

**IMPLEMENTASI ALGORITMA K-MEANS DENGAN
PRINCIPAL COMPONENT ANALYSIS PADA SEGMENTASI
PELANGGAN APLIKASI STARBUCKS BERDASARKAN
INDIKATOR FREQUENCY, MONETARY, DAN TENURE**

LAPORAN TUGAS AKHIR



**ALFIAN HIDAYATULLOH
5180411382**

**PROGRAM STUDI INFORMATIKA
FAKULTAS SAINS & TEKNOLOGI
UNIVERSITAS TEKNOLOGI YOGYAKARTA
YOGYAKARTA
2022**

LEMBAR PENGESAHAN

IMPLEMENTASI ALGORITMA K-MEANS DENGAN PRINCIPAL COMPONENT ANALYSIS PADA SEGMENTASI PELANGGAN APLIKASI STARBUCKS BERDASARKAN INDIKATOR FREQUENCY, MONETARY, DAN TENURE

Disusun oleh

ALFIAN HIDAYATULLOH

5180411382

Telah dipertahankan di depan Dewan Penguji
pada tanggal 12 Agustus 2022

DEWAN PENGUJI

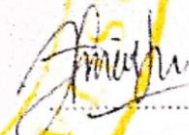
Nama dan Gelar

Jabatan

Tanda tangan Tanggal

Ahmad Tri Hidayat, S.Kom., M.Kom.
NIK. 110918164

Ketua Penguji

 25/8 2022

Yuli Asriningtias, S.Kom., M.Kom.
NIK. 110202053

Penguji I


 26/08/22

Suhirman, S.Kom., M.Kom., Ph.D.
NIK. 110698023

Penguji II

(Dosen

Pembimbing)

 26/08-2022

Yogyakarta, ... Agustus 2022

Program Studi Informatika



Dr. Elly Hic Sela, S.Si., M.Kom.
NIK. 11116086

LEMBAR PERNYATAAN

Yang bertanda tangan di bawah ini, saya

Nama : Alfian Hidayatulloh
NPM : 5180411382
Program Studi : Informatika
Program : Sarjana (S-1)
Fakultas : Sains & Teknologi

menyatakan bahwa tugas akhir dengan judul Implementasi Algoritma *K-Means* dengan *Principal Component Analysis* pada Segmentasi Pelanggan Aplikasi Starbucks Berdasarkan Indikator *Frequency*, *Monetary*, dan *Tenure* ini adalah karya ilmiah asli saya dan belum pernah dipublikasikan oleh orang lain, kecuali yang tertulis sebagai acuan dalam naskah ini dan disebutkan dalam daftar pustaka. Apabila di kemudian hari, karya saya disinyalir bukan merupakan karya asli saya, maka saya bersedia menerima konsekuensi apa yang diberikan Program Studi Informatika Fakultas Sains & Teknologi Universitas Teknologi Yogyakarta.

Demikian surat pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Yogyakarta
Pada tanggal : 27 Juli 2022

Yang menyatakan



10000
METERAL
TEMPEL
79937AJX951140109

Alfian Hidayatulloh

ABSTRAK

Starbucks adalah sebuah jaringan kedai kopi dari Amerika Serikat yang bermarkas di Seattle, Washington. Sejak era generasi milenial, persaingan bisnis kopi tidak lagi mengacu pada masalah nama *brand*, namun juga kualitas dan harga produk. Kualitas dan harga produk yang sesuai dengan *demand* pasar akan meningkatkan nama *brand* di mata konsumen dengan sendirinya. Tentu hal tersebut akan memengaruhi strategi bisnis Starbucks karena semakin banyaknya kompetitor yang bermunculan. Tujuan dari penelitian ini adalah untuk menganalisis dan mempraktikkan salah satu *brand strategy* Starbucks, yaitu segmentasi pelanggan. Dengan mengimplementasikan segmentasi pelanggan, Starbucks mampu mengetahui secara lebih baik kebutuhan dan keinginan para pelanggannya. Dengan begitu, Starbucks pun bisa menawarkan beragam jenis penawaran ataupun promosi yang sesuai dengan preferensi pelanggannya. Untuk mengetahui karakteristik tiap pelanggan, penulis merancang model FMT (*Frequency, Monetary, dan Tenure*) untuk proses *clustering* menggunakan algoritma *K-Means*. Sedangkan metode *Principal Component Analysis* (PCA) digunakan untuk mengurangi jumlah variabel (*reduksi dimensi*) tanpa menghilangkan informasi utama pada *dataset*. Berdasarkan hasil PCA, diperoleh bahwa dengan menggunakan 3 komponen dari 5 variabel, sudah mewakili sebesar 89% dari keseluruhan informasi dalam model FMT. Berdasarkan hasil proses *clustering* melalui Google Colaboratory, model mampu mengkategorikan pelanggan ke dalam empat segmen meliputi: *bronze customers*, *silver customers*, *gold customers*, dan *diamond customers*. Selain itu, hasil pengujian model menggunakan metode *Elbow* menunjukkan *Within-Cluster Sum of Squares* (WCSS), yang merupakan ukuran kedekatan antarobjek dalam suatu kluster, sebesar 979,80. Semakin kecil nilai WCSS mengindikasikan semakin optimal jumlah kluster yang dihasilkan. Namun hal tersebut dengan tetap mempertimbangkan besarnya penurunan nilai WCSS dari tiap jumlah kluster.

Kata Kunci: Segmentasi Pelanggan, Model FMT, *Principal Component Analysis*, Algoritma *K-Means*, Metode *Elbow*

ABSTRACT

Starbucks is a U.S. coffee shop chain based in Seattle, Washington. Since the era of the millennial generation, coffee business competition no longer refers to the problem of brand names, but also the quality and price of products. The quality and price of products that are in accordance with market demand will increase the brand name in the eyes of consumers by itself. Of course, this will affect Starbucks' business strategy because of the increasing number of competitors that have emerged. The purpose of this study is to analyze and practice one of Starbucks' brand strategies, namely customer segmentation. By implementing customer segmentation, Starbucks is able to better know the needs and desires of its customers. That way, Starbucks can also offer various types of offers or promotions that suit the preferences of its customers. To find out the characteristics of each customer, the authors designed an FMT (*Frequency, Monetary, and Tenure*) model for the clustering process using the *K-Means* algorithm. While the *Principal Component Analysis* (PCA) method is used to reduce the number of variables (*dimensionality reduction*) without eliminating the main information in the dataset. Based on the results of the PCA, it was obtained that using 3 components of 5 variables, it already represents 89% of the overall information in the FMT model. Based on the results of the clustering process through Google Colaboratory, the model is able to categorize customers into four segments including: *bronze customers*, *silver customers*, *gold customers*, and *diamond customers*. In addition, the test results of the model using the *Elbow* method showed *Within-Cluster Sum of Squares* (WCSS), which is a measure of proximity between objects in a cluster, of 979.80. The smaller the WCSS value indicates the more optimal the number of clusters generated. However, this still takes into account the magnitude of the decrease in the WCSS value of each cluster number.

Keywords: Customer Segmentation, FMT Model, *Principal Component Analysis*, *K-Means* Algorithm, *Elbow* Method

Mengetahui,
Ketua Program Studi Informatika



Dr. Enny Itje Sela, S.Si., M.Kom.
NIK. 111116086

Menyetujui,
Dosen Pembimbing Tugas Akhir



Suhirman, S.Kom., M.Kom., Ph.D.
NIK. 110698023

KATA PENGANTAR

Puji syukur dipanjatkan atas kehadiran Allah SWT karena dengan limpahan karunia-Nya, penulis dapat menyelesaikan laporan tugas akhir dengan judul Implementasi Algoritma *K-Means* dengan *Principal Component Analysis* pada Segmentasi Pelanggan Aplikasi Starbucks Berdasarkan Indikator *Frequency*, *Monetary*, dan *Tenure*. Penyusunan laporan tugas akhir diajukan sebagai salah satu syarat kelulusan pada Program Studi Informatika, Fakultas Sains & Teknologi, Universitas Teknologi Yogyakarta. Laporan tugas akhir ini dapat diselesaikan tidak lepas dari segala bantuan, bimbingan, dan doa dari berbagai pihak. Kemudian pada kesempatan ini, penulis ingin menyampaikan ucapan terima kasih kepada:

- a. Dr. Bambang Moertono Setiawan, M.M., C.A., Akt. selaku Rektor Universitas Teknologi Yogyakarta.
- b. Dr. Endy Marlina, S.T., M.T. selaku Dekan Fakultas Sains & Teknologi, Universitas Teknologi Yogyakarta.
- c. Dr. Enny Itje Sela, S.Si., M.Kom. selaku Ketua Program Studi Informatika, Fakultas Sains & Teknologi, Universitas Teknologi Yogyakarta.
- d. Suhirman, S.Kom., M.Kom., Ph.D. selaku Dosen Pembimbing yang telah memberikan bimbingan dan arahan kepada penulis dalam pengerjaan laporan tugas akhir ini.
- e. Wahyu Sri Utami, S.Si, M.Sc. selaku Dosen Wali S1 Informatika kelas F angkatan 2018 yang telah memberikan arahan dan nasihat kepada penulis dalam menjalani masa perkuliahan selama ini.
- f. Seluruh dosen dan karyawan Universitas Teknologi Yogyakarta, khususnya di Fakultas Sains & Teknologi yang telah memberikan ilmu dan pelayanan terbaik kepada penulis dalam menjalani masa perkuliahan selama ini.

- g. Kedua orang tua yang telah memberikan doa, semangat, dan dukungan sehingga penulis dapat menyelesaikan laporan tugas akhir ini.
- h. Teman-teman program studi S1 Informatika, khususnya pada kelas F yang telah memotivasi penulis untuk menyelesaikan laporan tugas akhir ini.
- i. Diri sendiri yang telah semangat dan berjuang melawan rasa malas dalam menjalani masa perkuliahan serta mengerjakan tugas akhir hingga tuntas.
- j. Semua pihak yang tidak bisa saya sebutkan satu per satu, yang telah memberikan bantuan baik secara langsung maupun tidak langsung sejak awal masa perkuliahan hingga terselesaikannya laporan tugas akhir ini.

Akhir kata, penulis menyadari sepenuhnya jika ada kesalahan dan kekurangan dalam penyusunan laporan tugas akhir ini. Untuk itu, saran serta kritik yang membangun dari para pembaca sangat diharapkan sebagai bahan evaluasi dan pelajaran berharga bagi penulis di masa yang akan datang. Penulis berharap, semoga laporan tugas akhir ini dapat bermanfaat bagi penulis, pengembangan ilmu pengetahuan, dan semua pihak yang membutuhkan.

Yogyakarta, 27 Juli 2022

Alfian Hidayatulloh

DAFTAR ISI

HALAMAN JUDUL	i
LEMBAR PENGESAHAN	ii
ABSTRAK	iv
ABSTRACT	iv
KATA PENGANTAR.....	v
DAFTAR ISI.....	vii
DAFTAR GAMBAR.....	ix
DAFTAR TABEL	xii
DAFTAR RUMUS	xiv
DAFTAR LAMPIRAN	xv
BAB I PENDAHULUAN.....	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah.....	4
1.4 Tujuan Penelitian	4
1.5 Manfaat Penelitian	5
1.6 Sistematika Penulisan	5
BAB II KAJIAN HASIL PENELITIAN DAN LANDASAN TEORI.....	7
2.1 Kajian Hasil Penelitian.....	7
2.2 Landasan Teori.....	12
2.2.1 Segmentasi Pelanggan.....	14
2.2.2 Analisis Klaster	15
2.2.3 <i>Exploratory Data Analysis</i>	16
2.2.4 Kemencengan Distribusi Data	16
2.2.5 Model RFMT	17
2.2.6 Normalisasi Data.....	19
2.2.7 <i>Principal Component Analysis</i>	21
2.2.8 Algoritma <i>K-Means</i>	22
2.2.9 Metode <i>Elbow</i>	23
2.2.10 Python	26
BAB III METODE PENELITIAN	31
3.1 Bahan dan Data	31
3.1.1 Data yang Diperoleh	31
3.1.2 Prosedur Pengumpulan Data	32
3.2 Aturan Bisnis (<i>Business Rule</i>).....	34
3.3 Tahapan Penelitian	36
BAB IV ANALISIS DAN PERANCANGAN SISTEM	41
4.1 Analisis Sistem yang Diusulkan	41
4.1.1 Analisis Fungsional.....	41
4.1.2 Analisis Nonfungsional.....	42
4.2 Desain Sistem.....	43
4.2.1 Desain Logis	43
4.2.2 Desain Fisik.....	48

BAB V IMPLEMENTASI DAN HASIL	53
5.1 Implementasi	53
5.1.1 Proses <i>Data Preprocessing</i>	53
5.1.2 Proses <i>Exploratory Data Analysis</i>	56
5.1.3 Proses Pembobotan FMT	65
5.1.4 Proses <i>Principal Component Analysis</i>	67
5.1.5 Proses <i>Clustering</i> dan Uji Performa	68
5.1.6 Proses Visualisasi Data	69
5.2 Hasil	71
5.2.1 Hasil <i>Data Preprocessing</i>	71
5.2.2 Hasil <i>Exploratory Data Analysis</i>	80
5.2.3 Hasil Pembobotan FMT	99
5.2.4 Hasil <i>Principal Component Analysis</i>	105
5.2.5 Hasil Proses <i>Clustering</i> dan Uji Performa	107
5.2.6 Hasil Visualisasi Data	111
BAB VI PENUTUP	122
6.1 Simpulan	122
6.2 Saran	123
DAFTAR PUSTAKA	124
LAMPIRAN	128

