukuran kemiripan teks antar jawaban murid dengan jawaban yang benar?
2. Bagaimana model pembelajaran mesin untuk penilaian jawaban isian pendek?
3. Bagaimana performansi dari model pembelajaran mesin yang digunakan?

I.3 Tujuan Penelitian

Tujuan ditulisnya tugas akhir ini adalah sebagai berikut:

1. Menentukan ukuran kemiripan teks antar jawaban murid dengan jawaban yang benar.
2. Menentukan model pembelajaran mesin untuk penilaian jawaban isian pendek.
3. Menentukan performansi dari model pembelajaran mesin yang digunakan.

I.4 Batasan Masalah

Berdasarkan perumusan masalah dan tujuan penelitian, maka ruang lingkup penelitian tugas akhir ini dibatasi oleh beberapa hal, antara lain:

1. Jawaban dari murid yang digunakan adalah jawaban yang bertipe isian pendek.
2. Penilaian jawaban murid didasarkan pada kemiripan jawaban tersebut dengan kunci jawaban.

I.5 Manfaat Penelitian

Dari penelitian Tugas Akhir ini, diharapkan dapat memberikan beberapa manfaat, antara lain:

1. Memudahkan guru dalam memeriksa jawaban murid.
2. Mengurangi kesalahan manusia yang dapat mengurangi keakuratan penilaian jawaban pengerja soal karena ketidakkonsistenan atau subjektifitas.
3. Meningkatkan efektifitas dan efisiensi dari penilaian jawaban pengerja soal.
4. Dapat dijadikan referensi selanjutnya untuk penelitian yang berkaitan dengan penilaian jawaban isian pendek otomatis.

I.6 Sistematika Penulisan

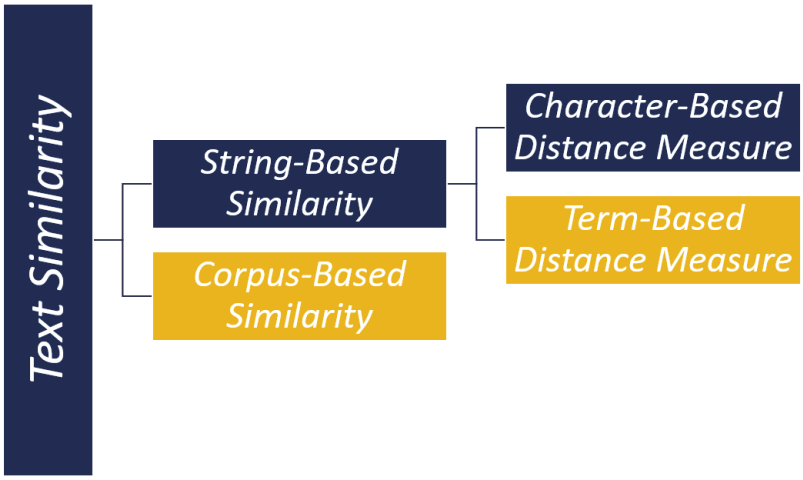
Tugas akhir ini terdiri dari 4 bab. Pada bab pertama dipaparkan mengenai gambaran awal tugas akhir. Pada bab dua dipaparkan tentang teori utama yang digunakan, yaitu ukuran kemiripan teks dan pembelajaran mesin. Bab tiga berisi model penilaian isian singkat otomatis dan analisisnya. Pada bab empat dipaparkan tentang penutup yang berisi kesimpulan dan saran.

Bab II Ukuran Kemiripan Teks dan Pembelajaran Mesin

Pada Bab II, akan dijelaskan tentang teori-teori utama yang digunakan dalam penelitian Tugas Akhir. Teori yang digunakan dalam Bab ini adalah teori yang menjelaskan kata kunci yang terdapat dalam rumusan masalah Tugas Akhir, yaitu ukuran kemiripan teks dan pembelajaran mesin.

II.1 Ukuran Kemiripan Teks (*Text Similarity*)

Ukuran kemiripan teks adalah suatu ukuran yang menggambarkan tingkat kemiripan antara satu teks dengan teks lainnya. Teks dapat terdiri dari beberapa kata, namun dapat pula terdiri dari banyak sekali kata yang tertulis dalam sebuah naskah (Sudardi, 2001:4-5). Secara umum, terdapat dua kategori cara untuk mengukur hal tesebut.



Gambar II.1: Jenis-jenis Ukuran Kemiripan Teks

Jenis yang pertama adalah ukuran kemiripan yang berdasarkan *string (String-Based Similarity)*. *String* adalah beberapa rangkaian dari karakter (huruf, angka, simbol) yang dipisahkan oleh spasi. *String-Based Similarity* memiliki 2 macam kategori, *Character-Based Distance Measure* dan *Term-Based Distance Measure*.

Dua kategori ini megukur kemiripan berdasarkan jarak antar teks. *Character-Based Distance Measure* mengukur jaraknya berdasarkan karakter, sedangkan *Term-Based Distance Measure* mengukurnya berdasarkan *term* atau kata.

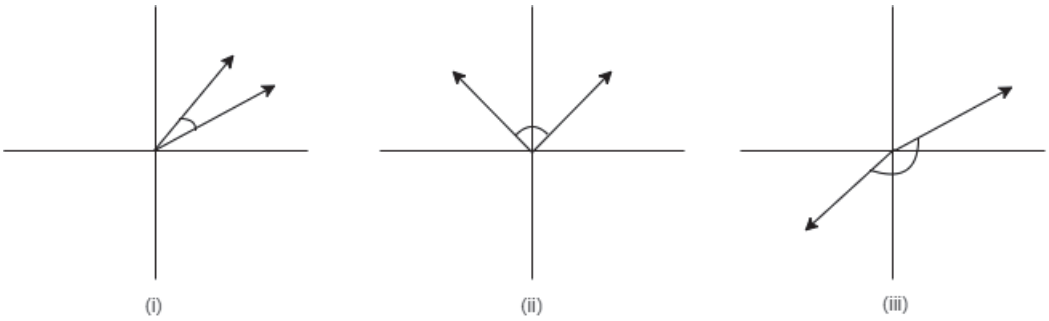
Jenis ukuran kemiripan yang kedua adalah *(Corpus-Based Similarity)*. Pengukuran ini adalah mempertimbangkan kemiripan dari sisi semantik yang menentukan kemiripan antar kata berdasarkan informasi yang ada. Berbeda dengan jenis kemiripan sebelumnya yang hanya melihat kemiripan secara permukaan saja, jenis ini mengukur kemiripan dari segi makna kata atau kalimat yang ada. Dalam Tugas Akhir ini, digunakan ukuran kemiripan *Term-Based Distance Measure* dan *Corpus-Based Similarity*.

II.1.1 Cosine Similarity

Salah satu contoh dari *Term-Based Distance Measure* adalah *Cosine Similarity*. Seperti namanya, ukuran ini menghitung nilai cosinus sudut antar dua vektor. Vektor yang digunakan dalam perhitungan ini adalah vektor mewakili teks yang ada. Persamaan matematikanya adalah sebagai berikut:

$$\cos\theta = \frac{\mathbf{a}.\mathbf{b}}{\|\mathbf{a}\|\|\mathbf{b}\|}$$

dengan a adalah vektor teks pertama dan b adalah vektor teks kedua



Gambar II.2: Ilustrasi 2D

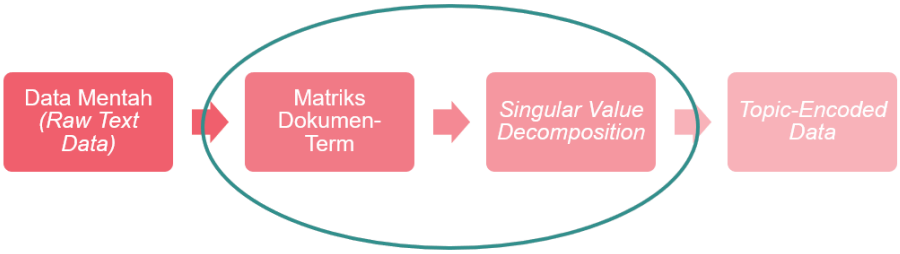
Rentang nilai *cosine similarity* adalah dari -1 hingga 1. Terdapat tiga kemungkinan kategori yang ada dalam ukuran ini :

(i) Kedua teks mirip,
yaitu ketika sudut antar vektornya mendekati nol dan nilai cosinusnya mendekati 1.
(ii) Kedua teks tidak berkaitan atau tidak mirip,
yaitu ketika sudut antar vektornya mendekati 90 derajat dan nilai cosinusnya mendekati nol.

(iii) Kedua teks berlawanan,
yaitu ketika sudut antar vektornya mendekati 180 derajat dan nilai cosinusnya negatif dan mendekati minus 1.

II.1.2 Latent Semantic Indexing

Latent Semantic Indexing (LSI) adalah salah satu contoh dari *Corpus-Based Similarity*, yang mengukur kemiripan suatu teks berdasarkan makna semantik antar teksnya. Teknik ini merupakan metode untuk mengekstrak dan mewakili penggunaan arti kontekstual dari suatu kata dengan komputasi statistik yang diaplikasikan untuk suatu *corpus* (tubuh) teks. LSI memberikan asumsi bahwa kata-kata yang memiliki kedekatan makna akan muncul di bagian yang sama dari suatu teks. LSI menggunakan prinsip matematis yaitu *Singular Value Decomposition* (SVD).



Gambar II.3: LSI

Dalam *Singular Value Decomposition*, dikenal istilah nilai eigen, vektor eigen, dan nilai singular *(singular value)*.

1. Nilai Eigen dan Vektor Eigen.
   Misal A adalah matriks berukuran n x n yang elemen-elemennya merupakan bilangan real. Jika **x** adalah vektor yang berdimensi n, maka perkalian matriks-vektor $A\mathbf{x}$ adalah vektor yang berdimensi n. Secara umum, perkalian matriks

mengubah arah dari vektor tak-nol x sehingga

$$A\mathbf{x} = \lambda\mathbf{x}$$

untuk suatu skalar λ.

Jika $\mathbf{Ax} = \lambda\mathbf{x}$, maka $(A-\lambda I)\mathbf{x} = 0$, di mana I adalah matriks identitas. Karena **x** tak-nol, maka $A - \lambda I$ mempunyai kolom yang bergantung sehingga determinan $|A - \lambda I| = 0$. Persamaan $|A - \lambda I| = 0$ memberikan solusi dari λ (nilai eigen dari A). Sedangkan **x** merupakan vektor eigen A.

2. Nilai Singular

   Sebelum dijelaskan mengenai nilai singular, perlu diketahui bahwa nilai eigen dari matriks $A^T A$ selalu bernilai non negatif karena matriks $A^T A$ adalah matriks definit non negatif. Bukti dari pernyataan tersebut yaitu:

   (a) Akan dibuktikan $A^T A$ dan merupakan matriks definit non negatif. Misal **x** adalah vektor berdimensi n, maka

   $$\mathbf{x}^{\mathrm{T}} A^{\mathrm{T}} A\mathbf{x} = (A\mathbf{x})^{\mathrm{T}} (A\mathbf{x}) = \|A\mathbf{x}\| \geq 0$$

   karena nilai norm suatu vektor selalu bernilai non negatif.

   (b) Akan dibuktikan nilai eigennya selalu bernilai non negatif. Misal λ adalah nilai eigen dari $A^T A$. Maka,

   $$\mathbf{x}^{\mathrm{T}} A^{\mathrm{T}} A\mathbf{x} = \mathbf{x}^{\mathrm{T}} \lambda\mathbf{x} = \lambda\|\mathbf{x}\|$$

   Karena telah dibuktikan bahwa $\mathbf{x}^{\mathrm{T}} A^{\mathrm{T}} A\mathbf{x} \geq 0$, maka $\lambda\|\mathbf{x}\| \geq 0$. Akibatnya, $\lambda \geq 0$ karena norm tidak bernilai negatif.

   Berikutnya, akan dijelaskan tentang nilai singular. Jika A adalah matriks yang berukuran mxn dan jika $\lambda_1, \lambda_2, \ldots, \lambda_n$ adalah nilai eigen dari $A^T A$ maka

   $$\sigma_1 = \sqrt{\lambda_1}, \sigma_2 = \sqrt{\lambda_2}, \ldots, \sigma_n = \sqrt{\lambda_n}$$

   adalah nilai singular dari A dengan

   $$\sigma_1 > \sigma_2 > \ldots > \sigma_n$$

Teorema SVD

Jika A adalah matriks yang berukuran mxn, maka A dapat dituliskan dalam bentuk

$$A_{m\times n} = U_{m\times m} S_{m\times n} V^T_{n\times n}$$

dengan U dan V adalah matriks ortonormal dan S adalah matriks diagonal yang memuat nilai singular dari A dengan urutan turun *(descending)*.
Kolom-kolom yang menyusun V merupakan vektor eigen dari $A^T A$ dan kolom-kolom yang menyusun U merupakan vektor eigen dari AA^T.

Misal terdapat A adalah matriks berukuran mxn dan dapat dilakukan dekomposisi SVD.

$$\mathbf{A} = \mathbf{USV}^{\mathrm{T}} \text{ dan } \mathbf{A}^{\mathrm{T}} = \mathbf{VSU}^{\mathrm{T}}$$
$$\mathbf{A}^{\mathrm{T}}\mathbf{A} = \mathbf{VSU}^{\mathrm{T}}\mathbf{USV}^{\mathrm{T}}$$
$$\mathbf{A}^{\mathrm{T}}\mathbf{A} = \mathbf{VS}^2\mathbf{V}^{\mathrm{T}}$$
$$\mathbf{A}^{\mathrm{T}}\mathbf{AV} = \mathbf{VS}^2$$

Dalam Tugas Akhir ini, SVD yang digunakan adalah SVD yang terpotong (*Truncated SVD*). Pemotongan ini dilakukan untuk mereduksi dimensi ke dimensi yang lebih rendah sehingga dapat merepresentasikan properti yang bermakna dari suatu data. Misal A adalah matriks yang berukuran mxn, maka A dalam *Truncated SVD* dapat dituliskan dalam bentuk:

1. Jika m<n, maka akan diperoleh "*truncated* V" SVD

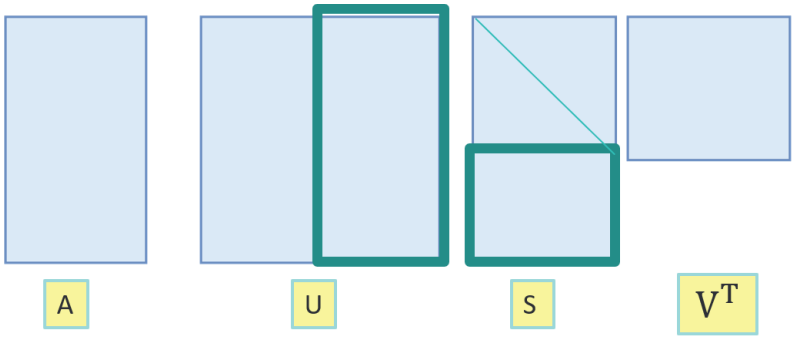
$$A_{m\times n} \approx U_{m\times m} S_{m\times m} V^T_{n\times m}$$

2. Jika n<m, maka akan diperoleh "*truncated* U" SVD

$$A_{m\times n} \approx U_{m\times n} S_{n\times n} V^T_{n\times n}$$

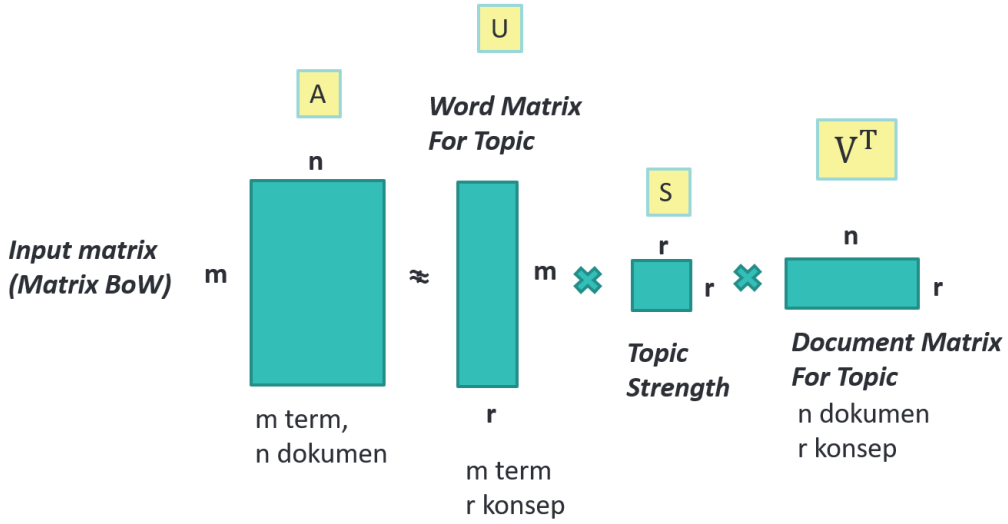
Selain itu, juga dapat dilakukan pemilihan besarnya pemotongan dengan memilih k dengan $0 < k < min(m, n)$, sehingga bentuk *Truncated SVD* menjadi:

$$A_{m\times n} \approx U_{m\times k} S_{k\times k} V^T_{n\times k}$$



Gambar II.4: Ilustrasi *Truncated SVD "U"*

Untuk penerapan konsepnya dalam LSI, matriks A dalam SVD merupakan matriks term-dokumen. Matriks U merupakan matriks term-konsep. Matriks S merupakan matriks kekuatan konsep *(Topic Strength)*. Matriks V merupakan matriks dokumen-konsep.



Gambar II.5: Aplikasi *Truncated* SVD dalam LSI

Metode ini digunakan untuk menyelesaikan masalah sinonim dan polisemi yang ada jika hanya menggunakan matriks term-dokumen biasa untuk menghitung ukuran kemiripan teks. Adanya sinonim dapat menyebabkan kata yang memiliki makna yang sama memiliki ukuran kemiripan yang berbeda, sedangkan adanya polisemi dapat menyebabkan kata yang memiliki makna yang berbeda akan memiliki ukuran kemiripan yang sama. Oleh karena itu, akan dicari makna tersembunyi (*Latent Semantic*) untuk mengekstrak informasi dari suatu dokumen atau teks dengan menggunakan LSI.

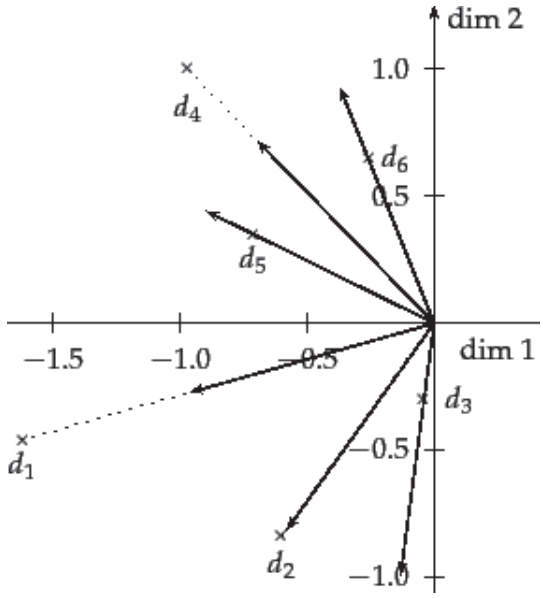
Setelah matriks didekomposisi dengan menggunakan *Truncated* SVD, perhitungan ukuran kemiripan teks dapat dilakukan dengan menggunakan *cosine similarity*. Persamaannya dalam kasus ini yaitu:

$$\cos\theta = \frac{\hat{q} \cdot \hat{v}_i}{\|\hat{q}\| \, \|\hat{v}_i\|}$$

dengan

$\hat{q}$: vektor 1,

$\hat{v}_i$: vektor 2, kolom ke - i dari matriks V (matriks dokumen-konsep)



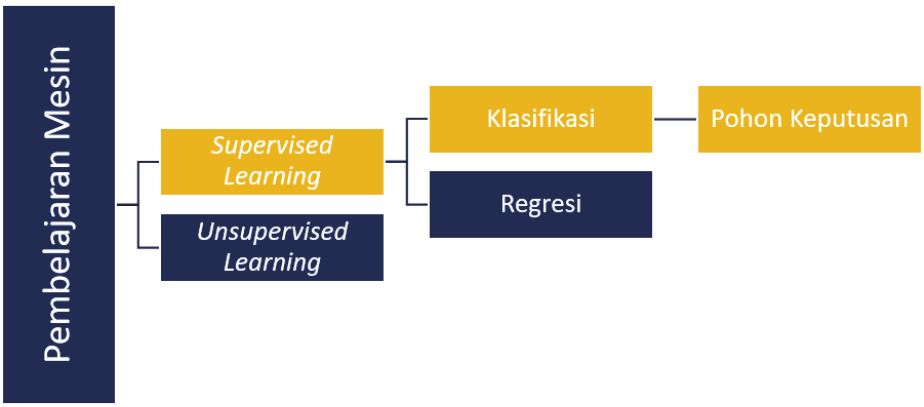
Gambar II.6: Ilustrasi LSI dalam 2 Dimensi

II.2 Pembelajaran Mesin *(Machine Learning)*

Pembelajaran Mesin adalah sains (dan seni) dari pemrograman komputer sehingga mereka dapat belajar dari data. Ide kunci dibalik pembelajaran mesin adalah untuk membangun algoritma yang dapat belajar dan melakukan prediksi dari data. Ilmu ini merupakan bagian dari Kecerdasan Buatan *(Artificial Intelligence)* yang berfokus kepada mengajarkan komputer bagaimana belajar tanpa perlu dilakukan pemrograman untuk tugas-tugas tertentu.