DAFTAR GAMBAR

Gambar 2.1 Jenis-jenis kurva kemencengan distribusi data.....	17
Gambar 2.2 Nilai k yang optimal pada metode <i>Elbow</i>	25
Gambar 3.1 Tangkapan layar beranda situs web Kaggle.com	32
Gambar 3.2 Tangkapan layar hasil pencarian situs web Kaggle.com	33
Gambar 3.3 Tangkapan layar halaman dataset pelanggan Starbucks.....	33
Gambar 3.4 Halaman <i>sign in</i> Kaggle.com.....	34
Gambar 3.5 Daftar harga produk kopi Starbucks di Indonesia	35
Gambar 3.6 Diagram alir tahap penelitian tugas akhir	36
Gambar 3.7 Diagram alir untuk tahap <i>Principal Component Analysis</i> (PCA)....	39
Gambar 3.8 Diagram alir untuk tahap <i>K-Means clustering</i>	40
Gambar 4.1 <i>Flowchart</i> sistem yang akan dibangun	44
Gambar 4.2 <i>Use case diagram</i> sistem yang akan dibangun	45
Gambar 4.3 <i>Activity diagram</i> sistem yang akan dibangun	47
Gambar 4.4 <i>Class diagram</i> sistem yang akan dibangun	50
Gambar 5.1 Hasil proses <i>read dataset portfolio.json</i>	71
Gambar 5.2 Hasil proses <i>read dataset profile.json</i>	71
Gambar 5.3 Hasil proses <i>read dataset transcript.json</i>	71
Gambar 5.4 Hasil proses dari fungsi <i>extract_from_iterable_col()</i> untuk dataset <i>portfolio.json</i>	75
Gambar 5.5 Hasil proses dari fungsi <i>extract_from_iterable_col()</i> untuk dataset <i>transcript.json</i>	75
Gambar 5.6 Hasil konversi tipe data pada dataset <i>portfolio.json</i>	76
Gambar 5.7 Hasil konversi tipe data pada dataset <i>profile.json</i>	77
Gambar 5.8 Hasil proses konversi tipe data pada dataset <i>transcript.json</i>	77
Gambar 5.9 Hasil proses pencarian data duplikat di dataset <i>transcript.json</i>	80
Gambar 5.10 Hasil penggabungan <i>string</i> dan perhitungan nilai <i>expire time</i> untuk data penawaran <i>reward</i>	82

Gambar 5.11 Hasil visualisasi data (<i>bar plot</i>) untuk jumlah penawaran <i>reward</i>	84
Gambar 5.12 Hasil penggabungan <i>string</i> dan perhitungan nilai <i>expire time</i> untuk data penawaran informasional.....	85
Gambar 5.13 Hasil visualisasi data untuk jumlah penawaran informasional.....	87
Gambar 5.14 Hasil visualisasi untuk data tingkat penyelesaian penawaran	88
Gambar 5.15 Histogram untuk distribusi demografi pengguna	90
Gambar 5.16 Hasil pemotongan kuantil pada <i>dataset profile.json</i>	91
Gambar 5.17 Hasil visualisasi data untuk pendapatan pelanggan berdasarkan kelompok usia dan jenis kelamin	91
Gambar 5.18 Jumlah total pembelian berdasarkan demografi pengguna.....	92
Gambar 5.19 <i>Line plot</i> pola demografi pengguna pada tiap <i>spend group</i>	95
Gambar 5.20 <i>Heatmap</i> untuk korelasi antarvariabel demografi pengguna	96
Gambar 5.21 Tingkat penyelesaian penawaran berdasarkan <i>gender</i>	97
Gambar 5.22 Tingkat penyelesaian penawaran berdasarkan <i>age group</i>	97
Gambar 5.23 Tingkat penyelesaian penawaran berdasarkan <i>income group</i>	97
Gambar 5.24 Tingkat penyelesaian penawaran berdasarkan <i>signup year</i>	98
Gambar 5.25 Tingkat penyelesaian penawaran berdasarkan <i>spend group</i>	98
Gambar 5.26 Hasil visualisasi untuk distribusi data model FMT	103
Gambar 5.27 Hasil visualisasi untuk jumlah komponen PCA yang optimal	105
Gambar 5.28 Hasil visualisasi untuk distribusi data komponen PCA.....	106
Gambar 5.29 Hasil visualisasi untuk korelasi data komponen PCA	107
Gambar 5.30 Hasil visualisasi untuk jumlah klaster yang optimal	108
Gambar 5.31 Data hasil penerapan fungsi <i>segment_means()</i>	109
Gambar 5.32 Hasil visualisasi data untuk perhitungan nilai WCSS	110
Gambar 5.33 <i>Heatmap</i> untuk tingkat korelasi variabel pada tiap klaster.....	112
Gambar 5.34 <i>Snake plot</i> untuk tingkat korelasi variabel pada tiap klaster.....	112
Gambar 5.35 <i>Scatter plot</i> dua dimensi untuk hubungan <i>comp1</i> dengan <i>comp2</i>	114
Gambar 5.36 <i>Scatter plot</i> dua dimensi untuk hubungan <i>comp1</i> dengan <i>comp3</i> serta hubungan <i>comp2</i> dengan <i>comp3</i>	114
Gambar 5.37 <i>Scatter plot</i> tiga dimensi untuk <i>comp1</i> , <i>comp2</i> , dan <i>comp3</i>	115
Gambar 5.38 Hasil visualisasi untuk jumlah pengguna tiap klaster.....	115



Gambar 5.39 Hasil visualisasi data untuk klaster <i>gold customers</i>	116
Gambar 5.40 Hasil visualisasi data untuk klaster <i>silver customers</i>	118
Gambar 5.41 Hasil visualisasi data untuk klaster <i>diamond customers</i>	119
Gambar 5.42 Hasil visualisasi data untuk klaster <i>bronze customers</i>	120



DAFTAR TABEL

Tabel 2.1 Hasil penelitian sebelumnya.....	10
Tabel 3.1 Variabel untuk proses transformasi data	37
Tabel 5.1 Deskripsi terkait tipe-tipe penawaran.....	57
Tabel 5.2 <i>Lambda expression with conditional statement</i> untuk mengetahui waktu kedaluwarsa penawaran <i>reward</i>	58
Tabel 5.3 Penjelasan dari tiap grup penawaran <i>reward</i>	59
Tabel 5.4 <i>Lambda expression with conditional statement</i> untuk mengetahui waktu kedaluwarsa penawaran informasional.....	60
Tabel 5.5 Penjelasan dari tiap grup penawaran informasional.....	61
Tabel 5.6 Rentang data demografi pengguna Starbucks	63
Tabel 5.7 Variabel pada fungsi agregat	64
Tabel 5.8 Operasi matematis untuk pembobotan FMT	66
Tabel 5.9 Hasil proses <i>data profiling</i> untuk <i>dataset portfolio.json</i>	72
Tabel 5.10 Hasil proses <i>data profiling</i> untuk <i>dataset profile.json</i>	72
Tabel 5.11 Hasil proses <i>data profiling</i> untuk <i>dataset transcript.json</i>	73
Tabel 5.12 Jumlah dan persentase data yang kosong di <i>dataset profile.json</i>	74
Tabel 5.13 Hasil proses pengecekan korelasi data yang kosong.....	74
Tabel 5.14 Dimensi <i>dataset</i> setelah penghapusan data yang kosong.....	74
Tabel 5.15 Hasil proses dari fungsi <i>get_unique_values()</i>	75
Tabel 5.16 Jumlah dan persentase data yang kosong di <i>dataset transcript.json</i> ..	78
Tabel 5.17 Hasil proses iterasi variabel <i>event</i> di <i>dataset transcript.json</i>	78
Tabel 5.18 Hasil proses konversi tipe data (lanjutan)	79
Tabel 5.19 Jumlah dan persentase penawaran berdasarkan tipe	81
Tabel 5.20 Jumlah penawaran berdasarkan ID penawaran	81
Tabel 5.21 Jumlah dan persentase penawaran <i>reward</i> pada tiap grup	83
Tabel 5.22 Jumlah dan persentase penawaran informasional pada tiap grup.....	86
Tabel 5.23 Tingkat penyelesaian penawaran pada tiap tipe penawaran.....	88
Tabel 5.24 Interpretasi untuk hasil visualisasi data pada Gambar 5.64	92

Tabel 5.25 Interpretasi untuk hasil visualisasi data pada Gambar 5.65	93
Tabel 5.26 Interpretasi untuk hasil visualisasi data pada Gambar 5.65 (lanjutan).....	93
Tabel 5.27 Hasil penerapan fungsi agregat pada <i>spend group</i>	94
Tabel 5.28 Hasil proses <i>data profiling</i> untuk <i>dataset profile</i> (lanjutan)	100
Tabel 5.29 Hasil proses <i>data profiling</i> untuk <i>dataset transcript</i> (lanjutan)	100
Tabel 5.30 Hasil proses <i>data profiling</i> untuk <i>dataset offers</i>	100
Tabel 5.31 Hasil proses <i>data profiling</i> untuk <i>dataset offers</i> (lanjutan).....	101
Tabel 5.32 Sekilas hasil pembobotan model untuk variabel FMT	102
Tabel 5.33 Analisis statistik sederhana untuk model variabel FMT	102
Tabel 5.34 Sekilas hasil pemotongan kuantil pada model FMT	104
Tabel 5.35 Hasil normalisasi data menggunakan <i>Min-Max Normalization</i>	104
Tabel 5.36 Sekilas hasil penerapan reduksi dimensi dengan PCA.....	106
Tabel 5.37 Hasil penamaan klaster dalam <i>string mapping</i>	109
Tabel 5.38 Hasil perhitungan dan selisih nilai WCSS dari tiap klaster	111
Tabel 5.39 Informasi statistik untuk klaster <i>gold customers</i>	116
Tabel 5.40 Informasi statistik untuk klaster <i>silver customers</i>	117
Tabel 5.41 Informasi statistik untuk klaster <i>diamond customers</i>	119
Tabel 5.42 Informasi statistik untuk klaster <i>bronze customers</i>	120

DAFTAR RUMUS

Rumus 2.1 <i>Coefficient of Skewness</i>	16
Rumus 2.2 <i>Min-Max Normalization</i>	20
Rumus 2.3 <i>Z-score Normalization</i>	20
Rumus 2.4 <i>Decimal Scaling Normalization</i>	20
Rumus 2.5 <i>Sigmoidal Normalization</i>	21
Rumus 2.6 Rata-rata semua sampel data	21
Rumus 2.7 Normalisasi pada setiap data	21
Rumus 2.8 Matriks kovariansi	22
Rumus 2.9 Nilai <i>Eigenvalue</i>	22
Rumus 2.10 Nilai <i>Eigenvector</i>	22
Rumus 2.11 Urutan nilai <i>Eigenvector</i>	22
Rumus 2.12 <i>Principal Component</i>	22
Rumus 2.13 Transformasi untuk memperoleh data PCA	22
Rumus 2.14 Nilai titik pusat klaster (<i>centroid</i>)	23
Rumus 2.15 <i>Euclidean Distance</i>	23
Rumus 2.16 <i>Within-Cluster Sum of Squares</i>	24
Rumus 2.17 <i>Between-Cluster Sum of Squares</i>	24
Rumus 2.18 <i>Total-Cluster Sum of Squares</i>	25



DAFTAR LAMPIRAN

Lampiran 1 Data-data yang diperoleh	128
Lampiran 2 Hasil tahap normalisasi data	131
Lampiran 3 Hasil tahap <i>Principal Component Analysis</i> (PCA).....	132
Lampiran 4 Hasil tahap <i>K-Means clustering</i>	133
Lampiran 5 Struktur <i>dataset</i> dalam format JSON.....	134
Lampiran 6 <i>Source code</i> sistem segmentasi pelanggan Starbucks	144



BAB I

PENDAHULUAN

1.1 Latar Belakang

Hingga saat ini, konsumsi kopi di Indonesia didorong oleh semakin bertambahnya kedai kopi yang bermunculan di daerah perkotaan. Ratusan kedai kopi baru dengan konsep *Ready-To-Drink* (RTD) terus bermunculan di pusat-pusat perbelanjaan dalam beberapa tahun terakhir. Kedai kopi baru tersebut juga menawarkan beragam produk kopi yang diminati oleh berbagai kalangan dari kelas menengah hingga kelas atas. Berdasarkan data Badan Pusat Statistik (2018), pesatnya pertumbuhan kedai kopi di Indonesia menyebabkan setiap kedai kopi harus menyusun strategi pemasaran yang tepat agar dapat bertahan di industri. Starbucks sebagai sebuah kedai kopi kekinian yang berasal dari Washington, Amerika Serikat, kini telah menyebar ke seluruh negara di dunia. Sebagai salah satu pelopor kedai kopi kekinian, Starbucks memiliki tanggung jawab untuk menjaga dan meningkatkan kepercayaan pelanggan agar dapat terus berkembang tiap tahunnya. Maka dari itu, Starbucks perlu mengetahui segmen pelanggan sehingga dapat menawarkan beragam produk kopi sesuai dengan minat pelanggan. Hal tersebut dapat diaplikasikan melalui proses segmentasi dengan mengelompokkan pelanggan yang memiliki kesamaan karakteristik ke dalam segmen tertentu.

Analisis kluster adalah suatu teknik multivariat dalam statistika yang berguna untuk mengelompokkan obyek-obyek yang memiliki karakteristik yang serupa ke dalam kelompok yang lebih kecil. Secara umum, analisis kluster dibedakan menjadi dua, yaitu metode hierarki (*hierarchical clustering*) dan metode non-hierarki (*non-hierarchical clustering*). Metode hierarki dilakukan tanpa menentukan jumlah kelompok terlebih dahulu. Beberapa metode yang biasa digunakan dalam analisis kluster hierarki antara lain *Single Linkage*, *Complete Linkage*, *Average Linkage*, *Ward's Method*, dan *Centroid Method*. Metode non-

hierarki digunakan jika banyaknya klaster sudah diketahui. Biasanya metode ini dipakai untuk mengelompokkan data yang berukuran besar. Inti dari metode non-hierarki ini adalah bagaimana memilih pusat awal klaster yang akan berpengaruh besar terhadap hasil analisis klaster nantinya. Beberapa metode yang termasuk dalam metode non-hierarki meliputi *K-Means*, *K-Medoids*, *Density Estimation*, dan *Gaussian Mixture* (Akhyar, S., 2017).

Pada segmentasi pelanggan ini, diterapkan algoritma *K-Means clustering* yang merupakan salah satu metode pada ilmu *data mining* untuk mengelompokkan sejumlah data menjadi kelompok-kelompok tertentu. Algoritma *K-Means* termasuk dalam teknik partisi yang membagi atau memisahkan obyek ke k daerah bagian yang terpisah. Pada algoritma *K-Means*, setiap obyek harus masuk ke dalam klaster tertentu, tetapi dalam satu tahapan berikutnya obyek dapat berpindah ke klaster lain (Kurniawati, I.Y., 2018). *K-Means clustering* dipilih karena mampu menghasilkan klaster yang optimal dan metode tersebut lebih sederhana dibandingkan dengan metode non-hierarki lainnya. Namun algoritma *K-Means* juga memiliki beberapa kelemahan, salah satunya yaitu penentuan pusat awal klaster. Hasil klaster yang terbentuk dari algoritma *K-Means* sangat bergantung pada inisiasi nilai pusat awal klaster yang diberikan. Hal ini menyebabkan hasil klaster berupa solusi yang sifatnya *local optimal*. Algoritma *K-Means clustering* menjadi bermasalah ketika diterapkan pada *dataset* berdimensi tinggi, yaitu kualitas hasil klaster yang menurun dan juga waktu komputasi yang lama. Permasalahan ini dapat diatasi dengan melakukan reduksi dimensi (Izzuddin, A., 2015).