Terdapat dua jenis pembelajaran mesin yang umum digunakan, yaitu *Supervised Learning* dan *Unsupervised Learning*. Perbedaan dari dua jenis ini adalah pada ada atau tidaknya variabel target/label dari data. Variabel target merupakan variabel yang diprediksi oleh suatu model. *Supervised Learning* adalah model yang memiliki variabel target, sedangkan *Unsupervised Learning* adalah model

yang tidak memiliki target. Dalam Tugas Akhir ini, algoritma pembelajaran mesin yang digunakan Pohon Keputusan *(Decision Tree)* yang merupakan bagian dari *Supervised Learning*.

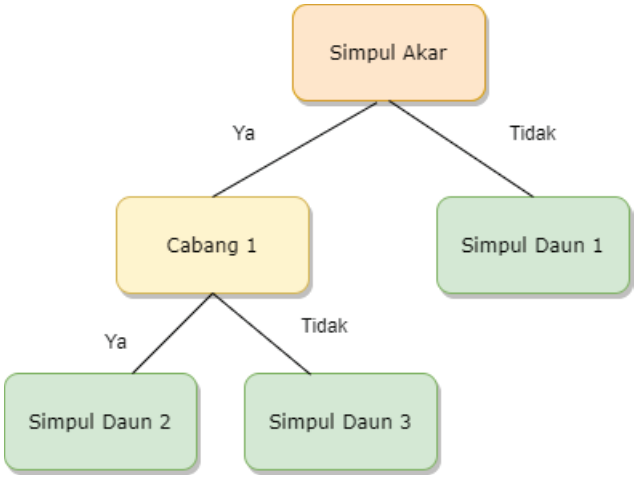


Gambar II.7: Jenis Pembelajaran Mesin

Supervised Learning	*Unsupervised Learning*
Data masukan memiliki label/target	Data masukan tidak memiliki label/target
Digunakan untuk prediksi	Digunakan untuk analisis
Dibagi menjadi dua tipe masalah: Regresi (hasil prediksi bertipe numerik) dan Klasifikasi (hasil prediksi bertipe kategorik)	Dibagi menjadi 2 tipe masalah: Asosiasi (menemukan aturan *(rules)* yang mendeskripsikan data dan *Clustering* (mengelompokkan data)

Tabel II.1: Perbedaan *Supervised Learning* dan *Unsupervised Learning*

Pohon keputusan adalah model prediksi yang menggunakan struktur pohon sebagai penyusunan algoritmanya. Seperti pohon pada umumnya, algoritma ini memuat simpul akar, cabang, dan daun dalam susunannya. Simpul akar merepresentasikan puncak dari pohon. Cabang berada pada bagian tengah dari pohon. Daun adalah bagian ujung dari pohon yang memuat target/hasil prediksi dari model.



Gambar II.8: Ilustrasi Pohon Keputusan

Sebuah pohon keputusan membagi kumpulan data yang besar menjadi himpunan-himpunan yang lebih kecil dengan menerapkan serangkaian aturan keputusan. Aturan pengambilan keputusannya adalah dari simpul akar hingga ke simpul daun. Dengan masing-masing rangkaian pembagian, anggota himpunan hasil pada simpul daun menjadi mirip satu dengan yang lain (Berry dan Linoff, 2004)

Dalam Pohon Keputusan, terdapat metodologi yang popular yang digunakan yaitu CART *(Classification and Regression Trees)*. Metodologi ini diperkenalkan pada tahun 1984 oleh Leo Breiman, Jerome Friedman, Richard Olshen dan Charles Stone. Konsep dasar dari metodologi CART adalah membangun pohon keputusan dengan menggunakan *Gini Impurity* sebagai kriteria pemisahan pohon. Pemisahan dilakukan dengan *(Binary Splitting)*, yaitu membuat pembentukan cabang dengan keputusan biner *(yes/no question)*.

Secara umum, terdapat dua urutan algoritma yang diproses dalam metodologi ini, yaitu membangun pohon dan memangkas pohon.

1. Membangun Pohon

   - Menentukan simpul akar *(Root-Node)*
     Untuk menentukan fitur yang akan dijadikan simpul akar, perlu dilakukan perhitungan *gini impurity*. *Gini Impurity* merupakan ukuran kemurnian suatu simpul dalam pohon keputusan. Misalkan diambil secara acak suatu data dari kumpulan data dan secara acak mengklasifikasi data tersebut ke suatu kelas/label. Maka, peluang ketika salah melakukan klasifikasi data tersebut adalah *Gini Impurity*.

$$
\begin{aligned}
\text{Gini}(t) &= \sum_{i=1}^{c} \sum_{j \neq i} p(i \mid t) * p(j \mid t) \\
&= \sum_{i=1}^{c} (p(i \mid t) * (1 - p(i \mid t))) \\
&= \sum_{i=1}^{c} \left(p(i \mid t) - (p(i \mid t))^2\right) \\
&= 1 - \sum_{i=1}^{c} (p(i \mid t))^2
\end{aligned}
$$

dengan t : simpul dalam pohon,

i,j : kelas dalam simpul

c : banyaknya kelas dalam simpul

Setelah menghitung Gini, maka dilakukan pemilihan fitur dengan menghitung Bobot Gini *(weighted gini)*. Pemilihan fitur didasarkan oleh pemilihan Bobot Gini paling rendah.

$$
\text{bobot Gini}(t) = \text{P}_\text{L} * \text{Gini}\,(t_L) + \text{P}_\text{R} * \text{Gini}\,(t_R)
$$

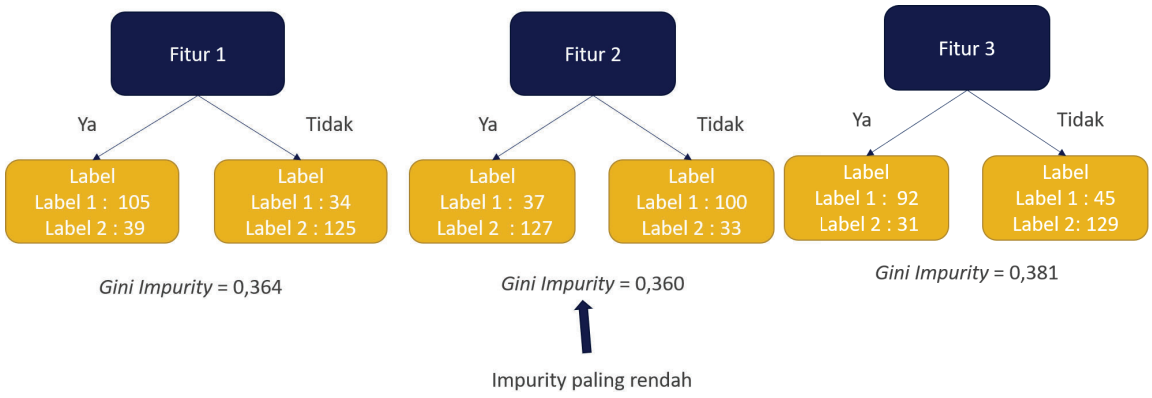
$$
\text{dengan } P_L = \frac{N_L}{N_L + N_R} \text{ dan } P_R = \frac{N_R}{N_L + N_R}
$$

Keterangan :

N : banyaknya data

L : cabang pohon kiri dan R : cabang pohon kanan

Untuk setiap fitur yang ada pada data latih, dihitung gininya. Fitur yang dijadikan simpul akar adalah fitur yang menghasilkan nilai gini paling rendah.



Gambar II.9: Contoh Penentuan Simpul Akar

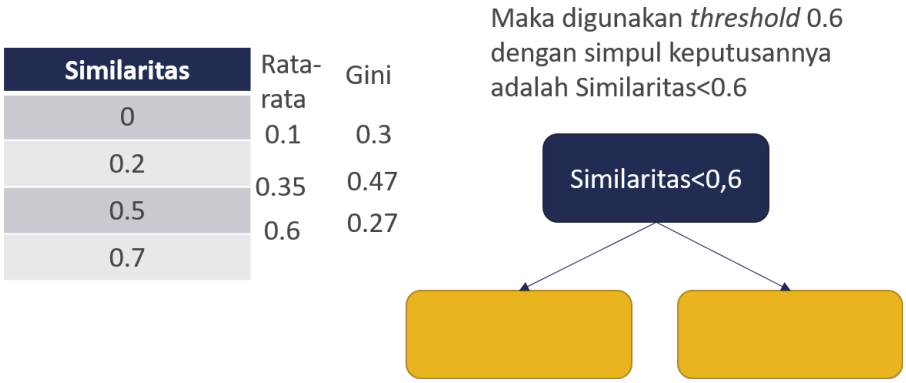
- Pemilahan *(Splitting Criteria)*

  Langkah selanjutnya yaitu pemilahan untuk membentuk cabang dari pohon.

Berikut algoritma dari proses ini:

1. Hitung semua skor *gini impurity* dan Bobot Gini
2. Jika simpul tersebut memiliki bobot gini paling rendah, maka tidak perlu dilakukan pemisahan lagi dan simpul tersebut menjadi simpul daun
3. Jika pemisahan data menghasilkan penurunan bobot *gini impurity*, maka ambil pemisahan dengan nilai Bobot Gini yang paling rendah

Dalam kasus Tugas Akhir ini, fitur bertipe numerik, maka penentuan *threshold* atau batas dari simpul keputusannya dilakukan dengan yang pertama yaitu mengurutkan fitur dengan urutan naik (*ascending*). Setelah itu, dihitung rata-rata 2 data yang berurutan dan dilakukan perhitungan bobot Gininya.



Gambar II.10: Ilustrasi Contoh Penentuan *threshold*

- Pemberhentian *(Stopping Criteria)*

  Proses pembentukan pohon klasifikasi berhenti saat memenuhi sekurang-kurangnya 1 dari:

  (a) Terdapat hanya satu pengamatan dalam tiap simpul, atau adanya batasan minimum banyakaknya data dalam simpul

  (b) Semua pengamatan dalam tiap simpul identik

  (c) Adanya batasan banyaknya level/kedalaman pohon maksimal.

Untuk menentukan kelas dalam simpul daun, dapat dipilih kategori kelas dengan banyaknya data terbanyak (modus). Atau dalam persamaan matematika dapat dituliskan sebagai:

$$p\left(j_0 \mid t\right) = \max_j p(j \mid t) = \max_j \frac{N_j(t)}{N(t)}$$

Dengan j_0 adalah label kelas yang ditentukan dalam simpul daun tersebut,
j adalah kelas yang ada,

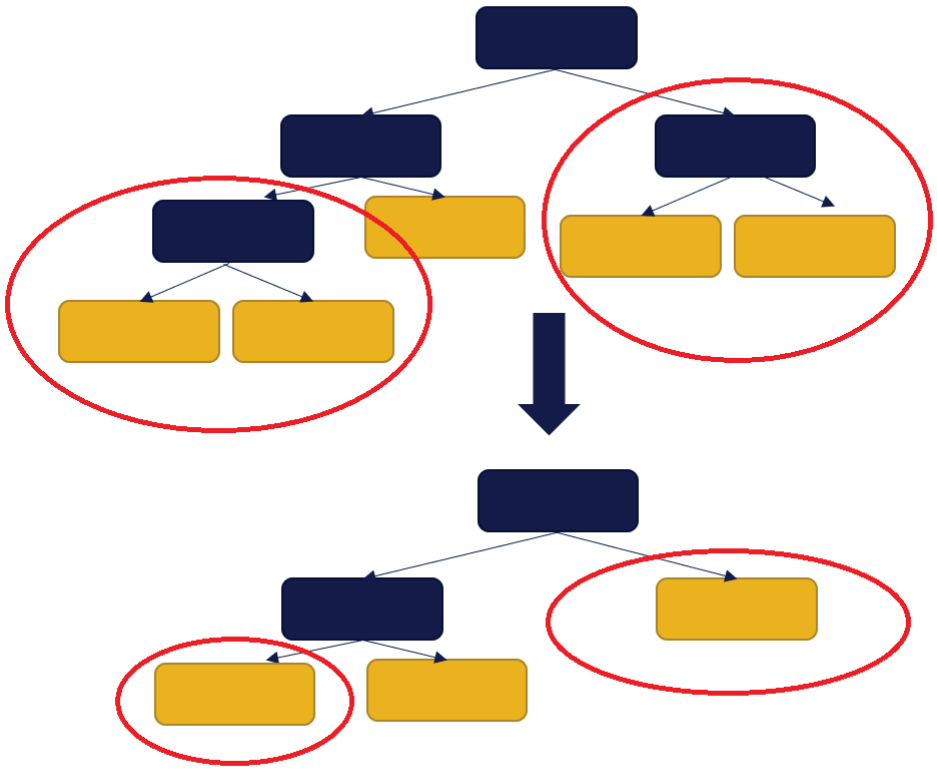
t adalah simpul daun,

$\mathrm{N}j(\mathrm{t})$ adalah banyaknya kelas j di t,

dan $\mathrm{N}(\mathrm{t})$ adalah banyaknya data dalam simpul daun

2. Memangkas Pohon *(Tree-Pruning)*

   Pada tahap ini, pohon keputusan telah tumbuh hingga ukuran maksimum. Selanjutnya dilakukan pemangkasan pohon untuk mengurangi kompleksitas dari pohon tersebut dan guna meningkatkan akurasi dalam melakukan prediksi pada data yang tidak terlihat. Pemangkasan tersebut dilakukan dengan mengganti subpohon atau cabang dengan simpul daun. Pemangkasan pohon memperhatikan ukuran evaluasi yang digunakan pada model.



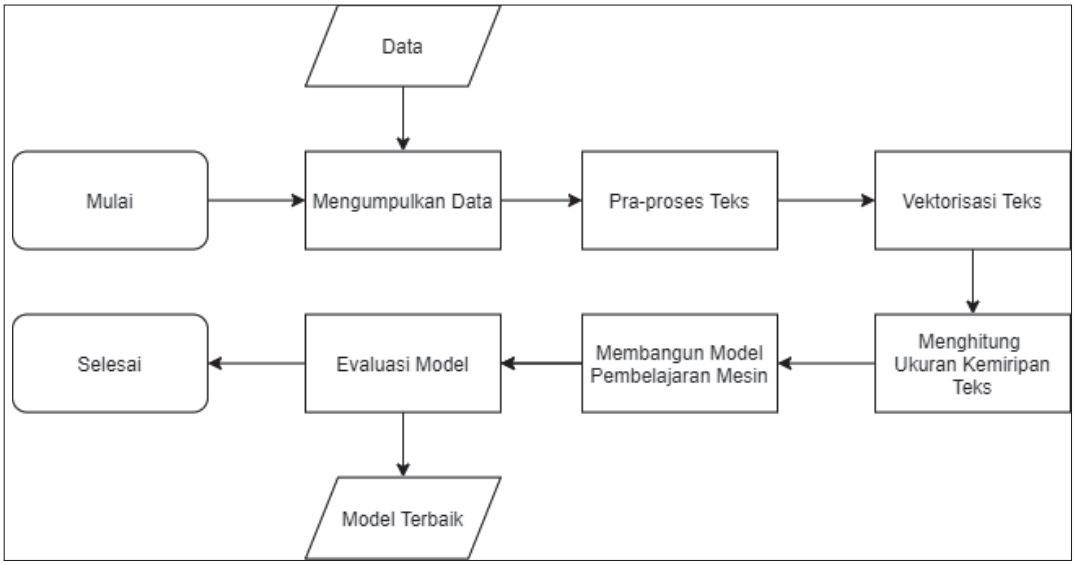
Gambar II.11: Contoh Pemangkasan Pohon

Bab III Analisis Model Penilaian Isian Singkat Otomatis

Pada Bab III, akan diterapkan landasan teori yang telah ada di Bab II dalam bentuk alur penelitian. Setelah mendapatkan hasil dari metode yang diteliti sesuai dengan alur penelitian, akan dilakukan proses analisis dan evaluasi terhadap hasil tersebut.

III.1 Alur Penelitian

Konsep dasar dari Tugas Akhir ini adalah mengolah data yang bertipe teks, menghitung ukuran kemiripan jawaban murid dengan jawaban sebenarnya (kunci jawaban) dan membangun model pembelajaran mesin untuk membuat model prediksi skor dari jawaban pengerja soal. Dari konsep dasar tersebut, terbentuk alur penelitian sebagai berikut:



Gambar III.1: Diagram Alir Penelitian Tugas Akhir

1. **Mengumpulkan Data**

   Data dari Tugas Akhir ini diperoleh dari *Automated Student Assessment Prize* (ASAP) oleh *The Willam and Flora Hewlett Foundation (Hewlett Foundation).* ASAP merupakan suatu kompetisi untuk menyelesaikan permasalahan dalam melakukan penilaian manual yang membutuhkan sumber daya yang tinggi.

ASAP diadakan pada tahun 2012. Kompetisi ini bertujuan untuk menemukan teknologi baru yang mendukung fasilitas sekolah dan kinerja guru.

Secara khusus, data yang diambil dalam Tugas Akhir dari ASAP adalah bagian *Short Answer Scoring* (SAS) (Penilaian Jawaban Isian Singkat). Data tersebut terdiri dari soal, kunci jawaban dan jawaban dari murid.

Detail dari data tersebut adalah sebagai berikut:

Nomor Soal	Subjek	Rata- Rata Panjang Jawaban	Jangkauan Skor
1	Sains	50 kata	0-3
2	Sains	50 kata	0-3
3	Bahasa Inggris	50 kata	0-2
4	Bahasa Inggris	50 kata	0-2
5	Biologi	60 kata	0-3
6	Biologi	50 kata	0-3
7	Bahasa Inggris	50 kata	0-2
8	Bahasa Inggris	50 kata	0-2
9	Bahasa Inggris	40 kata	0-2
10	Sains	60 kata	0-2

Tabel III.1: Detail data ASAP-SAS

Bahasa yang digunakan dalam soal maupun jawaban dalam data tersebut adalah Bahasa Inggris. Dari 10 nomor soal tersebut, dipilih soal nomor 5 untuk penelitian ini sebagai batasan masalah Tugas Akhir. Soal tersebut membahas tentang proses sintesis protein dalam tubuh manusia.

Teks dari soalnya adalah:

Starting with mRNA leaving the nucleus, list and describe four major steps involved in protein synthesis.

Kunci jawaban dari soal tersebut adalah elemen-elemen yang memuat:

• mRNA exits nucleus via nuclear pore. • mRNA travels through the cytoplasm to the ribosome or enters the rough endoplasmic reticulum. • mRNA bases are read in triplets called codons (by rRNA). • tRNA carrying the complementary (U=A, C+G) anticodon recognizes the complementary codon of the mRNA.

• The corresponding amino acids on the other end of the tRNA are bonded to adjacent tRNA's amino acids. • A new corresponding amino acid is added to the tRNA. • Amino acids are linked together to make a protein beginning with a START codon in the P site (initiation). • Amino acids continue to be linked until a STOP codon is read on the mRNA in the A site (elongation and termination).

Pada penjelasan data lanjutan, *Hewlett Foundation* memberikan kriteria penilaian jawaban murid yaitu:

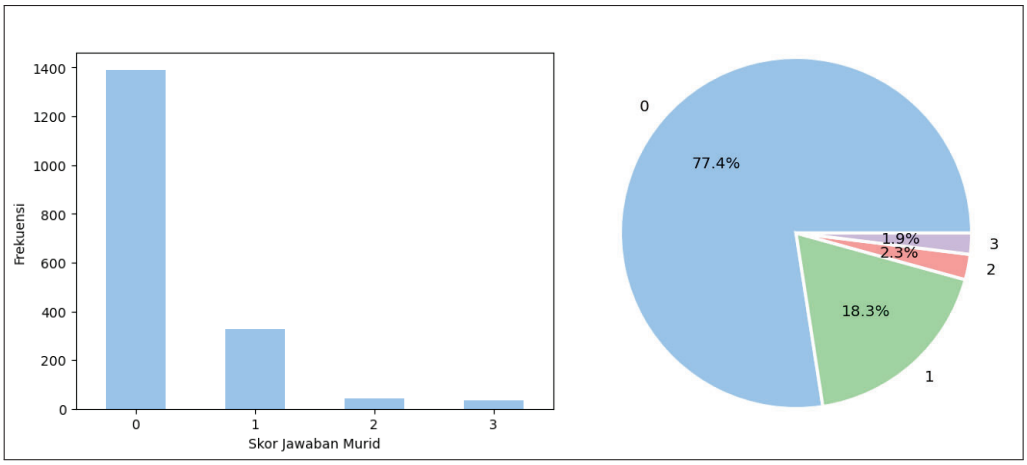
- 3 poin untuk jawaban yang memuat 4 elemen kunci.
- 2 poin untuk jawaban yang memuat 3 elemen kunci.
- 1 poin untuk jawaban yang memuat 1 atau 2 elemen kunci.
- 0 poin untuk jawaban lainnya.