Principal Component Analysis (PCA) adalah salah satu metode reduksi dimensi yang banyak digunakan dalam statistika multivariat. Tujuan dari PCA adalah mereduksi dimensi (variabel) data yang tinggi menjadi dimensi data yang lebih rendah dengan risiko kehilangan informasi yang sangat kecil. Oleh karena itu, untuk meningkatkan efisiensi, diterapkan metode PCA pada *dataset* asli sehingga variabel yang digunakan dari *dataset* asli hanya variabel yang berkorelasi. Sebelum menerapkan metode PCA, *dataset* perlu dinormalisasi sehingga setiap variabel dengan domain yang lebih besar tidak akan mendominasi

variabel dengan domain yang lebih kecil (Dash, B. dkk., 2010). Selain itu, metode *Elbow* juga akan dimanfaatkan dalam penelitian ini sebagai metode evaluasi hasil segmentasi.

Pada penelitian kali ini, penulis menggunakan studi kasus kedai kopi Starbucks yang memiliki sasaran pelanggan *Business-to-Customer* (B2C). Pencatatan data transaksi maupun data pelanggan dalam perusahaan ini sudah dilakukan secara komputerisasi. Namun data transaksi dan data pelanggan belum dimanfaatkan secara maksimal oleh perusahaan. Contoh pemanfaatan data tersebut yaitu dapat diolah untuk mengetahui pelanggan mana yang akan mendapatkan perlakuan khusus. Misalnya dengan memberikan diskon pada tiap transaksi terhadap pelanggan tersebut. Hal ini dilakukan agar perusahaan dapat mempertahankan hubungan yang baik terhadap pelanggan yang memiliki loyalitas tinggi. Pelanggan tersebut perlu dipertahankan karena sangat berpengaruh terhadap pertumbuhan laba perusahaan. Untuk itu, perlu dilakukan proses segmentasi dengan mengelompokkan pelanggan yang memiliki kesamaan karakteristik ke dalam klaster tertentu. Segmentasi dapat dilakukan dengan menerapkan algoritma *K-Means clustering* yang melibatkan beberapa variabel meliputi frekuensi transaksi (*frequency*), total transaksi (*monetary*), dan lama waktu menjadi pelanggan (*tenure*) pada periode tertentu. Berdasarkan penjabaran di atas, maka penulis memilih judul Implementasi Algoritma *K-Means* dengan *Principal Component Analysis* pada Segmentasi Pelanggan Aplikasi Starbucks Berdasarkan Indikator *Frequency*, *Monetary*, dan *Tenure*.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan di atas, berikut merupakan rumusan masalah yang menjadi acuan dalam laporan tugas akhir ini:

- a. Apakah algoritma *K-Means* dengan *Principal Component Analysis* mampu dalam melakukan segmentasi terhadap data pelanggan Starbucks?
- b. Bagaimana hasil normalisasi data dan *Principal Component Analysis* yang dilakukan sebelum melakukan proses *clustering*?

- c. Berapa jumlah klaster optimal yang dihasilkan oleh algoritma K-Means dan apa saja nama yang diberikan pada masing-masing klaster?
- d. Bagaimana performa algoritma *K-Means* dalam melakukan segmentasi terhadap data pelanggan Starbucks?

1.3 Batasan Masalah

Pada pengerjaan laporan tugas akhir ini, terdapat beberapa batasan mengenai area penelitian yang dilakukan antara lain:

- a. Studi kasus yang digunakan yaitu data pelanggan Starbucks yang diperoleh dari situs web Kaggle.com.
- b. Data yang digunakan, yaitu data pelanggan Starbucks dalam rentang waktu pendaftaran tertentu (29 Maret 2013 s/d 26 Juli 2018).
- c. Variabel utama yang digunakan dalam tugas akhir ini adalah frekuensi transaksi (*frequency*), total transaksi (*monetary*), dan lama waktu menjadi pelanggan (*tenure*).
- d. Metode normalisasi data yang digunakan dalam penelitian untuk mengatasi distribusi data yang tidak simetris adalah *MinMax Normalization*.
- e. Metode reduksi dimensi data yang digunakan untuk mengatasi *curse of dimensionality problem* adalah *Principal Component Analysis* (PCA).
- f. Algoritma yang digunakan dalam penelitian untuk melakukan segmentasi data pelanggan adalah algoritma *K-Means*.
- g. Metode yang digunakan dalam penelitian untuk menguji performa algoritma *K-Means* adalah metode *Elbow*.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah merancang model FMT (*Frequency, Monetary, Tenure*) dan menggunakan model tersebut ke dalam segmentasi pelanggan berbasis algoritma *K-Means*. Selain itu, penulis mendeskripsikan hasil karakteristik tiap segmen pelanggan Starbucks ke dalam bentuk visualisasi data (grafik) agar lebih mudah dipahami oleh para pembaca.

1.5 Manfaat Penelitian

Penelitian ini diharapkan memberikan manfaat yaitu dapat membantu Starbucks dalam merencanakan strategi pemasaran untuk mempertahankan loyalitas pelanggan sesuai dengan minat dan perilaku tiap segmennya. Selain itu, diharapkan dapat menjadi referensi mengenai proses perancangan model FMT dan implementasi dengan algoritma *K-Means* pada data perusahaan.

1.6 Sistematika Penulisan

Sistematika penulisan laporan tugas akhir disusun sesuai dengan format yang telah ditentukan. Penulisan ini dibagi menjadi lima bab dan setiap bab terdiri dari beberapa sub-bab yang saling berhubungan satu dengan yang lainnya. Sistematika penulisan akan mencakup hal-hal sebagai berikut.

Bab I Pendahuluan

Bab ini memberikan penjelasan terkait latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penelitian.

Bab II Kajian Hasil Penelitian dan Landasan Teori

Pada bab ini, akan dijelaskan tentang referensi-referensi yang terkait dan penjelasan teori-teori tentang *data mining*, segmentasi pelanggan, analisis klaster, *Exploratory Data Analysis* (EDA), kemencengan distribusi data, model RFMT, normalisasi data, *Principal Component Analysis* (PCA), algoritma *K-Means*, dan metode *Elbow*.

Bab III Metode Penelitian

Bab ini menyajikan secara lengkap setiap langkah eksperimen yang dilakukan dalam penelitian meliputi data yang diperoleh, prosedur pengumpulan data, alat pengumpulan data, aturan bisnis (*business rule*), dan tahapan penelitian (dalam bentuk diagram).

Bab IV Analisis dan Desain Sistem

Pada bagian analisis dan desain sistem ini, akan dijelaskan proses yang penulis usulkan dari pengumpulan data, praproses dan pembersihan data, *Exploratory Data Analysis* (EDA), pembobotan model FMT, *Principal Component Analysis* (PCA), segmentasi pelanggan menggunakan algoritma *K-Means*, hingga menghasilkan luaran berupa jumlah kluster yang optimal menggunakan metode *Elbow* yang sesuai dengan kebutuhan perusahaan. Selain itu, pada bab ini juga akan dijelaskan alur proses sistem dalam bentuk grafik UML (*Unified Modelling Language*) seperti *use case diagram*, *activity diagram*, dan *class diagram*.

Bab V Penutup

Pada bab ini, dijelaskan mengenai kesimpulan yang diperoleh dari serangkaian pengerjaan tugas akhir yang telah dilakukan, serta memberikan beberapa saran sebagai pengembangan terhadap penelitian pada topik segmentasi pelanggan.

BAB II

KAJIAN HASIL PENELITIAN DAN LANDASAN TEORI

2.1 Kajian Hasil Penelitian

Dalam melakukan penelitian ini, penulis mengacu pada beberapa penelitian yang telah dilakukan sebelumnya, antara lain penelitian oleh Angelie, A.V. (2017) yang menjelaskan segmen pelanggan PT. Bina Adidaya Surabaya berdasarkan hasil *clustering* dengan model RFM dan menampilkan hasil *clustering* ke dalam visualisasi data berbasis web. Metode yang digunakan dalam segmentasi pelanggan yaitu algoritma *K-Means* dan *Elbow Method* untuk membantu menemukan jumlah kluster k dalam penerapan *K-Means*. Variabel yang digunakan untuk melakukan segmentasi pelanggan yaitu model RFM (*Recency, Frequency, Monetary*). Penerapan analisis RFM dan algoritma *K-Means* menghasilkan dua macam kelompok yaitu menjadi dua segmen (hasil pertama) dan tiga segmen (hasil kedua). Namun hasil kedua dengan tiga segmen pelanggan merupakan pembagian yang terbaik. Hasil *clustering* dengan tiga segmen telah diuji performanya menggunakan tiga metode, yaitu performa berdasarkan *Sum of Squared Errors* sebesar 3.08, *Dunn Index* sebesar 0.25, dan *Connectivity* sebesar 7.89.

Adapun referensi lain yaitu penelitian oleh Savitri, A.D. (2018) yang mana luaran dari penelitian ini adalah visualisasi dasbor dengan beberapa grafik yang memuat hasil segmentasi pelanggan berdasarkan nilai RFM pada layanan-layanan yang diberikan oleh klinik kecantikan Belle Crown Malang. Metode yang digunakan dalam segmentasi pelanggan yaitu dengan algoritma *K-Means*. Variabel yang digunakan untuk melakukan segmentasi pelanggan yaitu model RFM (*Recency, Frequency, Monetary*). Selain itu, metode *Elbow* digunakan untuk membantu dalam menentukan jumlah segmen secara intuitif dalam penerapan algoritma *K-Means*. Data yang digunakan dalam penelitian ini adalah riwayat transaksi pelanggan pada periode Mei hingga Oktober 2017 sejumlah 21.513 transaksi dan 4716 pelanggan. Pada penelitian ini menghasilkan dua segmen

(hasil pertama) dan tiga segmen (hasil kedua). Analisis berdasarkan nilai RFM menunjukkan bahwa segmen pertama merupakan kelompok pelanggan yang *profitable* karena memiliki nilai RFM yang lebih besar dibanding segmen lainnya.

Pada penelitian selanjutnya oleh Adiana, B.E. dkk. (2018) di mana mereka melakukan pengelompokan pelanggan yang kemudian akan digunakan untuk mengetahui tingkat loyalitas pelanggan dan berhubungan dengan strategi pemasaran yang tepat untuk setiap pelanggan. Segmentasi pelanggan ini digunakan untuk membantu menganalisis data transaksi di Usaha Kecil Menengah (UKM) UD Gemilang Kencana, serta dikembangkan menggunakan algoritma *K-Means* dan model RFM. *Davies Bouldin Index* (DBI) digunakan untuk menemukan jumlah kluster k yang optimal. Hasilnya kelompok pelanggan yang terbentuk ada tiga kelompok dengan kelompok pertama berjumlah 30 pelanggan masuk dalam kategori *typical customer*, kelompok kedua terdapat 8 pelanggan yang masuk dalam kategori *superstar customer*, dan kelompok ketiga berjumlah 89 pelanggan pada kategori *dormant customer*.

Dalam penelitian yang dilakukan oleh Harani, N.H. dkk. (2020) yang menjelaskan analisis karakteristik pelanggan sebagai dasar penetapan segmentasi pelanggan dan *customer profiling* pelanggan produk *digital service add ons* Indihome menggunakan algoritma *K-Means*. Penentuan jumlah kluster terbaik dilakukan menggunakan metode *Elbow* dan diperoleh nilai $k = 3$, sehingga data pelanggan dikelompokkan ke dalam tiga segmen. Pengolahan data pelanggan dibagi menjadi tiga simulasi dengan persentase *data train* dan *data test* secara berurutan yaitu 80% : 20%, 70% : 30%, dan 50% : 50%. Data yang digunakan berjumlah 1392 *record* sebagai populasi di mana data tersebut akan digunakan untuk mencari karakteristik setiap data tersebut. Evaluasi kluster dilakukan menggunakan metode *Silhouette Index*, *Davies Bouldin Index*, dan *Calinski Harabasz Index*. Hasil dari penelitian menunjukkan bahwa simulasi ketiga merupakan simulasi terbaik berdasarkan evaluasi kluster di mana *customer profiling* dilihat dengan menganalisis anggota masing-masing kluster dari simulasi ketiga di mana kluster 0 memiliki anggota 396 pelanggan dengan kategori pelanggan yang memberikan keuntungan terbesar bagi perusahaan, kluster 1

memiliki anggota 286 pelanggan dengan kategori pelanggan yang tanpa disadari memiliki potensi besar dalam memberikan keuntungan bagi perusahaan, dan klaster 2 memiliki anggota 14 pelanggan dengan kategori pelanggan yang memberikan keuntungan lebih sedikit daripada biaya untuk memberikan pelayanan.

Acuan penelitian oleh Hadi, F. dkk. (2017) yang mana untuk mengetahui perilaku konsumen dan menerapkan strategi pemasaran yang tepat sehingga mendatangkan keuntungan bagi PT. Herbal Penawar Alwahidah Indonesia. Langkah-langkahnya dimulai dari mengumpulkan data transaksi pelanggan, selanjutnya *data preprocessing* dengan memilih data yang dibutuhkan, kemudian mencari *customer value* sesuai parameter RFM (*Recency, Frequency, Monetary*). Proses selanjutnya adalah dengan mengelompokkan pelanggan menggunakan algoritma *K-Means*. Terakhir, pelanggan akan diberikan rekomendasi strategi *marketing* berdasarkan karakteristik dari masing-masing klaster. Hasil yang didapatkan dari studi kasus PT. Herbal Penawar Alwahidah Indonesia menunjukkan bahwa segmen pelanggan yang terbentuk adalah empat klaster yaitu klaster pertama berjumlah 4 pelanggan, klaster kedua berjumlah 339 pelanggan, klaster ketiga berjumlah 200 pelanggan, dan klaster keempat berjumlah 8 pelanggan. Rincian penelitian yang dijadikan referensi oleh penulis ditunjukkan pada Tabel 2.1.