Sedangkan contoh jawaban dari murid adalah sebagai berikut:

No.	Jawaban Murid	Skor
1	ATP is created and broken down. It is broken down when 2 phosphate groups don't connect with the third causing ADP. ADP can make ATP again by remaking itself.	0
2	The mRNA travels to the ribosomes. At the ribosomes the mRNA is copied so that is can be used to create proteins. The copies mRNA is then used as instructions to create proteins. The proteins are made, using the instructions given.	1
3	'First thing the mRNA does is go to the ribosome. Next, the tRNA comes and decodes the first three nucleotides. Then, the tRNA brings over the amino acid that matches the codon. Last, the tRNA shifts over and the amino acids form peptide bonds with one another.	2
4	Four major steps involved in protein synthesis are as follows, mRNA enters a ribosome. Then, tRNA delivers amino acids to the ribosome. The mRNA codon matches with the tRNA anti-codon. Also, the amino acids are joined by a condensation reaction. This process is repeated until a stop code is reached.	3

Tabel III.2: Contoh Jawaban Murid dari Soal Nomor 5

Untuk mengetahui gambaran informasi dan distribusi skor murid, dilakukan analisis data dengan menggunakan statistika deskriptif. Pengolahan data menggunakan bahasa Python. Banyaknya jawaban murid dalam data adalah sebanyak 1795 jawaban. Dari total jawaban murid tersebut, terdapat 77,4% jawaban murid memiliki skor bernilai 0, 18,3% memiliki skor bernilai 1, 2,3% memiliki skor bernilai 2, dan bernilai 1,9% memiliki skor 3. Dapat dilihat bahwa mayoritas jawaban murid memiliki skor 0.



Gambar III.2: Visualisasi Distribusi Skor Jawaban Murid. Kiri : Histogram, Kanan : Diagram *Pie*

2. **Pra-proses Teks.** ***(Text Pre-Processing)***

Dalam Tugas Akhir ini, data yang digunakan adalah data yang bertipe teks atau tulisan. Agar data lebih mudah dipahami, diprediksi dan dianalisis dalam algoritma Pembelajaran Mesin, dilakukan proses Pra-proses Teks. Teknik Pra-proses Teks berkontribusi dalam mengekstrak informasi dalam bentuk lebih sederhana sehingga mesin dapat menjalankan tugasnya dalam membangun model. Ada beberapa macam teknik yang tersedia untuk melakukan Pra-proses Teks. Dalam Tugas Akhir ini, Pra-proses Teks dilaksanakan dalam 5 tahap, diantaranya:

(a) Mengubah teks menjadi huruf kecil. *(Lowercasing)*

Pada umumnya, teks atau tulisan memuat huruf kecil dan huruf kapital. Di tahap ini, akan dilakukan konversi atau pengubahan setiap huruf yang ada pada kalimat menjadi huruf kecil. Proses pemrosesan data selanjutnya akan *case sensitive*, artinya terdapat perbedaan perlakuan terhadap kata yang mengandung huruf kapital dengan huruf kecil. Misalnya kata 'Protein'

akan diperlakukan berbeda dengan 'protein', meskipun kata tersebut memiliki susunan huruf dan arti yang sama. Jadi, proses pengubahan ini dibutuhkan agar informasi yang terdapat pada data tidak berkurang dan memudahkan untuk proses pemrosesan data selanjutnya.

(b) Menghapus tanda baca. *(Remove Punctuation)*

Mesin tidak memahami tanda baca, akibatnya keberadaan tanda tersebut akan membuat teks menjadi kotor dan *noisy*. Secara khusus, dokumen yang tidak terstruktur memiliki banyak sekali tanda baca, seperti tanda seru, tanda petik, tanda tanya, koma, titik, dan lain sebagainya. Proses penghapusan tanda baca ini menggunakan metode *Regular Expressions (Regex)*. Regex adalah metode yang umum digunakan untuk mencari pola yang terdapat di setiap huruf, sehingga dapat diidentifikasi dan dihapus tanda baca dengan aturan-aturan yang dibangun.

(c) Tokenisasi. *(Tokenization)*

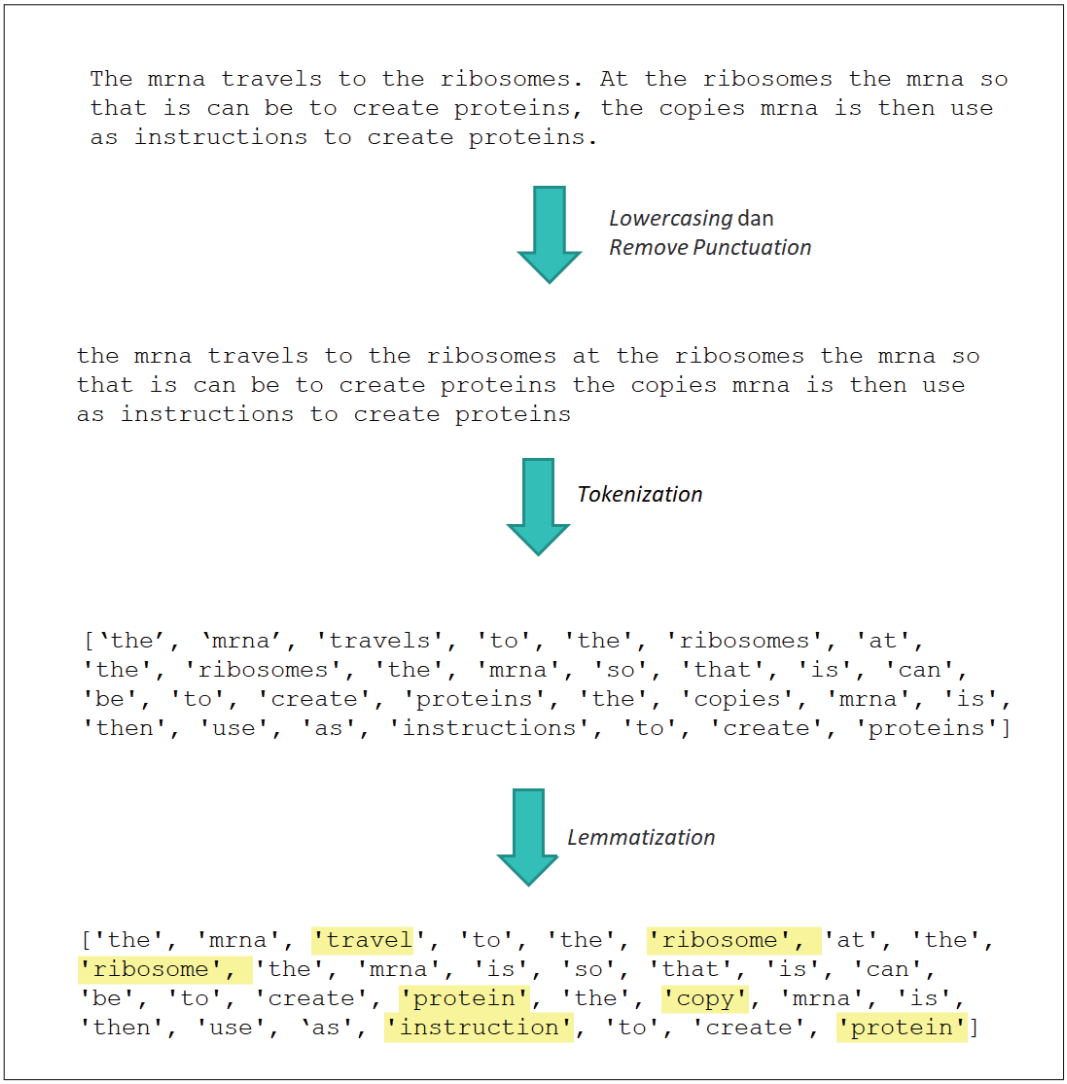
Tokenisasi merujuk kepada pemilahan atau pembagian teks yang berbentuk paragraf, kalimat dan kata menjadi suatu token atau bagian-bagian tertentu. Dalam tugas akhir ini, tokenisasi digunakan untuk membagi setiap jawaban pada murid dalam token yang berbentuk kata. Jadi, keluaran dari proses tokenisasi ini adalah kumpulan kata-kata yang didaftarkan dari setiap jawaban murid yang ada.

(d) Lematisasi. *(Lemmatization)*

Lematisasi merupakan proses pengubahan kata ke dalam bentuk dasar yang disebut dengan lema. Lema merupakan bentuk baku dari sebuah kata, khususnya entri kata dalam kamus. Proses lematisasi menghapus atau mengganti imbuhan yang ada pada suatu kata sehingga terbentuk lema baru yang bersesuaian dengan kata tersebut.

3. **Vektorisasi Teks**

Vektorisasi Teks adalah teknik untuk mengubah teks menjadi bentuk yang mudah dipahami oleh mesin yaitu vektor, susunan dari angka-angka bilangan bulat. Setiap kalimat yang ada, diekstrak dan direpresentasikan dalam bentuk vektor. Teknik vektorisasi teks yang digunakan dalam Tugas Akhir ini adalah



Gambar III.3: Pra-proses Data

Bag of Word (BoW). BoW merupakan salah satu teknik vektorisasi teks yang membangun vektor berdasarkan banyaknya kata unik yang terdapat dalam kalimat. Secara fundamental, BoW lebih berperan dalam mengekstrak informasi berdasarkan banyaknya kemunculan kata daripada di mana kata itu muncul dalam suatu teks. Alur pembentukan vektornya adalah:

(a) Menyusun setiap kata yang unik ke angka dan didaftarkan.

(b) Dalam setiap kalimat kemudian dihitung banyaknya setiap kata unik yang ada.

(c) Terbentuklah vektor BoW dari masing-masing teks.

	0	1	2	3	4	5	6	7	8	9	...
kata	with	nuclear	corresponding	after	rest	codon	u	new	travel	or	...
banyaknya	2	0	0	1	1	0	0	0	0	0	...

Gambar III.4: Contoh BoW

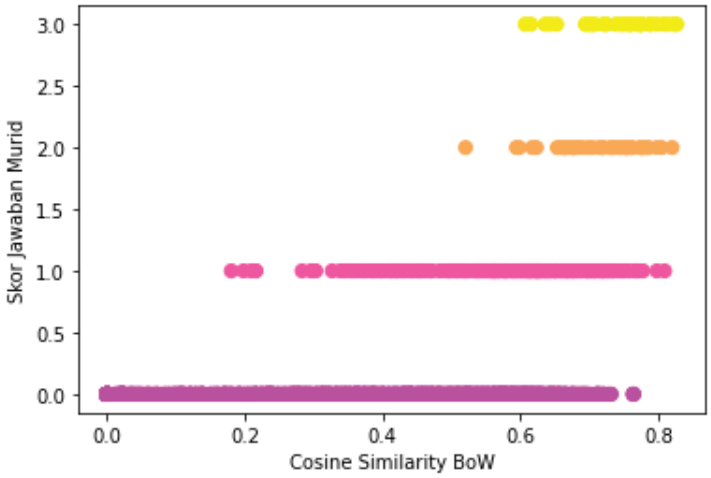
4. **Menghitung Ukuran Kemiripan Teks *(Text Similarity)*.**

   Setelah setiap jawaban murid dan kunci jawaban dikonversi menjadi vektor yang berisi bilangan, dilakukan perhitungan ukuran kemiripan antara jawaban murid dengan kunci jawaban. Perhitungan ukuran kemiripan ini dilakukan berdasarkan persamaan *cosine similarity*. Keluaran dari tahap ini adalah ukuran *cosine similarity* berdasarkan *Bag Of Word* dan *Latent Semantic Indexing* yang akan dijadikan fitur pemodelan pembelajaran mesin.

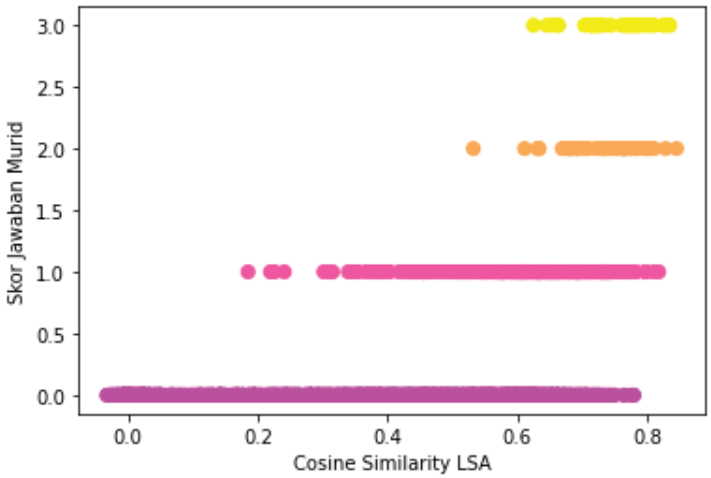
   Berikut contoh hasil penerapan *cosine similarity*:

Jawaban Murid	*Cosine Similarity* BoW	*Cosine Similarity* LSI
ATP is created and broken down. It is broken down when 2 phosphate groups don't connect with the third causing ADP. ADP can make ATP again by remaking itself.	0,159384	0,171577
The mRNA travels to the ribosomes. At the ribosomes the mRNA is copied so that is can be used to create proteins. The copies mRNA is then used as instructions to create proteins. The proteins are made, using the instructions given.	0,640092	0,648480
protein synthesis give the body protein so it can maintain at tip top shape,without this are body could not survive	0,172806	0,203893

Tabel III.3: Contoh *Cosine Similarity*



Gambar III.5: Visualisasi *Cosine Similarity* BoW

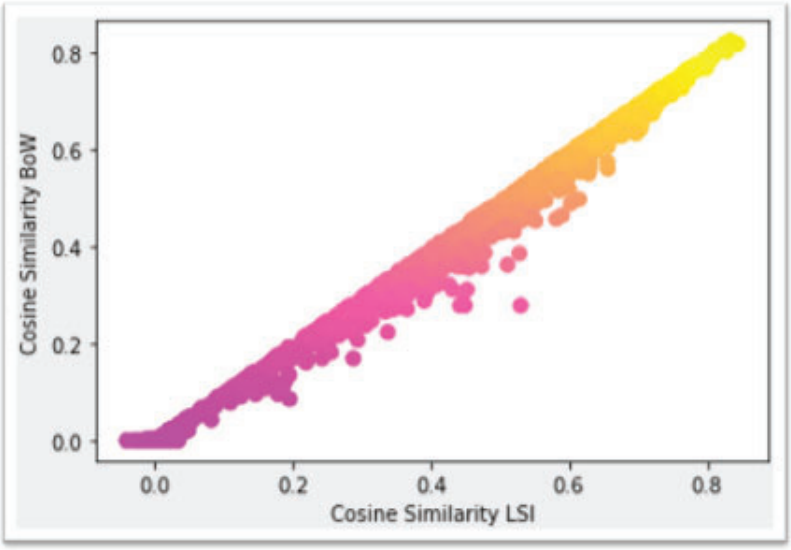


Gambar III.6: Visualisasi *Cosine Similarity* LSI

Berdasarkan Gambar III.5, III.6 dapat dilihat bahwa ukuran kemiripan dengan BoW dan LSI memiliki pola yang mirip. Berikutnya akan dilihat hubungan antara dua ukuran kemiripan ini:

Skor	Rata-rata *Cosine Similarity* BoW	Rata-rata *Cosine Similarity* LSI
0	0,339753	0,359741
1	0,585728	0,604872
2	0,711328	0,728474
3	0,746068	0,758259

Tabel III.4: Rata-rata *Cosine Similarity*



Gambar III.7: Visualisasi *Cosine Similarity* BoW dengan *Cosine Similarity* LSI

Skor	Jangkauan *Cosine Similarity* BoW	Jangkauan *Cosine Similarity* LSI
0	0 - 0,764069	-0,030532 - 0,779944
1	0,180139 - 0,808779	0,185218 - 0,818093
2	0,520096 - 0,819689	0,532510 - 0,844268
3	0,606761 - 0,826112	0,626255 - 0,834699

Tabel III.5: Jangkauan Cosine Similarity per Skor

Berdasarkan Gambar III.7, *cosine similarity* BoW dengan LSI memiliki hubungan kelineran yang positif. Kemudian terlihat juga bahwa titik *scatter*nya berhimpit ke garis y=x. Pada tabel III.5 dan III.6 juga dilihat bahwa antara *cosine similarity* BoW dan LSI memiliki rata-rata dan jangkauan yang diduga hampir mirip. Untuk mengujinya digunakan uji hipotesis dengan statistika non parametrik.

Uji tersebut adalah uji Mann–Whitney U, dengan H_0 adalah distribusi dua populasi sama dan H_1nya adalah distribusi dua populasi tidak sama. Dengan menggunakan modul *scipy* di python didapat bahwa p-value=0.0022. Misalkan tingkat signifikansi (*alpha*) yaitu 0.05. Karena pvalue<alpha maka H_0 ditolak. Artinya distribusi dua populasi (*Cosine Similarity* BoW dan LSI) tidak sama.

5. **Membangun Model Pembelajaran Mesin *(Machine Learning)*.**
   Selanjutnya dilakukan pembangunan model dengan menggunakan pohon

keputusan (*decision tree*) yang telah dijelaskan pada Bab II. Pada tugas akhir ini, terdapat empat kelas yang menggambarkan skor jawaban murid yaitu 0, 1, 2 dan 3. Berikut adalah tahap-tahap pembangunan modelnya:

(a) Pemilahan Data (*Data Splitting*).

Pemilahan data dilakukan dengan tujuan agar model dapat dievaluasi menggunakan data yang tidak digunakan pada saat pembangunan model. Data dipilah dengan aturan 80/20, di mana 80% dari data dikategorikan menjadi data latih. Data latih adalah data yang digunakan dalam pemodelan pohon keputusan. Sedangkan 20% dari data dikategorikan menjadi data uji. Data uji adalah data yang digunakan dalam evaluasi model.

(b) Pembangunan Model.

Model pohon keputusan dibangun dengan model dasar terlebih dahulu dengan *hyperparameter* yang ditentukan di awal. *Hyperparameter* adalah variabel yang memengaruhi keluaran model. Nilai *hyperparameter* tidak diubah selama model dioptimisasi. Dengan kata lain, nilai *hyperparameter* tidak bergantung pada data dan selalu diambil saat pendefinisian model.

(c) *Hyperparameter Tuning*.

Setelah model dasar *(baseline model)* terbentuk, dilakukan *hyperparameter tuning*. Tujuannya adalah untuk menemukan nilai *hyperparameter* yang dapat menghasilkan model dengan performa yang paling baik dengan. Dalam Tugas Akhir ini digunakan algoritma *GridSearchCV* yang terdapat pada *library* scikit-learn dalam python untuk menemukan *hyperparameter* yang optimal.

(d) Prediksi Data Uji.

Setelah itu, dilakukan prediksi pada data uji dengan model yang telah dipilih. Berikut beberapa contoh hasil prediksi dengan model:

Jawaban Murid	Skor Aktual	Skor Model
it takes it to whatever place it goes then takes it to all of the other RNAs	0	0
It forms from DNA. Then it forms protiens	0	0

Tabel III.6: Contoh Skor Model dan Skor Aktual dari Murid

Jawaban Murid	Skor Aktual	Skor Model
1. the mRNA leaves the nucleus and searches for a ribosome 2. the mRNA finds a ribosome and attaches to it 3. the mRNA is decoded by tRNA via base pairing rules and the amino acids, previously synthesized, are brought in as indicated by the mRNA. 4. the ribosome moves on down the mRNA strand, assembling the amino acids in the correct order.	1	2
mRNA travels to a tRNA and relays that message. Then the tRNA starts to copy the mRNA so the cell can have a replicate. The new mRNA travels to ribosomes. The ribosomes begin to make the protein.	1	1

Tabel III.7: Contoh Skor Model dan Skor Aktual dari Murid

6. **Evaluasi Model.**

   Setelah dilakukan pembangunan model dengan menggunakan pohon keputusan, dilakukan proses peninjauan hasil. Hasil klasifikasi dari model dirangkum dengan menggunakan *confusion matrix*. *Confusion matrix* atau matriks konfusi adalah tabel yang merangkum banyaknya kejadian antara hasil klasifikasi sebenarnya atau aktual dengan hasil klasifikasi dari model. Matriks konfusi memiliki informasi tentang:

   (a) *True Positive* (TP) : Banyaknya data yang hasil aktualnya positif dan model mengidentifikasinya dengan benar sebagai positif.

   (b) *True Negative* (TN) : Banyaknya data yang hasil aktualnya negatif dan model mengidentifikasinya dengan benar sebagai negatif.

   (c) *False Positive* (FP) : Banyaknya data yang hasil aktualnya negatif dan model mengidentifikasinya dengan salah sebagai positif.

   (d) *False Negative* (FN) : Banyaknya data yang hasil aktualnya positif dan model mengidentifikasinya dengan salah sebagai negatif.

	Aktual		
		Positif	Negatif
Model	Positif	TP *(True Positive)*	FP *(False Positive)*
	Negatif	FN *(False Negative)*	TN *(True Negative)*

Tabel III.8: Matriks Konfusi

Untuk mengevaluasi model, digunakan metrik akurasi untuk melihat kebagusan model. Akurasi mengukur seberapa benar suatu model memprediksi suatu data.