Tabel 2.1 Hasil penelitian sebelumnya

No.	Judul Penelitian	Penulis & Tahun	Keterkaitan
1.	Segmentasi Pelanggan Menggunakan <i>Clustering K-Means</i> dan Model RFM (Studi Kasus: PT Bina Adidaya Surabaya)	Annisa Veronika Angelie; 2017	Penelitian ini digunakan untuk melihat pengelompokan terkait pelanggan yang menggunakan variabel dari hasil analisis RFM dan algoritma <i>K-Means</i> . Model RFM yang digunakan berkaitan dengan model FMT yang digunakan pada penelitian oleh penulis.
2.	Segmentasi Pelanggan Menggunakan Metode <i>K-Means Clustering</i> Berdasarkan Model RFM Pada Klinik Kecantikan (Studi Kasus: Belle Crown Malang)	Aulia Dewi Savitri; 2018	Penelitian ini digunakan untuk melihat pengelompokan terkait pelanggan yang menggunakan variabel dari hasil analisis RFM dan algoritma <i>K-Means</i> . Metode <i>Elbow</i> yang digunakan untuk menentukan jumlah klaster yang optimal berkaitan dengan metode evaluasi yang digunakan dalam penelitian oleh penulis.

Lanjutan Tabel 2.1 Hasil penelitian sebelumnya

No.	Judul Penelitian	Penulis & Tahun	Keterkaitan
3.	Analisis Segmentasi Pelanggan Menggunakan Kombinasi RFM Model dan Teknik <i>Clustering</i>	Beta Estri Adiana, Indah Soesanti, dan Adhistya Permanasari; 2018	Penelitian ini digunakan untuk melihat pengelompokan terkait pelanggan yang menggunakan variabel dari hasil analisis RFM dan algoritma <i>K-Means</i> . Model RFM yang digunakan berkaitan dengan model FMT yang digunakan pada penelitian oleh penulis.
4.	Segmentasi Pelanggan Produk <i>Digital Service</i> Indihome Menggunakan Algoritma <i>K-Means</i> Berbasis Python	Nisa Hanum Harani, Cahyo Prianto, dan Fikri Aldi Nugraha; 2020	Penelitian ini digunakan untuk melihat pengelompokan terkait pelanggan yang menggunakan variabel dari hasil analisis RFM, dan algoritma <i>K-Means</i> . Metode <i>Elbow</i> yang digunakan untuk menentukan jumlah klaster yang optimal berkaitan dengan metode evaluasi yang digunakan dalam penelitian oleh penulis.

Lanjutan Tabel 2.1 Hasil penelitian sebelumnya

No.	Judul Penelitian	Penulis & Tahun	Keterkaitan
5.	Penerapan <i>K-Means Clustering</i> Berdasarkan RFM Mofek Sebagai Pemetaan dan Pendukung Strategi Pengelolaan Pelanggan (Studi Kasus: PT Herbal Penawar Alwahidah Indonesia Pekanbaru)	Fakhri Hadi, Dini Octari Rahmadia, Ferdian Hadi Nugraha, Nada Putri Bulan, Mustakim, dan Siti Monalisa; 2017	Penelitian ini digunakan untuk melihat pengelompokan terkait pelanggan yang menggunakan variabel dari hasil analisis RFM, dan algoritma <i>K-Means</i> . Model RFM yang digunakan berkaitan dengan model FMT yang digunakan pada penelitian oleh penulis.

2.2 Landasan Teori

Penggalian data atau yang biasa disebut dengan *data mining* merupakan proses eksplorasi dan analisis untuk menemukan pola yang berarti dan aturan pada data yang berukuran besar (Berry, M. dan Linoff, G., 2000). Data tersebut berasal dalam suatu basis data, *data warehouse*, atau tempat penyimpanan data lainnya (Han, J. dkk. 2012). Pengetahuan yang digunakan untuk menyajikan pengetahuan kepada pengguna. Proses penggalian data biasanya dilakukan dengan menggunakan teknik pengenalan pola seperti statistik dan matematika. Penggalian data sebagai langkah penting dalam proses penemuan pengetahuan (*knowledge discovery process*). Urutan proses penemuan pengetahuan yaitu seperti langkah-langkah berikut (Han, J. dkk. 2012).

- a. Pembersihan data merupakan proses awal yang dilakukan dalam penemuan pengetahuan dengan menghapus data yang dianggap mengganggu penelitian (*noise*) dan data yang tidak konsisten.

- b. Integrasi data merupakan proses inti yang perlu dilakukan karena beberapa sumber data harus digabungkan menjadi satu.
- c. Pemilihan data yaitu proses memperoleh data yang relevan pada basis data dengan memilih data yang dianggap relevan dengan penelitian.
- d. Transformasi data yaitu proses mengubah format data sesuai keperluan dalam proses penemuan pengetahuan, di mana data ditransformasikan dan dikonsolidasikan ke dalam bentuk yang sesuai dengan melakukan ringkasan (agregasi) pada data hasil proses transformasi.
- e. Penggalan data merupakan proses penting dan utama karena metode sistem cerdas akan diterapkan untuk mengekstraksi pola data.
- f. Evaluasi pola untuk mengidentifikasi pola yang benar-benar menarik dan mewakili pengetahuan berdasarkan ukuran *interestingness*.
- g. Presentasi pengetahuan merupakan proses visualisasi dan representasi data.

Tugas dalam penggalan data terbagi menjadi dua kategori, yaitu penggalan data langsung (*directed*) dan penggalan data tidak langsung (*undirected*) (Berry, M. dan Linoff, G., 2000). Penggalan data langsung tujuannya adalah menggunakan data yang tersedia untuk membangun sebuah model yang menggambarkan salah satu variabel tertentu yang menarik dalam data tersebut. Terdapat tiga tugas yang termasuk dalam penggalan data langsung yaitu *classification*, *estimation*, dan *prediction*. Penggalan data tidak langsung berarti tidak terdapat variabel yang dipilih sebagai target karena tujuannya untuk membangun beberapa hubungan pada semua variabel. Beberapa tugas yang termasuk dalam penggalan data tidak langsung adalah *clustering*, *affinity grouping* atau *association rules*, *description*, dan *visualization*. Pada penelitian ini, proses penggalan data digunakan untuk membantu pengolahan data transaksi pelanggan Starbucks. Tugas penggalan data pada penelitian ini termasuk pada penggalan data tidak langsung yaitu *clustering* karena bertujuan untuk membuat kelompok dengan menghubungkan semua variabel pada data yang ada.

2.2.1 Segmentasi Pelanggan

Segmentasi merupakan salah satu cara untuk memiliki komunikasi yang lebih dengan pelanggan. Tujuan dari segmentasi adalah untuk menyesuaikan produk, jasa, dan pesan pemasaran untuk setiap segmen (Berry, M. dan Linoff, G., 2000). Proses segmentasi menempatkan pelanggan sesuai dengan karakteristik kelompok pelanggan yang serupa. Segmentasi pelanggan merupakan langkah persiapan untuk mengklasifikasikan setiap pelanggan sesuai dengan kelompok pelanggan yang sudah ditetapkan (Jansen, S.M.H., 2007). Karakteristik pelanggan dapat direpresentasikan oleh beberapa kategori variabel yang terkait dengan pengelompokan, seperti berikut ini (Berry, M. dan Linoff, G., 2000):

- a. Demografi meliputi: usia, jenis kelamin, jumlah gaji atau pemasukan, profesi, pendidikan, status sosial ekonomi, agama, dan kewarganegaraan.
- b. Psikografi meliputi: kepribadian, gaya hidup, nilai-nilai, dan sikap.
- c. Perilaku meliputi: manfaat yang dicari, status pembelian, tingkat penggunaan produk, dan frekuensi pembelian.
- d. Geografi meliputi: negara, provinsi, kota, kode pos, dan iklim.

Skema segmentasi yang berbeda dapat dikembangkan menurut tujuan bisnis yang spesifik dari organisasi. Segmentasi umumnya digunakan melalui riset data pasar untuk mendapatkan wawasan tentang sikap pelanggan, keinginan, pandangan, preferensi, dan opini tentang perusahaan dan kompetisi (Berry, M. dan Linoff, G., 2000). Segmentasi pelanggan berdasarkan pada riset pasar dan demografi membutuhkan pemahaman karakteristik semua pelanggan agar lebih efektif mengetahui segmen apa yang menjadi menarik pelanggan. Penggalan data dapat mengembangkan segmentasi pelanggan yang juga mengidentifikasi segmentasi pada perilaku pelanggan (Tsiptsis, K. dan Chorianopoulos, A., 2010). Selain data penelitian eksternal, data transaksi dan pembayaran pelanggan juga dapat digunakan untuk mendapatkan wawasan tentang perilaku pelanggan. Segmentasi dengan cara tersebut, dapat mengalokasikan pelanggan untuk membentuk kelompok berdasarkan jumlah pengeluaran mereka. Hal ini dapat digunakan untuk mengidentifikasi pelanggan yang bernilai tinggi dan memprioritaskan pelayanan (Berry, M. dan Linoff, G., 2000).

2.2.2 Analisis Klaster

Clustering merupakan proses untuk mengelompokkan sekumpulan obyek data yang memiliki kemiripan tinggi ke dalam satu kelompok atau *cluster*, dan setiap kelompok yang terbentuk, tidak memiliki kemiripan dengan kelompok lainnya (Jiawei Han, M. K. dan J. P., 2012). Kemiripan dinilai berdasarkan nilai atribut yang mendeskripsikan obyek data. *Clustering* atau juga disebut sebagai segmentasi ini adalah salah satu metode penggalian data yang *unsupervised*, karena tidak ada atribut yang digunakan sebagai panduan atau tidak adanya label pada data dalam proses pembelajaran. Teknik dalam *clustering* dilakukan untuk menemukan pengetahuan dari kumpulan data (Chiu, C.Y. dkk., 2009).

Analisis *cluster* dalam konteks penggalian data yaitu proses menempatkan pelanggan atau prospek ke dalam kelompok yang memiliki ciri-ciri yang sama (Berry, M. dan Linoff, G., 2000). Algoritma *clustering* membangun sebuah model dengan melakukan serangkaian pengulangan dan berhenti ketika model tersebut telah terpusat dan batasan dari segmentasi telah stabil. Hasil dari *clustering* yang bagus tergantung dengan ukuran kesamaan dan metode yang digunakan. Pendekatan dalam *cluster* berdasarkan saran dari Fraley dan Raftery, membagi metode pengelompokan menjadi dua kelompok utama yaitu metode hierarki dan metode partisi (Fraley, C. dan Raftery, A. E., 1998).

- a. Metode hierarki yaitu metode yang membentuk *cluster* dengan mempartisi secara berulang-ulang dari atas ke bawah atau sebaliknya. Hasil dari metode hierarki berupa *dendogram* yang mewakili kelompok obyek dan tingkat kesamaan di mana terdapat perubahan pengelompokkan. Sebuah pengelompokan obyek data diperoleh dengan memotong *dendogram* pada tingkat kemiripan yang diinginkan (Maimon, O. dan Rokach, L., 2005).
- b. Metode partisi (non-hierarki) adalah metode yang membuat partisi k (*cluster*) dari sebuah n himpunan obyek data yang diberikan. Awalnya, setiap obyek data ditugaskan ke beberapa partisi. Kemudian secara iteratif, menggunakan teknik relokasi dengan mencoba berulang-ulang memindahkan obyek dari satu kelompok ke kelompok lain untuk mendapatkan partisi k (*cluster*) yang optimal. Jenis metode partisi ini

antara lain: *K-Means*, *K-Medoids*, dan *CLARANS* (Han, J. dkk., 2012).

2.2.3 Exploratory Data Analysis

Exploratory Data Analysis (EDA) diperkenalkan pertama kali oleh ahli statistik bernama John W. Tukey pada 1977. EDA didefinisikan sebagai pekerjaan detektif karena peneliti melakukan eksplorasi terhadap suatu data tanpa memiliki ide atau prasangka terhadap informasi apa yang akan didapatkan dari data yang sedang dianalisis (Martinez, W. L. dkk., 2017). EDA adalah pendekatan analisis data yang menggunakan berbagai teknik terutama visualisasi grafis untuk mengekstrak variabel penting dari *dataset* untuk memaksimalkan wawasan dari *dataset* tersebut (Heckert, N. A. dkk., 2002). EDA diterapkan pada data yang dikumpulkan tanpa hipotesis yang jelas dengan tujuan menemukan suatu petunjuk yang dapat menginspirasi ide dan hipotesis lainnya (De Mast, J. dan Kemper, B. P. H., 2009). Adapun langkah-langkah yang umumnya dilakukan oleh para analis data saat proses EDA (Radhi, M. dkk., 2021) sebagai berikut.

- a. Memaksimalkan wawasan ke dalam kumpulan data;
- b. Mengungkap struktur data;
- c. Ekstraksi variabel yang penting;
- d. Mendeteksi data menyimpang (*outlier*) dan anomali;
- e. Melakukan uji asumsi;
- f. Mengembangkan model; dan
- g. Menentukan faktor yang optimal.

2.2.4 Kemencengan Distribusi Data

Kemencengan (*skewness*) adalah tingkat ketidaksimetrisan atau kejauhan simetri dari sebuah distribusi. *Skewness* juga dapat diartikan sebagai kemencengan distribusi data. Sebuah distribusi yang tidak simetris akan memiliki rata-rata, median, dan modus yang tidak sama besarnya sehingga distribusi akan terkonsentrasi pada salah satu sisi dan kurvanya akan menceng. Nilai *skewness* menunjukkan data normal ketika nilai-nilai tersebut berada di antara rentang nilai dari -2 sampai dengan 2. Jika distribusi memiliki ekor yang lebih panjang ke kanan daripada yang ke kiri, maka distribusi disebut menceng ke kanan (*right skewed*) atau memiliki kemencengan positif. Sebaliknya, jika distribusi memiliki ekor yang lebih panjang ke kiri daripada yang ke kanan, maka distribusi disebut

menceng ke kiri (*left skewed*) atau memiliki kemencengan negatif (Mahendra, I. W. E. dan Parmithi, N. N., 2015). Besarnya kemencengan suatu kurva dapat diukur dengan koefisien kemencengan (*Coefficient of skewness*) yang disimbolkan dengan C_s . Nilai koefisien kemencengan distribusi data dihitung dengan Persamaan 2.1.

$$C_s = \frac{n \cdot \sum_{i=1}^n (x_i - \bar{x})^3}{(n-1)(n-2) \cdot Std^3} \quad (2.1)$$

Keterangan:

C_s = koefisien kemencengan

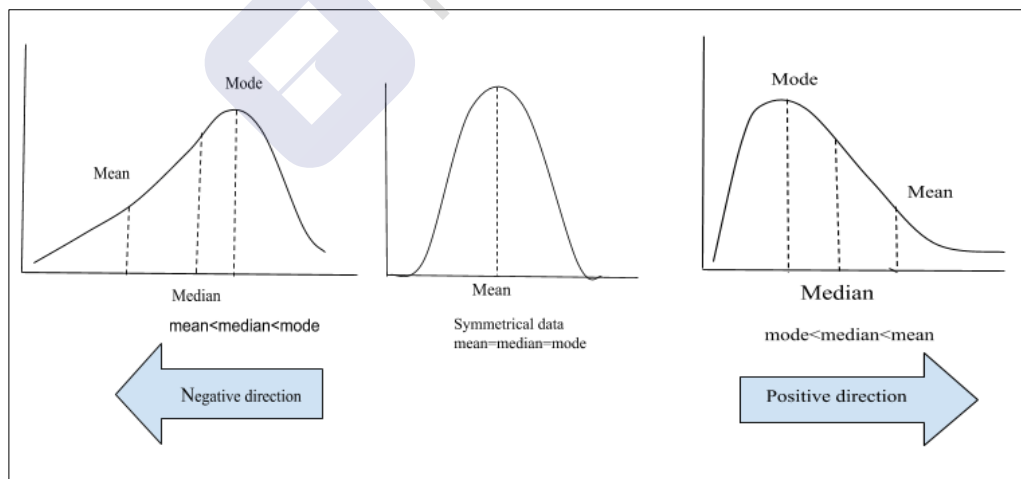
Std = deviasi standar (simpangan baku)

\bar{x} = rata-rata (*mean*)

x_i = nilai dari suatu variat

n = jumlah data

Kemencengan suatu kurva dapat dilihat dari perbedaan letak rata-rata, median, dan modusnya. Jika ketiga ukuran pemusatan data tersebut berada pada titik yang sama, maka dikatakan simetris atau data terdistribusi secara normal. Sedangkan, jika tidak berarti data tidak simetris atau tidak terdistribusi secara normal. Kurva distribusi yang menceng ke kanan, distribusi normal, dan distribusi yang menceng ke kiri dapat dilihat pada Gambar 2.1.