Jika memilih suatu unit dari data secara acak dan memprediksi kelasnya, akurasi adalah peluang bahwa prediksi dari model tersebut benar.

$$\text{Akurasi} = \frac{TP + TN}{TP + TN + FP + FN}$$

Dalam Tugas Akhir ini, matriks konfusi yang digunakan adalah:

		Aktual			
		0	1	2	3
Model	0	N_{00}	N_{01}	N_{02}	N_{03}
	1	N_{10}	N_{11}	N_{12}	N_{13}
	2	N_{20}	N_{21}	N_{22}	N_{23}
	3	N_{30}	N_{31}	N_{32}	N_{33}

Tabel III.9: Matriks Konfusi Tugas Akhir

dengan N_{ij} adalah banyaknya data dengan skor model i dan skor aktual j. Sehingga metriks evaluasinya adalah:

$$\text{Akurasi} = \frac{\sum_{n=0}^{3} N_{ii}}{\sum_{n=0}^{3} N_{ij}}$$

Fitur dalam Model	*Hyperparameter Tuning*	Akurasi
Cosine similarity BoW	Tidak	0,79
Cosine similarity BoW	Ya	0,79
Cosine similarity LSI	Tidak	0,79
Cosine similarity LSI	Ya	0,79
Cosine similarity BoW dan LSI	Tidak	0,82
Cosine similarity BoW dan LSI	Ya	0,83

Tabel III.10: Perbandingan Model

Model terbaik dipilih yang memuat kriteria ukuran akurasi terbaik yaitu pohon keputusan dengan fitur *cosine similarity* BoW dan LSI. Matriks konfusi dari model pohon keputusan yang terbaiknya yaitu:

	Aktual				
		0	1	2	3
Model	0	291	43	3	1
	1	3	6	4	5
	2	0	0	0	0
	3	2	1	1	1

Tabel III.11: Matriks Konfusi Tugas Akhir

III.2 ANALISIS HASIL

Dari gambar matriks konfusi dan akurasi, diketahui bahwa hasil klasifikasi mempunyai tingkat akurasi sebesar 83%. Hal ini menunjukan model dengan fitur ukuran kemiripan *cosine similarity* BoW dan LSI telah membuat model memiliki peluang kebenaran untuk mengklasifikasi sesuatu sebesar 83%. Namun, masih terdapat 17% kesalahan dalam klasifikasi jawaban skor murid. Kesalahan terjadi karena terdapat jawaban skor murid yang terklasifikasi tidak sesuai dengan data referensi yang digunakan. Kesalahan-kesalahan tersebut dapat disebabkan oleh beberapa faktor, diantaranya:

1. Ketidakseimbangan banyaknya data dalam suatu target/label/kelas

   Pada data yang dimiliki, diketahui bahwa distribusi targetnya miring *(skew)*. Dari total jawaban murid tersebut, terdapat 77,4% jawaban murid memiliki skor bernilai 0, 18,3% memiliki skor bernilai 1, 2,3% memiliki skor bernilai 2, dan bernilai 1,9% memiliki skor 3. Dapat dilihat bahwa mayoritas jawaban murid memiliki skor 0. Hal ini menunjukkan ketidakseimbangan distribusi target yang cukup besar karena pada aturan umum dikatakan bahwa data yang minoritas kurang dari 10% dari total data termasuk kategori tak seimbang.

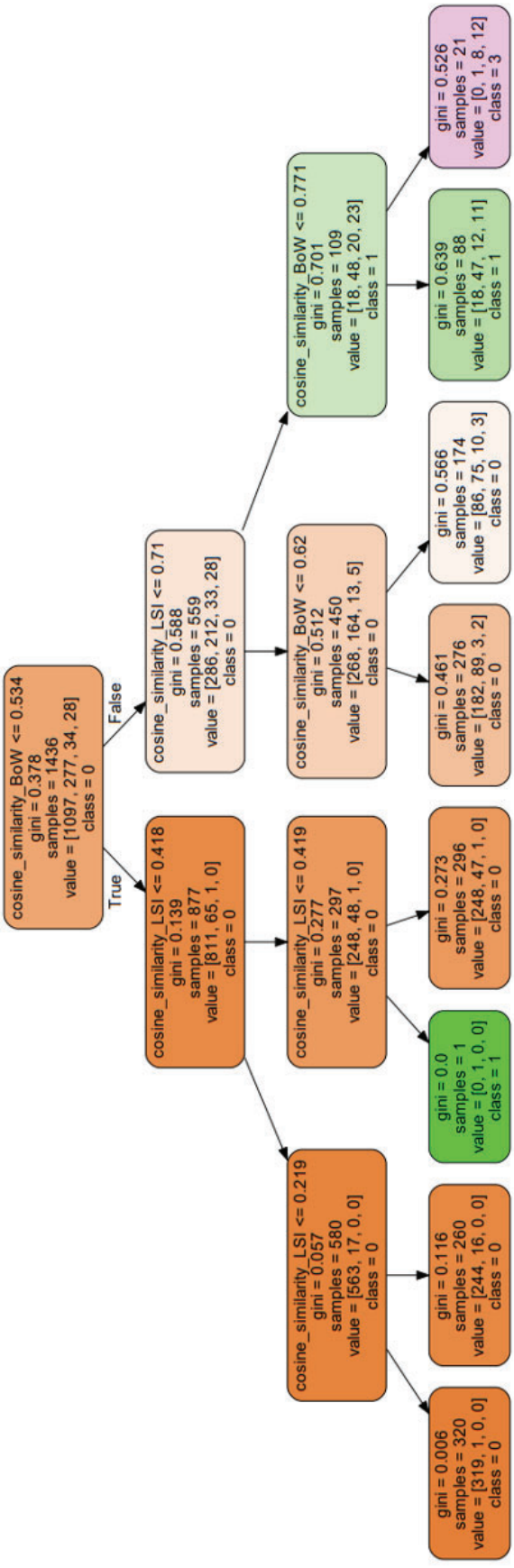
   Dalam model pembelajaran mesin, model akan lebih baik belajar ketika distribusi data targetnya seimbang. Model akan lebih bias ke mayoritas target dan cenderung untuk memprediksi target sesuai dengan target yang mayoritas.

   Untuk kasus Tugas Akhir ini dapat dilihat bahwa dalam model bahkan tidak berhasil memprediksi data yang memiliki skor 2. Seperti yang diketahui, skor jawaban murid yang bernilai 3 pada data hanya sebesar 2,3% dari total data.

2. Ukuran kemiripan teks yang tumpang tindih *(overlapping)*

   Seperti yang dilihat dalam Gambar III.5 dan III.6, nilai *cosine similarity*, baik dari BoW maupun LSI, antara satu skor dengan skor lainnya tumpang tindih. Misal nilai *cosine similarity* antara 0.4-0.5 dapat termasuk ke dalam skor jawaban murid 0 dan 1. Hal ini dapat menimbulkan ambiguitas model saat

melatih data. Faktor yang dapat menyebabkan hasil ini adalah ukuran kemiripan teks yang digunakan tidak memperhatikan urutan kata dan tidak memperhatikan *grammar* pada jawaban murid.



Gambar III.8: Hasil Model Pohon Keputusan

Bab IV Penutup

IV.1 Kesimpulan

Kesimpulan dari penelitian Tugas Akhir ini yaitu:

1. Ukuran kemiripan teks yang digunakan dalam Tugas Akhir ini adalah *cosine similarity* dengan BoW (*Bag of Words*) dan LSI (*Latent Semantic Indexing*). *Cosine similarity* dengan BoW mengukur kemiripan dari segi leksikal (berkaitan dengan kata). Sedangkan *cosine similarity* dengan LSI mengukur kemiripan dari segi semantik (makna dari kata) tersebut. Dalam analisis Tugas Akhir ini, diperoleh bahwa antar 2 ukuran kemiripan tersebut memiliki hubungan kelinearan (korelasi) yang positif.
2. Model pembelajaran mesin dalam Tugas Akhir ini adalah pohon keputusan dengan algoritma CART (*Classification And Regression Tree*). Algoritma ini memiliki konsep *binary-splitting* dan *gini impurity* dalam pembentukan pohonnya. Sedangkan fitur yang dipilih dalam model ini adalah *cosine similarity* dengan BoW (*Bag of Words*) dan LSI (*Latent Semantic Indexing*).
3. Peformansi dari model pohon keputusan dalam Tugas Akhir ini cukup baik. Ukuran performansi menggunakan akurasi yang berdasarkan matriks konfusi yang ada dari model terbaik yang dipilih. Nilai akurasi yang didapat yaitu sebesar 83%.

IV.2 Saran

Saran untuk penelitian selanjutnya dapat dilakukan untuk meningkatkan performansi dari model. Berikut hal-hal yang dapat dipertimbangkan untuk membangun model:

1. Melakukan praproses data yang berkaitan dengan data yang tidak seimbang.
2. Menggunakan metrik evaluasi lain yang bersesuaian dengan kondisi data.
3. Menambah fitur lain yang relevan dengan model.

DAFTAR PUSTAKA

Aly, F., Gomaa,W. (2012). Short Answer Grading Using String Similarity And Corpus-Based Similarity. International Journal of Advanced Computer Science and Applications, 3(11). https://doi.org/10.14569/ijacsa.2012.031119

Awad, M., Khanna, R. (2015). Efficient learning machines: Theories, concepts, and applications for engineers and system designers. Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers, January, 1–248. https://doi.org/10.1007/978-1-4302-5990-9

Burrows, S., Gurevych, I., Stein, B. (2015). The eras and trends of automatic short answer grading. In International Journal of Artificial Intelligence in Education (Vol. 25, Issue 1). https://doi.org/10.1007/s40593-014-0026-8

Chiu, S., Tavella, D. (2008). Introduction to Data Mining. Data Mining and Market Intelligence for Optimal Marketing Returns, 137–192. https://doi.org/10.1016/b978-0-7506-8234-3.00007-1

Galitskaya, E. G., Galitskkiy, E. B. (2013). Classification trees. Sotsiologicheskie Issledovaniya, 3, 84–88. https://doi.org/10.4018/978-1-60960-557-5.ch006

Grandini, M., Bagli, E., Visani, G. (2020). Metrics for multi-class classification: An overview. ArXiv, 1–17. Mili ˇ, M. (2020). Document Comparison Based on the Page Layout. 1, 2–6.

Shalev-Shwartz, S., Ben-David, S. (2013). Understanding machine learning: From theory to algorithms. In Understanding Machine Learning: From Theory to Algorithms (Vol. 9781107057135). https://doi.org/10.1017/CBO9781107298019

Barbara, R. (2000). Latent Semantic Indexing: An overview. Infosys 240, 1–16.