Gambar 2.1 Jenis-jenis kurva kemencengan distribusi data

2.2.5 Model RFMT

Model RFM diperkenalkan pertama kali oleh Hughes dan saat ini banyak digunakan oleh industri termasuk manufaktur, ritel, dan industri jasa (Hughes, A. M., 2000). Model RFM ini bertujuan untuk menentukan segmentasi pelanggan

berdasarkan tiga variabel yaitu *Recency of the last purchases*, *Frequency of the purchases*, dan *Monetary value of the purchases*. Umumnya, nilai RFM dihasilkan dari data pembelian historis dan kemudian dipetakan ke dalam sejumlah kelompok diskrit, yang merupakan langkah yang disebut pembobotan RFM. Ada dua metode umum untuk mengelompokkan pelanggan ke dalam kelompok tertentu, yaitu metode pembobotan kuantil pelanggan dan pembobotan kuantil perilaku (Miglautsch, J. R., 2000). Pembobotan kuantil pelanggan akan mengurutkan pelanggan dalam urutan *Frequency* dan *Monetary* secara menurun atau dalam urutan *Recency* secara menaik, kemudian membagi mereka ke kelompok dengan skor yang sama. Contohnya adalah penelitian yang dilakukan oleh (Christy, A. J. dkk., 2021) yang mengelompokkan pelanggan berdasarkan skor RFM mereka ke lima kelompok secara berurutan.

Kemudian penelitian yang dilakukan oleh (Zhou, J. dkk., 2021) memperkenalkan variabel *T* ke dalam model RFM, yang menjadi RFMT, untuk mendeskripsikan waktu pendaftaran konsumen dalam periode tertentu. Dalam penelitian, mereka mengembangkan sistem pengambilan konten web untuk mengumpulkan informasi yang diperlukan secara sistematis untuk periode tertentu dari situs web ritel yang ditargetkan. Lalu mereka menghitung nilai dari empat variabel (*Recency*, *Frequency*, *Monetary*, dan *Time*) dari data mentah dan memetakan masing-masing data RFMT ke dalam lima skala diskrit. Selanjutnya, mereka menerapkan algoritma *clustering* untuk mengelompokkan pelanggan ke dalam jumlah kluster yang optimal dan menganalisis karakteristik yang serupa dari setiap kluster. Informasi demografis dan preferensi produk di setiap kluster diperiksa untuk mendapatkan wawasan yang berguna untuk rekomendasi produk yang disesuaikan. Adapun rincian dari empat variabel RFMT sebagai berikut.

- a. *Recency* merupakan jarak antara waktu terakhir transaksi dengan waktu saat ini. Apabila jarak semakin kecil, maka nilai *R* semakin besar.
- b. *Frequency* merupakan seberapa sering jumlah transaksi yang dilakukan oleh pelanggan pada periode tertentu, misalnya seperti tiga kali dalam periode satu tahun. Apabila jumlah transaksi makin besar, maka nilai *F* juga makin besar.

- c. *Monetary* berarti jumlah uang yang dihabiskan pelanggan saat transaksi pada periode tertentu. Jika jumlah uang makin besar, maka nilai M juga makin besar.
- d. *Time* atau *Tenure* adalah interval waktu dari pendaftaran awal hingga waktu saat ini dalam waktu referensi tertentu. Variabel ini berpengaruh terhadap tingkat loyalitas pelanggan.

Semakin besar nilai R dan F, maka pelanggan akan melakukan transaksi kembali dengan perusahaan tersebut (Wu, J. dan Lin, Z., 2005). Selain itu, semakin besar nilai M, maka ada kecenderungan pelanggan akan memberikan respon kepada produk dan layanan perusahaan tersebut. Perusahaan menghitung skor RFM setiap pelanggan untuk menentukan probabilitas pelanggan akan merespon dengan baik, misalnya mengenai penawaran atau promosi. Banyak perusahaan percaya bahwa pelanggan yang telah menjadi pembeli baru, berfrekuensi tinggi, serta telah menghabiskan dalam jumlah besar pada jangka waktu tertentu, akan merespon positif penawaran dari perusahaan di masa depan. Skor tersebut dapat menentukan misalnya apakah seorang pelanggan akan dikirimkan katalog mahal atau hanya berupa diskon produk.

2.2.6 Normalisasi Data

Normalisasi dalam ilmu *data mining* adalah suatu proses penskalaan nilai atribut dari sebuah data sehingga data tersebut dapat memiliki skala atau rentang nilai yang telah ditetapkan sebelumnya (Widiowati, M. K., 2014). Masukan dari normalisasi data ini ialah *dataset* yang belum berada pada interval yang sama sehingga dapat mempengaruhi hasil penelitian. Luaran dari normalisasi data ini adalah *dataset* yang telah berada pada interval baru yang telah ditentukan. Terdapat beberapa metode yang digunakan dalam normalisasi data antara lain (Hanifa, T. T. dkk., 2017).

- a. *Min-Max Normalization*

Min-Max Normalization merupakan metode normalisasi dengan melakukan transformasi linier terhadap data asli sehingga menghasilkan keseimbangan nilai perbandingan antardata saat sebelum dan sesudah

proses. Metode ini dapat menggunakan Persamaan 2.2.

$$MinMax = \frac{(V - MinValue)}{(MaxValue - MinValue)} (MaxRange - MinRange) + MinRange \quad (2.2)$$

Keterangan:

V = nilai atribut yang akan dilakukan normalisasi data

$MinValue$ = nilai paling kecil (minimal) pada atribut

$MaxValue$ = nilai paling besar (maksimal) pada atribut

$MinRange$ = skala atau rentang nilai paling kecil (minimal)

$MaxRange$ = skala atau rentang nilai paling besar (maksimal)

b. *Z-score Normalization*

Z-score Normalization merupakan metode normalisasi data berdasarkan *mean* (nilai rata-rata) dan *standard deviation* (simpangan baku) dari data. Metode ini sangat berguna jika tidak diketahui nilai aktual minimum dan maksimum dari data. Metode ini menggunakan Persamaan 2.3.

$$Z - score = \frac{(x - \mu)}{\sigma} \quad (2.3)$$

Keterangan:

x = nilai atribut yang akan dilakukan normalisasi data

μ = nilai rata-rata (*mean*) pada atribut

σ = deviasi standar (simpangan baku) pada atribut

c. *Decimal Scaling Normalization*

Decimal Scaling Normalization merupakan metode normalisasi data dengan menggerakkan nilai desimal dari data ke arah yang diinginkan. Metode ini dapat menggunakan Persamaan 2.4.

$$V_i' = \frac{V_i}{10^j} \quad (2.4)$$

Keterangan:

V_i' = nilai baru hasil proses normalisasi data

V_i = nilai atribut yang akan dilakukan normalisasi data

j = nilai integer terkecil dengan nilai maksimum kurang dari 1

d. *Sigmoidal Normalization*

Sigmoidal Normalization merupakan metode normalisasi data secara nonlinier ke dalam rentang dari -1 sampai 1 dengan menggunakan fungsi

Sigmoid. Metode ini sangat berguna untuk data-data yang melibatkan data-data *outlier* (data yang menyimpang terlalu jauh dari data lainnya). Metode ini dapat menggunakan Persamaan 2.5.

$$V' = \frac{1 - e^{-x}}{1 + e^{-x}} \quad (2.5)$$

Keterangan:

V' = nilai baru hasil proses normalisasi data

x = deviasi standar (simpangan baku)

e = nilai eksponensial (2,718281828)

2.2.7 Principal Component Analysis

Principal Component Analysis (PCA) berguna untuk mengurangi dimensi permasalahan menjadi lebih sederhana dengan cara mengidentifikasi sebagian kecil komponen utama dan secara efektif merangkum sebagian besar variasi data (Ait-Sahalia, Y. dan Xiu, D., 2017). PCA menghitung matriks kovariansi dari kumpulan data, kemudian mencari *eigenvector* dan *eigenvalue*. Selanjutnya memilih beberapa *eigenvector* dari *eigenvalue* yang lebih banyak untuk membentuk matriks transformasi sehingga dapat mengurangi dimensi kumpulan data (Ganesh, V. dkk., 2018). PCA mempertahankan variansi untuk menemukan variabel baru yang merupakan fungsi linier dari yang ada di *dataset* asli. Variabel baru tersebut disebut *Principal Component* (PC) untuk menyelesaikan masalah *eigenvector* dan *eigenvalue* (Bi, M. dkk., 2016). Berikut adalah tahap-tahap yang dilakukan dalam metode PCA (Hardiansyah, B. dan Primandari, P. N., 2018) antara lain:

- a. Menghitung rata-rata semua sampel data di mana ⁽ⁱ⁾ adalah sampel data dan m adalah kolom yang diperoleh dengan Persamaan 2.6.

$$\mu = \frac{1}{m} \sum_{i=1}^m x^{(i)} \quad (2.6)$$

- b. Melakukan normalisasi pada setiap data dengan Persamaan 2.7.

$$x^{(i)} = x^{(i)} - \mu \quad (2.7)$$

- c. Menghitung matriks kovariansi (c) di mana (i) adalah *transpose* dari matriks $x(i)$ yang diperoleh menggunakan Persamaan 2.8.

$$C = \frac{1}{m} x^{(i)} x^{(i)T} \quad (2.8)$$

- d. Menghitung nilai *eigenvector* dan *eigenvalue* dengan menggunakan persamaan 2.9 dan 2.10.

$$C - \lambda I = 0 \quad (2.9)$$

$$(C - \lambda I)v = 0 \quad (2.10)$$

Keterangan:

C = Matriks kovariansi λ = *eigenvalue*

I = Matriks identitas v = *eigenvector*

- e. Menghitung nilai *eigenvector* terbesar yang bersesuaian dengan nilai *eigenvalue* dan dipilih menjadi *Principal Component* (PC). *Eigenvector* disusun dari yang terbesar ke yang terkecil seperti pada Persamaan 2.11.

$$v = (eig\ ke - 1, eig\ ke - 2, \dots, eign\ ke - n) \quad (2.11)$$

- f. Menghitung *Principal Component* dengan Persamaan 2.12.

$$PC = x^{(i)}v \quad (2.12)$$

- g. Melakukan transformasi data untuk menghasilkan data PCA menggunakan Persamaan 2.13.

$$PCA\ data = PC^T x^{(i)T} \quad (2.13)$$

2.2.8 Algoritma K-Means

Algoritma *K-Means* merupakan salah satu algoritma *clustering* yang bertujuan untuk membagi data menjadi beberapa kelompok. Algoritma ini menerima masukan berupa data tanpa label kelas (*unsupervised learning*). Proses *K-Means clustering* dilakukan oleh komputer dengan mengelompokan sendiri data-data yang menjadi masukannya tanpa mengetahui terlebih dulu target kelasnya. Pada setiap *cluster* terdapat titik pusat (*centroid*) yang merepresentasikan *cluster* tersebut. Algoritma atau langkah-langkah untuk melakukan *K-Means clustering* adalah sebagai berikut (Tan dkk., 2005).

- Menentukan nilai k sebagai jumlah klaster yang akan dibentuk.
- Menentukan nilai titik pusat klaster (*centroid*) awal. Pada tahapan ini ditentukan nilai *centroid* secara acak (Wakhidah, N., 2010), namun untuk perhitungan berikutnya menggunakan Persamaan 2.14.

$$\overrightarrow{V_{ij}} = \frac{1}{N_i} \sum_{k=0}^{N_i} X_{kj} \quad (2.14)$$

Di mana:

V_{ij} = *centroid cluster* ke- i untuk variabel ke- j

N_i = banyaknya data pada *cluster* ke- i

i, k = indeks dari *cluster*

j = indeks dari variabel

X_{kj} = nilai data ke- k yang ada di dalam *cluster* untuk variabel ke- j

- c. Menghitung jarak antara titik *centroid* dengan titik setiap obyek, dapat dilakukan dengan menggunakan *Euclidean Distance* (Wakhidah, N., 2010) dengan Persamaan 2.15.

$$D_e = \sqrt{(x_i - s_i)^2 + (y_t - t_t)^2} \quad (2.15)$$

Di mana:

D_e = *Euclidean Distance*

i = banyaknya data

(x, y) = koordinat data

(s, t) = koordinat *centroid*

- d. Mengelompokan data hingga terbentuk *cluster* dengan titik *centroid* dari setiap *cluster* yang merupakan titik *centroid* terdekat. Penentuan anggota *cluster* adalah dengan memperhitungkan jarak minimum obyek.
- e. Mengulangi langkah kedua (menghitung nilai *centroid*) hingga akhir sampai nilai dari titik *centroid* dan posisi data tidak ada lagi yang berubah.