Martin, D. I., Berry, M. W. (2015). Mathematical Foundations Behind Latent Semantic Analysis. Handbook of Latent Semantic Analysis, 35–55. https://doi.org/10.4324/9780203936399.ch2

Wael, G Fahmy, A. (2012). Short Answer Grading Using String Similarity And Corpus-Based Similarity. International Journal of Advanced Computer Science and Applications, 3(11). https://doi.org/10.14569/ijacsa.2012.031119

LAMPIRAN

Lampiran A Algoritma Python untuk Praproses Data

```
# Mengumpulkan Data

# Impor Paket
import pandas as pd
#impor data jawaban murid
df=pd.read_table("train2.tsv")
df.head()

# Saring Data untuk Soal Nomor 5
df5jaw=df[df['EssaySet']==5]
df5jaw

# Eksplorasi Data Analisis dengan Visualisasi
df5jaw.groupby('Score1').size().plot.bar()
ax = plt.gca()
labels =["0","1","2","3"]
ax.set_xticklabels(labels=labels,rotation=0)
plt.xlabel("Skor Jawaban Murid")
plt.ylabel("Frekuensi")
fig, ax = plt.subplots(figsize=(6, 6))
ax.pie(df5jaw.groupby('Score1').size(), labels=labels,
       autopct='%.1f%%',
       wedgeprops={'linewidth': 3.0, 'edgecolor': 'white'},
       textprops={'size': 'x-large'})
plt.tight_layout()

# Pra-proses Data
#Impor Paket
```

```
import nltk
import numpy as np
import re
jaw5=df5jaw['EssayText']

#Lowercasing
jaw5=jaw5.str.lower()
jaw5bener=np.char.lower(jaw5bener)

#Remove Punctuation
pd.set_option('max_colwidth', None)
items = []
for kata in jaw5:
    item = []
    item.append (' '.join(re.sub("(@[A-Za-z0-9]+)|
    ([^0-9A-Za-z \t])| (\w+:\/\/\S+)",
    " ", kata).split()))
    items.append(item)
jaw5 = pd.DataFrame(data=items, columns=['jawaban 5'])
jaw5
symbols = "!\"#$%&()*+-./:;<=>?@[\]^_`{|}~\n"
for i in symbols:
    jaw5bener = np.char.replace(jaw5bener, i, ' ')
jaw5bener

#Tokenize
nltk.download('punkt')
jaw5t = []
for i in jaw5['jawaban 5']:
    jaw5t.append(nltk.word_tokenize(i))
jaw5t

#Lemmatisasi
nltk.download('wordnet')
from nltk.stem import WordNetLemmatizer
lemmatizer = WordNetLemmatizer()
```

```
lemma=[]
for i in range(len(jaw5t)):
    lemmat = [lemmatizer.lemmatize(word) for word in jaw5t[i]]
    lemma.append(lemmat)
print(lemma)

#list of list to list of sentence
lst=[]
for i in range(len(lemma)):
    lst.append(' '.join(word for word in lemma[i]))
lst
```

Lampiran B Algoritma Python untuk Ukuran Kemiripan Teks

```
#Impor Paket
import math

#Definisi Fungsi Cosine Similarity
def cosine_sim(vec1, vec2):
    vec1 = list(vec1)
    vec2 = list(vec2)
    dot_prod = 0
    for i, v in enumerate(vec1):
        dot_prod += v * vec2[i]
    mag_1 = math.sqrt(sum([x**2 for x in vec1]))
    mag_2 = math.sqrt(sum([x**2 for x in vec2]))
    return dot_prod / (mag_1 * mag_2)


#Hitung Cosine Similarity dari Matriks Bag of Words
cosineBoW=[]
for i in range(len(lst)):
    bagOfWordsA = lst[0].split(' ')
    bagOfWordsB = lst[i].split(' ')
    uniqueWords = set(bagOfWordsA).union(set(bagOfWordsB))
    numOfWordsA = dict.fromkeys(uniqueWords, 0)
    for word in bagOfWordsA:
        numOfWordsA[word] += 1
    numOfWordsB = dict.fromkeys(uniqueWords, 0)
    for word in bagOfWordsB:
        numOfWordsB[word] += 1
    cosineBoW.append(cosine_sim(numOfWordsA.values(),
```

```
                    numOfWordsB.values()))
cosineBoW

#Gabungkan hasil dengan dataframe awal
df2=pd.DataFrame(cosineBoW, columns=['cosine_similarity'])
dfhasil = pd.concat([df2, df5jaw.reset_index()], axis=1)
dfhasil

#Visualisasi Hasil Cosine Similarity
import seaborn as sns

X = dfhasil.cosine_similarity
y = dfhasil.Score1
plt.scatter(X[1:], y[1:], c=y[1:], s=50, cmap='spring')

#LSA
#Impor Paket
import nltk
import re
import numpy as np
from gensim import corpora, similarities
from gensim.models import LsiModel
from nltk.corpus import stopwords
nltk.download("stopwords")
nltk.download("wordnet")

#Proses
documents_test=lemma[0]
documents_train=lemma[1:]
dictionary = corpora.Dictionary(
    lemma
   for lemma in lemma)
corpus = [
  dictionary.doc2bow(lemma)
  for lemma in lemma]
lsi_model = LsiModel(
```

```
        corpus=corpus,
        id2word=dictionary
        )
print(lsi_model.get_topics())
print([lsi_model[lemma] for lemma in corpus])
cosine_similarity_matrix =
similarities.MatrixSimilarity(lsi_model[corpus])
print([row for row in cosine_similarity_matrix])
vector_lsi_test = lsi_model[dictionary.doc2bow(documents_test)]
cosine_similarities_test =
cosine_similarity_matrix[vector_lsi_test]

#Penggabungan dengan DF awal
dflsa=pd.DataFrame(cosine_similarities_test,
    columns=['cosine_similarity_LSA'])
dfakhir = pd.concat([dflsa, dfhasil.reset_index()], axis=1)
dfakhir

#Visualisasi Hasil
X_LSA = dfakhir.cosine_similarity_LSA
y = dfakhir.Score1
plt.scatter(X_LSA[1:], y[1:], c=y[1:], s=50, cmap='spring')
plt.xlabel("Cosine Similarity LSA")
plt.ylabel("Skor Jawaban Murid")

#Visualisasi Antar Cosine Similarity
plt.scatter(X_LSA[1:], X[1:], c=X[1:], s=50, cmap='spring')
plt.xlabel("Cosine Similarity LSA")
plt.ylabel("Cosine Similarity BoW")

#Summary
dfakhir.groupby(by=["Score1"]).mean()
dfakhir[dfakhir['Score1']==0].describe()
dfakhir[dfakhir['Score1']==1].describe()
dfakhir[dfakhir['Score1']==2].describe()
dfakhir[dfakhir['Score1']==3].describe()
```

```
#Uji Signifikansi Rataan
import scipy
scipy.stats.mannwhitneyu(X_LSA[1:], X[1:],
use_continuity=True, alternative=None)
```

Lampiran C Algoritma Python untuk Pembelajaran Mesin

```
#impor data
X_BoW = dfakhir.cosine_similarity[1:]
X_LSA = dfakhir.cosine_similarity[1:]
y = dfakhir.Score1[1:]

#model pohon keputusan untuk fitur cosine similarity BoW
#DECISION TREE
#Impor paket
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
#Pemilahan data
X_train, X_test, y_train, y_test = train_test_split(X_BoW,
y, test_size = 0.2, random_state = 42)
clf_gini = DecisionTreeClassifier(criterion='gini',
max_depth=3, random_state=0)

# fit the model
X_train= np.array(X_train).reshape(-1, 1)
clf_gini.fit(X_train, y_train)
X_test= np.array(X_test).reshape(-1, 1)
y_pred_gini = clf_gini.predict(X_test)
from sklearn.metrics import accuracy_score
print('Model accuracy score with criterion gini index:
{0:0.4f}'. format(accuracy_score(y_test, y_pred_gini)))

#Hyperparameter Tuning
from sklearn.model_selection import GridSearchCV
params = {'max_features': ['auto', 'sqrt', 'log2'],
```

```
          'min_samples_split': [2,3,4,5,6,7,8,9,10,11,12,13,
          14,15],
          'min_samples_leaf':[1,2,3,4,5,6,7,8,9,10,11],
          'random_state':[123]}
model1 = GridSearchCV(clf_gini, param_grid=params,
n_jobs=-1)
model1.fit(X_train,y_train)
print("Best Hyper Parameters:",model1.best_params_)
prediction=model1.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print(metrics.confusion_matrix(prediction,y_test))

#Pohon Keputusan dengan Fitur Cosine Similarity LSI
X_train, X_test, y_train, y_test = train_test_split(X_LSA,
y, test_size = 0.2, random_state = 42)
clf_gini = DecisionTreeClassifier(criterion='gini',
max_depth=3, random_state=0)

# fit the model
X_train= np.array(X_train).reshape(-1, 1)
clf_gini.fit(X_train, y_train)
X_test= np.array(X_test).reshape(-1, 1)
y_pred_gini = clf_gini.predict(X_test)
from sklearn.metrics import accuracy_score
print('Model accuracy score with criterion gini index:
{0:0.4f}'. format(accuracy_score(y_test, y_pred_gini)))

from sklearn.model_selection import GridSearchCV
params = {'max_features': ['auto', 'sqrt', 'log2'],
          'min_samples_split': [2,3,4,5,6,7,8,9,10,11,12,13,
          14,15],
          'min_samples_leaf':[1,2,3,4,5,6,7,8,9,10,11],
          'random_state':[123]}
```

```
model1 = GridSearchCV(clf_gini, param_grid=params, n_jobs=-1)
model1.fit(X_train,y_train)
print("Best Hyper Parameters:",model1.best_params_)
prediction=model1.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",
metrics.confusion_matrix(prediction,y_test))

#Pohon Keputusan dengan Fitur Cosine Similarity BoW dan LSI
#DECISION TREE
from sklearn.tree import DecisionTreeClassifier
XBoWLSA=dfakhir[['cosine_similarity','cosine_similarity_LSA']]
yBoWLSA=dfakhir['Score1']
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(XBoWLSA,
yBoWLSA, test_size = 0.2, random_state = 42)
clf_gini = DecisionTreeClassifier(criterion='gini',
max_depth=3, random_state=0)
clf_gini.fit(X_train, y_train)
y_pred_gini = clf_gini.predict(X_test)

from sklearn.metrics import accuracy_score
print('Model accuracy score with criterion gini index: {0:0.4f}'.
format(accuracy_score(y_test, y_pred_gini)))

#Hyperparameter Tuning

from sklearn.model_selection import GridSearchCV
params = {'max_features': ['auto', 'sqrt', 'log2'],
          'min_samples_split':
          [2,3,4,5,6,7,8,9,10,11,12,13,14,15],
          'min_samples_leaf':[1,2,3,4,5,6,7,8,9,10,11],
          'random_state':[123]}
```

```
model1 = GridSearchCV(clf_gini, param_grid=params, n_jobs=-1)
model1.fit(X_train,y_train)
print("Best Hyper Parameters:",model1.best_params_)
prediction=model1.predict(X_test)
from sklearn import metrics
#evaluation(Accuracy)
print("Accuracy:",
metrics.accuracy_score(prediction,y_test))
#evaluation(Confusion Metrix)
print("Confusion Metrix:\n",
metrics.confusion_matrix(prediction,y_test)
```

. . . . . .

Info cetak .....

Revisi/cetak terakhir: 11 Juni 2021, pukul 04:46

Nomor halaman: i–x, 1–45 Total: 55 halaman

. . . . . .