2.2.9 Metode Elbow

Metode *Elbow* memberikan pengetahuan melalui cara memilih nilai *cluster* dan kemudian menambah nilai *cluster* tersebut untuk dijadikan model data dalam penentuan *cluster* terbaik (Merliana, N.P.E. dkk., 2015). Metode ini memberikan ide dengan cara memilih nilai *cluster* dan kemudian menambah nilai *cluster* tersebut untuk dijadikan model data dalam penentuan *cluster* terbaik. Selain itu, persentase perhitungan yang dihasilkan menjadi pembanding antara jumlah *cluster* yang ditambah. Hasil persentase yang berbeda dari setiap nilai

cluster dapat ditunjukkan dengan grafik sebagai sumber informasinya. Jika nilai *cluster* pertama dengan nilai *cluster* kedua memberikan sudut dalam grafik atau nilainya mengalami penurunan paling besar, maka nilai *cluster* tersebut adalah yang terbaik.

Terdapat beberapa cara yang dapat dilakukan untuk mengevaluasi performa *K-Means clustering*. Beberapa peneliti menggunakan istilah *clustering validity* terhadap proses ini. Banyak usulan nilai statistik atau indeks yang digunakan, tapi kebanyakan menggunakan prinsip bahwa antarobyek dalam satu klaster bersifat dekat atau mirip, dan antarklaster bersifat jauh atau tak mirip. Dari sekian banyak kriteria evaluasi, yang akan dibahas adalah kriteria yang menggunakan konsep jumlah kuadrat (*Sum of Squares*). Nilai jumlah kuadrat dari suatu hasil *clustering* dapat dihitung menggunakan beberapa metode sebagai berikut (Sartono, B., 2016).

a. *Within-Cluster Sum of Squares*

Within-Cluster Sum of Squares yang dinotasikan dengan SS_w . Ini merupakan ukuran kedekatan antarobyek dalam suatu klaster. Nilai SS_w yang kecil mengindikasikan kemiripan antarobyek dalam suatu klaster. Dari setiap klaster bisa dihitung nilai SS_w menggunakan Persamaan 2.16.

$$SS_w = \sum_i \sum_j (x_{ij} - \bar{x}_j)^2 \quad (2.16)$$

dengan i adalah indeks amatan, j adalah indeks variabel, dan \bar{x}_j adalah rata-rata nilai variabel ke- j pada suatu klaster tertentu.

b. *Between-Cluster Sum of Squares*

Between-Cluster Sum of Squares yang dinotasikan dengan SS_b . Ini merupakan ukuran keterpisahan antarklaster. Nilai SS_b yang besar mengindikasikan hasil *clustering* yang semakin baik. Dari setiap klaster bisa dihitung nilai SS_b menggunakan Persamaan 2.17.

$$SS_b = \sum_k \sum_j (\bar{x}_{jk} - \bar{\bar{x}}_j)^2 \quad (2.17)$$

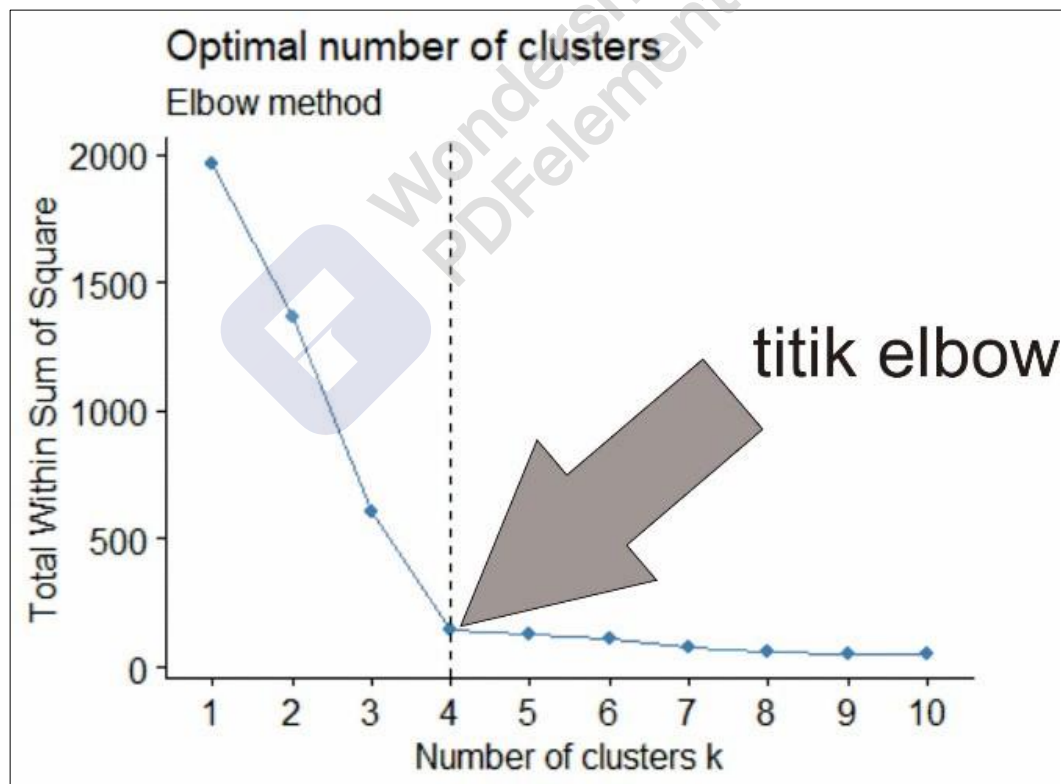
dengan k adalah indeks klaster, j adalah indeks variabel, \bar{x}_{jk} merupakan nilai rata-rata variabel ke- j pada klaster ke- k , dan $\bar{\bar{x}}_j$ adalah nilai rata-rata variabel ke- j dari semua amatan.

c. *Total-Cluster Sum of Squares*

Total-Cluster Sum of Squares yang dinotasikan dengan SS_t . Ini merupakan gabungan antara ukuran kedekatan dan keterpisahan obyek dalam suatu klaster. Dari setiap klaster yang diproses bisa dihitung nilai SS_t menggunakan Persamaan 2.18.

$$SS_t = SS_w + SS_b \quad (2.18)$$

Setelah dihitung, akan ada beberapa nilai K yang mengalami penurunan paling besar, dan selanjutnya hasil dari nilai K akan turun secara perlahan-lahan sampai hasil dari nilai K tersebut stabil. Misalnya, nilai cluster $K=2$ ke $K=3$, kemudian dari $K=3$ ke $K=4$. Terlihat penurunan drastis membentuk siku pada titik $K=4$ maka nilai *cluster* K yang ideal adalah $K=4$. Ilustrasi penentuan nilai K yang optimal dapat dilihat pada Gambar 2.2.



Gambar 2.2 Nilai k yang optimal pada metode *Elbow*

Langkah-langkah yang perlu dilakukan pada metode *Elbow* dalam menentukan nilai K pada algoritma *K-Means* yaitu (Merliana, N. P. E.dkk., 2015).

- a. Mulai.
- b. Inisialisasi awal nilai *cluster K*.
- c. Naikkan nilai K secara bertahap.
- d. Hitung *Sum of Squared Errors* dari tiap nilai K .
- e. Memilih *Sum of Squared Errors* dari nilai K yang turun secara drastis.
- f. Tetapkan nilai K yang berbentuk siku.

2.2.10 Python

Dibuat oleh Guido van Rossum dan dirilis pada tahun 1991, Python sekarang menjadi bahasa pemrograman multifungsi yang canggih (Python Core Team, 2015) Python dirancang untuk menekankan pada keterbacaan kode dengan memanfaatkan *whitespace* yang substansial dan kesederhanaan karena memungkinkan *programmer* pemula untuk menulis model dan konsepsi dalam jumlah baris kode yang lebih sedikit dibandingkan dengan bahasa lain seperti C++ atau Java. Hal ini menjadikan Python sebagai bahasa pemrograman yang sangat populer, banyak digunakan untuk aplikasi *desktop* atau web, serta proyek pengembangan proyek skala kecil atau besar.

Sebagai bahasa pemrograman multifungsi, Python adalah yang bisa dimanfaatkan untuk mengembangkan proyek apa pun. Python dapat digunakan untuk pemrograman *front end* (sisi klien) yang berinteraksi dengan pengguna dan *back end* (sisi server) yang berinteraksi dengan basis data dalam sebuah situs web. Python juga dapat digunakan untuk analisis data pada penelitian ilmiah, mengembangkan kecerdasan, mengembangkan aplikasi *online* dan *offline*, *game*, dan jenis aplikasi lain yang untuk keperluan produktivitas. Singkatnya, Python adalah bahasa pemrograman *jack-of-all-trade*. Dan menguasainya, berpotensi memungkinkan pengembang perangkat lunak menjadi ahli dalam semua jenis pemrograman aplikasi. Dengan fungsi yang begitu banyak, pastinya Python juga memiliki banyak pustaka (*library*) untuk mendukung fungsionalitas yang ada. Adapun beberapa pustaka yang digunakan dalam penelitian ini antara lain:

a. NumPy

NumPy adalah pustaka Python yang digunakan untuk keperluan dengan *array*. Pustaka ini juga memiliki fungsi untuk bekerja dalam domain aljabar linier, transformasi *Fourier*, dan matriks. NumPy dibuat pada tahun 2005 oleh Travis Oliphant (Harris, C. R., dkk., 2020). Pustaka ini adalah proyek *open source* dan dapat digunakan secara bebas. NumPy adalah singkatan dari *Numerical Python*. Di Python sebenarnya ada tipe data *list* yang menangani keperluan *array*, tetapi memiliki performa yang lambat. NumPy bertujuan untuk menyediakan obyek *array* hingga 50x lebih cepat daripada *list*. Obyek *array* di NumPy disebut *ndarray*, menyediakan banyak fungsi pendukung yang membuat bekerja dengan *ndarray* menjadi sangat mudah. *Array* sangat sering digunakan dalam ilmu data, di mana kecepatan dan sumber daya merupakan hal yang sangat penting.

b. Pandas

Pandas adalah pustaka Python yang digunakan untuk keperluan dengan kumpulan data (*dataset*). Pandas memiliki fungsi untuk menganalisis, membersihkan, mengeksplorasi, dan memanipulasi data. Nama "Panda" memiliki referensi ke "*Panel Data*" dan "*Python Data Analysis*". Pandas dibuat oleh Wes McKinney pada tahun 2008 (McKinney, W., 2010). Pandas memungkinkan seorang peneliti untuk menganalisis data besar dan membuat kesimpulan berdasarkan teori statistik. Pandas dapat membersihkan kumpulan data yang berantakan dan membuatnya dapat dibaca dan relevan. Data yang relevan sangat penting dalam ilmu data. Selain itu, Pandas juga dapat menghapus baris yang tidak relevan, atau berisi nilai yang salah, seperti nilai kosong atau NULL.

c. Matplotlib

Matplotlib adalah pustaka (*library*) *open-source* yang membantu dalam pembuatan visualisasi grafik. Pustaka ini dibuat oleh John D. Hunter, yang merupakan seorang ahli saraf (Hunter, J. D., 2007). John menulis Matplotlib pada saat penelitian pasca-doktoralnya di jurusan Neurobiologi.

Tujuan dari pustaka ini adalah untuk mempelajari aktivitas yang terjadi di korteks serebral pasien yang menderita epilepsi dengan memplot aktivitas ini dalam bentuk visualisasi grafik. Satu-satunya tujuan mem-*plot* grafik adalah untuk visualisasi yang lebih baik dan untuk mempelajari pola umum di dalamnya. Rilis pertama Matplotlib adalah pada tahun 2003. Seiring waktu, Matplotlib menjadi salah satu pustaka *plot* yang paling banyak digunakan untuk *plot* data dan grafik komputasi.

d. Seaborn

Seaborn adalah pustaka (*library*) infografis statistik yang dibangun berdasarkan kemampuan Matplotlib. Seaborn dirancang untuk meningkatkan fungsi dari Matplotlib, mengatasi beberapa masalah umum yang dialami oleh pengguna Matplotlib, dengan tujuan membuat visualisasi yang secara estetika lebih cepat dan lebih mudah (Waskom, M. L., 2021). API (*Application Programming Interface*) Matplotlib adalah tingkat rendah dibandingkan dengan Seaborn, dan akibatnya pengguna sering perlu menulis cukup banyak kode *boilerplate*. Seaborn berupaya mengurangi banyak redundansi di Matplotlib dan membuat tugas visualisasi Matplotlib yang awalnya sulit menjadi lebih mudah. Ada beberapa fitur penting dari Seaborn yang membuatnya lebih disukai banyak orang daripada Matplotlib. Seaborn memungkinkan pengguna untuk membuat grafik statistik dengan mudah berkat fitur seperti: antarmuka tingkat tinggi, tema estetis, perbandingan mudah antara beberapa variabel, kisi *multi-plot*, visualisasi univariat dan bivariat, estimasi otomatis untuk regresi, dan *plot* waktu yang mudah data *time series* (data berkelanjutan).

e. Plotly

Plotly adalah pustaka (*library*) visualisasi data *open-source* untuk Python. Mirip dengan Altair, Plotly menggunakan format data JSON untuk memvisualisasikan data. Plotly mendukung pembuatan lebih dari 40 jenis bagan yang berbeda untuk memvisualisasikan berbagai jenis data seperti data statistik, data geografis, data keuangan, dan data ilmiah (Plotly

Technologies Inc, 2015). Plotly semakin populer dalam beberapa tahun terakhir, dan merupakan salah satu pustaka yang paling banyak digunakan. Diperluas dengan Plotly Express (API tingkat tinggi resmi milik Plotly) yang terlihat mirip dengan Seaborn, menjadikannya transisi yang cukup mudah bagi kebanyakan orang yang sudah mengenal Seaborn. Hal ini membuat adopsi jauh lebih intuitif dan sederhana dan Plotly semakin sering muncul dalam visualisasi berbasis web.

f. Scikit-learn

Scikit-learn (Sklearn) adalah pustaka Python yang dirancang untuk memfasilitasi penggunaan pembelajaran mesin dan algoritma AI. Pustaka ini mencakup algoritma yang digunakan untuk klasifikasi, regresi, dan pengelompokan seperti *random forest* dan *gradient boosting* (Pedregosa, F., dkk., 2011). Scikit-learn dirancang untuk berinteraksi dengan pustaka umum seperti NumPy dan SciPy. Meskipun scikit-learn tidak dirancang secara khusus, scikit-learn juga dapat berinteraksi secara baik dengan Pandas. Scikit-learn menyertakan alat yang berguna untuk memfasilitasi penggunaan algoritma pembelajaran mesin. Mengembangkan alur pembelajaran mesin yang secara akurat memprediksi perilaku suatu sistem dalam melakukan pemisahan *dataset* pelatihan dan pengujian, menguji performa algoritma untuk menentukan seberapa baik fungsinya, dan memastikan model tidak mengalami *overfitting* atau *underfitting*.

g. Yellowbrick

Yellowbrick adalah rangkaian alat analisis dan pembuatan visualisasi yang dirancang untuk memfasilitasi pembelajaran mesin dengan pustaka Scikit-learn. Pustaka ini mengimplementasikan obyek API bernama *Visualizer*, yang merupakan *estimator* milik Scikit-learn (Bengfort, dkk., 2018). Mirip dengan *Transformers* atau *Model*, *Visualizer* belajar dari data dengan membuat representasi visual dari alur kerja pemilihan model. *Visualizer* juga memungkinkan pengguna untuk mengarahkan proses pemilihan model, membangun *feature engineering*, pemilihan algoritma, dan penyesuaian *hyperparameter*. Misalnya, pustaka ini dapat membantu

mendiagnosis masalah umum seputar kompleksitas dan bias pada model tertentu, heteroskedastisitas, *underfitting* dan *overfitting*, atau masalah keseimbangan distribusi pada tiap kelas. Dengan menerapkan *Visualizer* ke alur kerja pemilihan model, Yellowbrick memungkinkan untuk mengarahkan model prediktif menuju hasil yang lebih akurat dan cepat.

h. Joblib

Untuk sebagian besar masalah pemrograman, komputasi paralel benar-benar dapat meningkatkan kecepatan komputasi. Seiring dengan peningkatan daya komputasi PC (*Personal Computer*), sebenarnya cukup meningkatkan komputasi dengan menjalankan kode paralel di PC pribadi. Joblib adalah pustaka yang dapat dengan mudah mengubah kode Python menjadi mode komputasi paralel dan tentu saja mampu meningkatkan kecepatan komputasi (Moreau, T., dkk., 2021). Joblib adalah seperangkat alat untuk menyediakan *pipelining* ringan dengan Python, khususnya meliputi: fungsi *disk caching* yang transparan dan *lazy re-evaluation* (*memoize pattern*), komputasi paralel sederhana yang mudah, serta dioptimalkan untuk menjadi pustaka yang cepat dan kuat khususnya pada data besar dan memiliki pengoptimalan khusus untuk *array* NumPy.

BAB III

METODE PENELITIAN

3.1 Bahan dan Data

3.1.1 Data yang Diperoleh

Dataset pelanggan Starbucks yang digunakan pada penelitian ini berupa data sekunder yang diperoleh dari situs web Kaggle (<https://www.kaggle.com/datasets/blacktile/starbucks-app-customer-reward-program-data>). *Dataset* yang dimanfaatkan berjumlah tiga *dataset* dengan rincian *portfolio.json* terdiri dari enam kolom variabel dan memiliki jumlah *record* sebanyak 10 baris. *Dataset* ini berisi metadata untuk setiap jenis penawaran yang dikirimkan kepada pelanggan Starbucks. *Dataset* ini terdiri dari enam variabel meliputi: *id* (nomor penawaran), *offer_type* (jenis penawaran), *difficulty* (tingkat kesulitan penawaran), *reward* (hadiah berupa diskon yang diberikan setelah menyelesaikan penawaran), *duration* (waktu untuk menyelesaikan penawaran), dan *channels* (media untuk mengirim atau menerima penawaran).

Kemudian *dataset profile.json* terdiri lima kolom variabel dan memiliki jumlah *record* sebanyak 17.000 baris. *Dataset* ini berisi profil demografis untuk setiap pelanggan Starbucks. Pada *dataset* ini terdapat variabel *become_member_on* yang berisi tanggal pendaftaran pelanggan ke aplikasi Starbucks, yang dimulai dari 29 Maret 2013 s/d 26 Juli 2018. *Dataset* ini terdiri dari lima variabel meliputi: *age* (usia pelanggan), *became_member_on* (tanggal pendaftaran pelanggan), *gender* (jenis kelamin pelanggan), *id* (nomor pendaftaran pelanggan), dan *income* (pendapatan pelanggan).

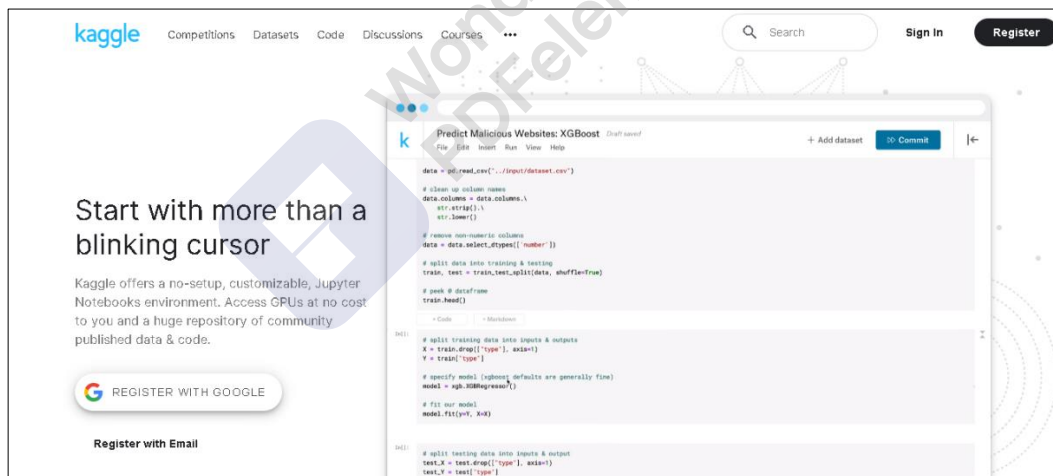
Lalu *dataset transcript.json* terdiri dari empat kolom variabel dan memiliki jumlah *record* sebanyak 306.534 baris. *Dataset* ini berisi catatan peristiwa yang dilakukan oleh setiap pelanggan di aplikasi Starbucks. *Dataset* ini terdiri dari empat variabel meliputi: *event* (nama peristiwa yang dilakukan), *person* (nomor pendaftaran pelanggan), *time* (waktu yang dihitung sejak dimulainya peristiwa), dan *value* (sebuah tipe data *dictionary* yang berisi nilai

nomor penawaran atau jumlah transaksi). Tiga *dataset* Starbucks yang telah dikonversi menjadi format berkas Microsoft Excel (.xlsx) dapat dilihat pada Lampiran 1.

3.1.2 Prosedur Pengumpulan Data

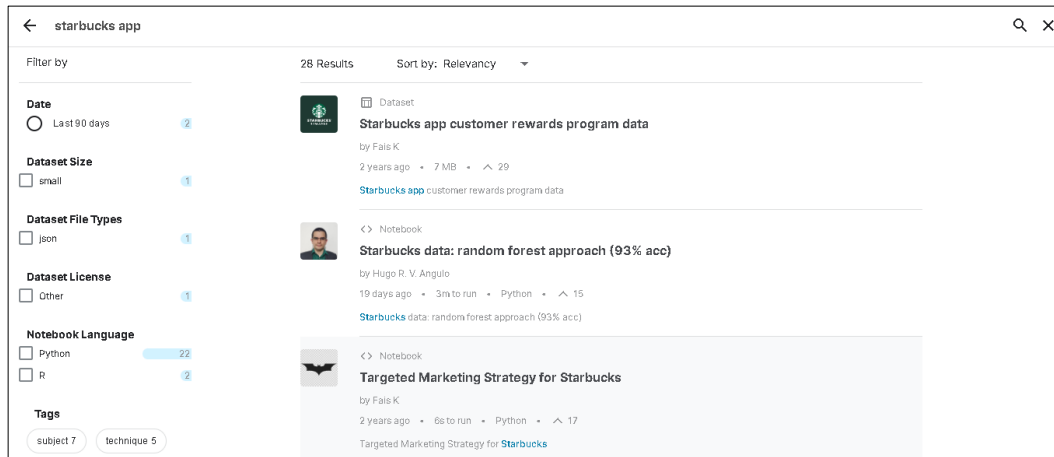
Prosedur pengumpulan data adalah proses yang dilakukan untuk mengumpulkan data yang akan digunakan dalam penelitian. Prosedur pengumpulan data dalam penelitian ini dilakukan dengan cara mengunduh *dataset* dari situs web Kaggle.com berupa *file JavaScript Object Notation* (JSON). Adapun langkah-langkah pengumpulan data yang dilakukan oleh penulis sebagai berikut.

a. Buka situs web Kaggle.com dengan alamat URL (<https://www.kaggle.com>) pada *address bar* di browser, lalu tekan *enter*. Setelah proses pemuatan situs web Kaggle.com selesai, maka akan muncul tampilan seperti Gambar 3.1.



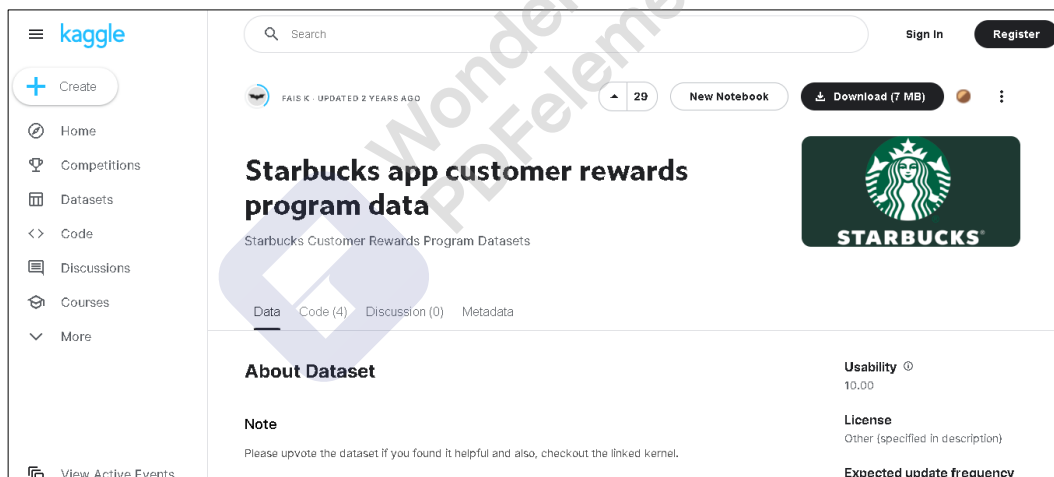
Gambar 3.1 Tangkapan layar beranda situs web Kaggle.com

b. Kemudian klik *search bar* yang ada di menu navigasi dan tuliskan kata kunci “starbucks app”, lalu tekan *enter*. Pada halaman berikutnya akan muncul beberapa pilihan *dataset*. Silakan pilih *dataset* yang paling atas dari hasil pencarian seperti terlihat pada Gambar 3.2.



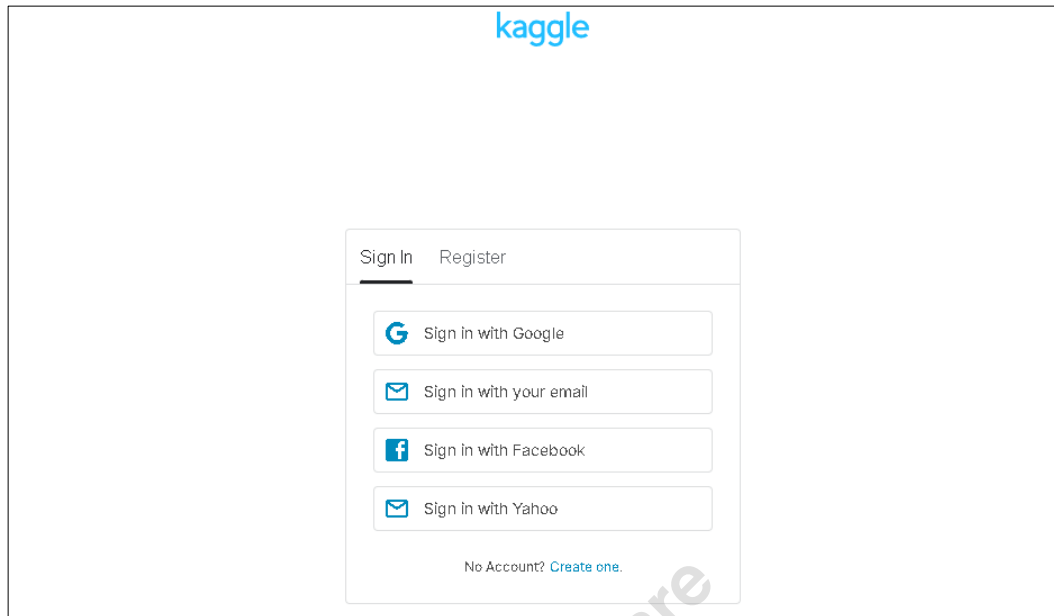
Gambar 3.2 Tangkapan layar hasil pencarian situs web Kaggle.com

c. Setelah memilih *dataset*, selanjutnya adalah mengunduh *dataset* yang akan digunakan dalam penelitian dengan menekan tombol *download* yang terdapat di halaman tersebut seperti terlihat pada Gambar 3.3.



Gambar 3.3 Tangkapan layar halaman dataset pelanggan Starbucks

d. Saat akan mengunduh *dataset*, situs web Kaggle.com mengharuskan semua pengunjung untuk *sign in* terlebih dahulu. Jika belum memiliki akun Kaggle, maka dapat mendaftarkan akun atau *sign in* secara langsung melalui beberapa metode *sign in* yang tersedia, misalnya melalui alamat *email*. Adapun tampilan halaman *sign in* dari situs web Kaggle.com dapat dilihat pada Gambar 3.4.



Gambar 3.4 Halaman *sign in* Kaggle.com

3.2 Aturan Bisnis (*Business Rule*)

Berdasarkan data dari dokumen profil perusahaan Starbucks Corporation tahun 2014, Starbucks adalah *brand* kedai kopi paling populer yang telah tersebar hingga 72 negara di seluruh dunia. Faktor utama Starbucks dalam meraih kesuksesan adalah kemampuan mereka untuk memanfaatkan strategi diferensiasi produk dengan menawarkan produk premium dan berkualitas tinggi. Dengan memberikan setiap pelanggan pelayanan terbaik yang mereka sebut dengan “Starbucks *Experience*”, kedai yang bersih dan terawat di negara manapun, sehingga Starbucks berhasil membangun loyalitas pelanggan yang tinggi. Namun di tengah popularitas *brand* Starbucks, muncul beberapa kelemahan dan ancaman yang dapat membuat Starbucks menjadi kurang diminati di beberapa negara. Salah satu contoh kelemahan dan ancaman Starbucks ialah harga produknya yang cukup tinggi bagi masyarakat menengah ke bawah dan banyaknya kompetitor yang menawarkan produk berkualitas namun dengan harga yang lebih terjangkau. Adapun daftar harga produk kopi Starbucks dapat dilihat pada Gambar 3.5.

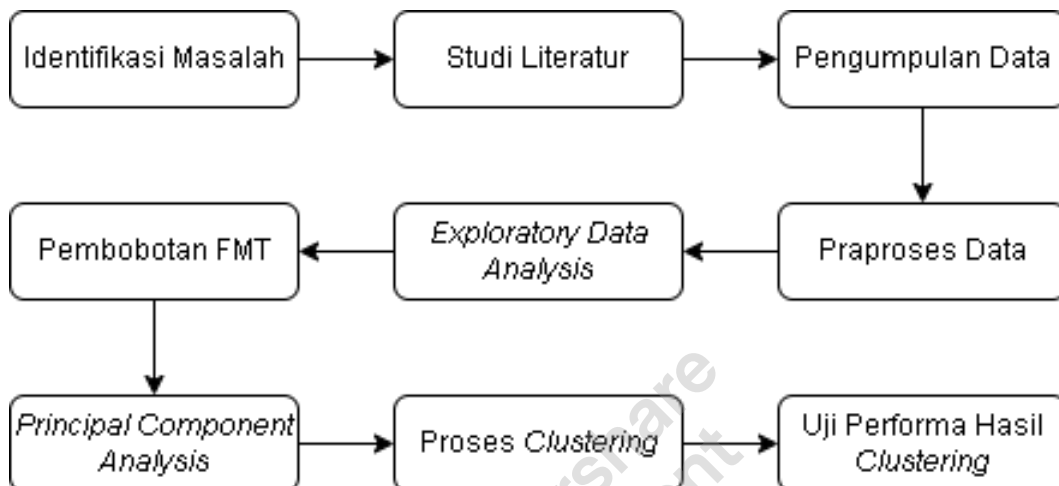
ESPRESSO & BREWED COFFEE			
HANDCRAFTED ESPRESSO			
	tall 355 mL	grande 473 mL	venti 592 mL
Double Shots® Iced Shaken Espresso	48 (10 Oz)		
Caffè Americano	34	37	39
Caramel Macchiato	57	62	65
Asian Dolce Latte	53	58	60
Vanilla/Caramel/Hazelnut Latte	50	55	58
Caffè Latte	44	49	52
Cappuccino	44	49	52
Caffè Mocha	53	58	60
BREWED COFFEE			
Cold Brew	41	45	48
Vanilla Sweet Cream Cold Brew	47	51	54
Freshly Brewed Coffee Starbucks® Medium or Dark Roast	23	25	27

Gambar 3.5 Daftar harga produk kopi Starbucks di Indonesia

Salah satu cara agar tetap mampu bersaing dengan kompetitor adalah dengan melakukan promosi. Dengan menjalankan promosi yang sesuai dan efektif, usaha yang dijalankan akan lebih dikenal oleh masyarakat luas, mendatangkan konsumen baru, dan mempertahankan konsumen loyal (Fildzah, A. N. dan Mayangsari, I. D., 2018). Namun yang perlu diperhatikan dalam menjalankan promosi yang efektif yaitu mengetahui profil pelanggan melalui segmentasi pasar. Pada dasarnya, segmentasi pasar adalah suatu strategi yang didasarkan pada falsafah manajemen pemasaran yang orientasinya adalah konsumen. Dengan melaksanakan segmentasi pasar, kegiatan pemasaran dapat dilakukan dengan lebih terarah, serta sumber daya yang dimiliki perusahaan dapat digunakan secara lebih efektif dan efisien dalam rangka memberikan kepuasan bagi konsumen (Budiman, A. dan Tjahjadi, B., 2019). Segmentasi pasar ini dapat dilakukan dengan cara membagi pelanggan berdasarkan profil demografi, geografi, psikografi, dan perilaku mereka saat membeli produk.

3.3 Tahapan Penelitian

Tahapan penelitian digunakan sebagai panduan dalam penyusunan laporan tugas akhir agar terarah dan sistematis. Adapun urutan dari pengerjaan dalam membangun sistem seperti terlihat pada diagram alir di Gambar 3.6.



Gambar 3.6 Diagram alir tahap penelitian tugas akhir

Tahap identifikasi masalah merupakan tahapan awal penyusunan laporan tugas akhir yaitu melakukan analisis permasalahan yang akan dijadikan topik pengerjaan tugas akhir. Obyek yang dipilih dan menjadi masukan pada proses identifikasi adalah data pelanggan Starbucks. Luaran dari proses identifikasi berupa permasalahan yang diangkat yaitu segmentasi pelanggan berdasarkan variabel *frequency*, *monetary*, dan *tenure* (FMT) dalam perencanaan strategi pengelolaan hubungan pelanggan. Pada tahap studi literatur, berbagai informasi didapatkan dari pengumpulan referensi, di antaranya dari buku, *e-book*, penelitian sebelumnya seperti pada jurnal dan *paper*, tugas akhir, artikel, serta dokumen yang terkait dengan segmentasi pelanggan. Segmentasi ini akan melakukan proses penggalian data, sehingga referensi yang diperlukan berhubungan dengan segmentasi pelanggan, analisis klaster, *Exploratory Data Analysis* (EDA), model RFMT, normalisasi data, *Principal Component Analysis* (PCA), algoritma *K-Means*, dan metode *Elbow*. Luaran yang didapatkan pada proses studi literatur berupa kajian pustaka sebagai acuan dalam landasan teori untuk pembelajaran dan pedoman pengerjaan tugas akhir.

Pada tahap pengumpulan data, penulis melakukan pengumpulan data-data yang dibutuhkan sebagai pendukung utama dalam pengerjaan tugas akhir. Data yang diambil disesuaikan dengan topik dan batasan permasalahan pada tugas akhir. Pengumpulan data dilakukan dengan cara mengunduh *dataset* di situs web Kaggle.com. Selanjutnya ada tahap praproses data, data mentah yang diperoleh dari *dataset* pelanggan Starbucks akan melalui serangkaian proses pembersihan dan perbaikan struktur data yang meliputi:

- a. Proses pemeriksaan dan pembersihan data (*data profiling and cleaning*) untuk menghilangkan baris data dengan nilai kosong atau data tidak valid.
- b. Proses pemilihan data (*data selection*) berdasarkan variabel yang disesuaikan dengan model FMT yaitu: jumlah frekuensi transaksi, jumlah nominal transaksi, dan waktu pendaftaran pelanggan.
- c. Proses persiapan data (*data preprocessing*) dengan cara mereduksi kolom yang tidak sesuai dengan kebutuhan dalam melakukan proses *clustering*.
- d. Proses transformasi data (*data transformation*) ke dalam bentuk model *frequency*, *monetary*, dan *tenure* sehingga dapat digunakan sebagai variabel untuk proses *clustering*. Variabel yang akan digunakan dalam proses transformasi data ditunjukkan pada Tabel 3.1.

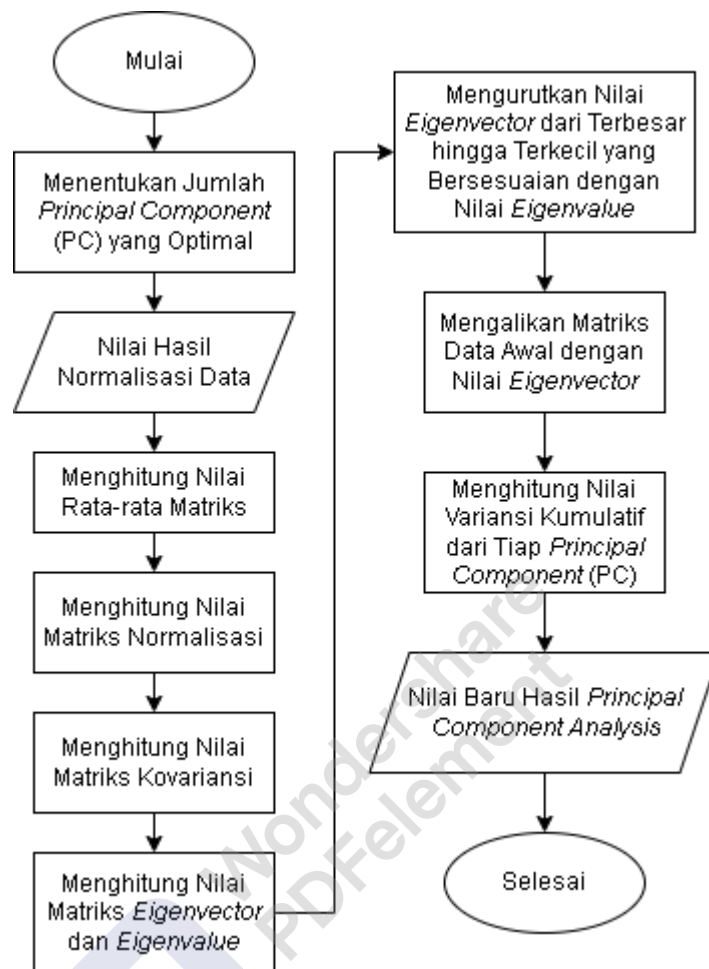
Tabel 3.1 Variabel untuk proses transformasi data

No.	Variabel Awal	Variabel Transformasi Data
1.	Frekuensi pembelian pelanggan	<i>Frequency</i>
2.	Jumlah (dalam nominal) yang dibelanjakan oleh pelanggan	<i>Monetary</i>
3.	Waktu pendaftaran pelanggan ke aplikasi Starbucks	<i>Tenure</i>

Kemudian obyek yang menjadi target dari tahap *Exploratory Data Analysis* (EDA) adalah variabel pada *dataset* pelanggan Starbucks. Dari ketiga *dataset* yang ada, hanya dua *dataset* yang akan digunakan oleh penulis dalam proses EDA yaitu *dataset profile.json* dan *dataset transcript.json*. Pada *dataset profile.json*, penulis melakukan proses EDA untuk mengetahui profil demografi

dan kebiasaan membeli pelanggan Starbucks. Pada *dataset transcript.json*, penulis melakukan proses EDA untuk memahami seberapa sukses penawaran-penawaran yang telah dikirim kepada para pelanggan Starbucks. Luaran dari proses ini berupa visualisasi data berbentuk *bar plot*, *line plot*, histogram, dan jenis visualisasi grafis lainnya. Tahap selanjutnya adalah melakukan pembobotan fitur *frequency*, *monetary*, dan *tenure* (FMT) pada *dataset* hasil praproses data. Nilai pembobotan ini diperoleh dan dihitung berdasarkan kebutuhan bisnis perusahaan Starbucks. Setelah dilakukan pembobotan, akan dilakukan normalisasi data agar rentang nilai pada variabel yang diberi bobot tidak menyimpang jauh sehingga terbentuk *outlier* (data yang terpaut jauh dari titik data lainnya). Metode normalisasi data yang digunakan dalam penelitian ini yaitu *Min-Max Normalization*. *Outlier* harus mendapatkan perlakuan khusus karena dapat menyebabkan terjadinya bias pada hasil penelitian.

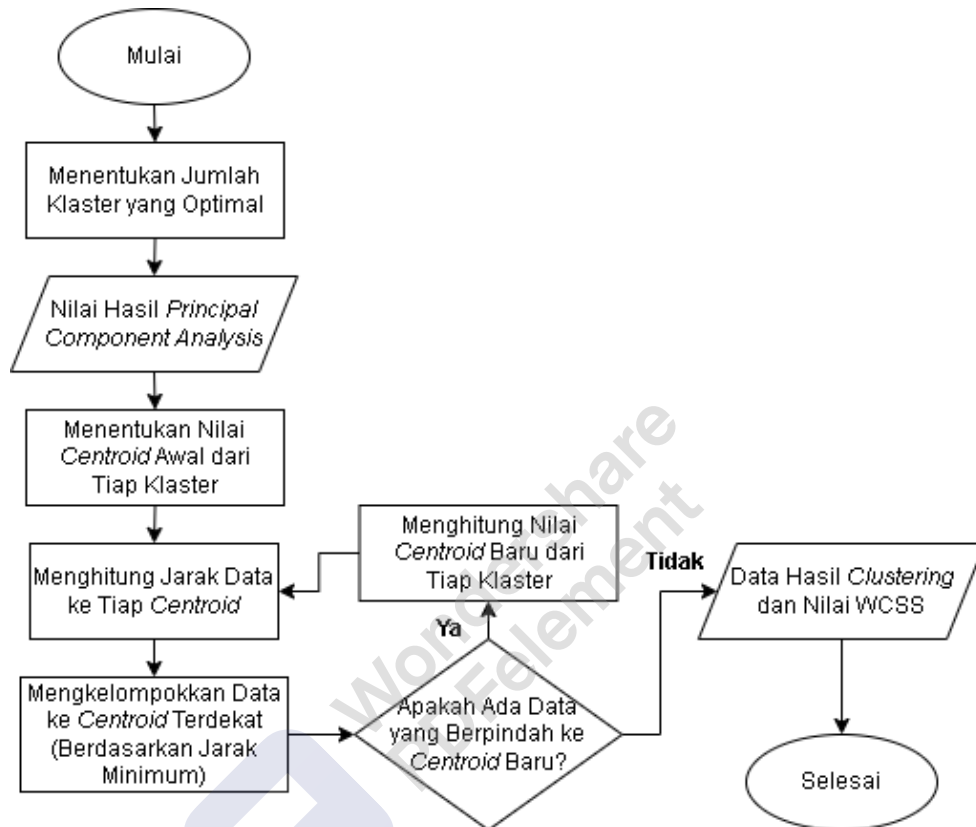
Sebelum melakukan *clustering* dengan algoritma *K-Means*, perlu dilakukan reduksi jumlah dimensi dengan metode *Principal Component Analysis* (PCA). Metode ini digunakan untuk mendekomposisi suatu matriks yang besar menjadi komponen vektor *orthogonal* menggunakan nilai *eigenvector* dan *eigenvalues* dari matriks tersebut. Langkah awal PCA adalah menentukan jumlah komponen yang optimal. Penentuan ini dilakukan dengan bantuan visualisasi grafik yang menjelaskan persentase variansi data dari tiap jumlah komponen. Lalu dilakukan perhitungan rata-rata matriks menggunakan Persamaan 2.6. Setelah itu, menstandarisasi nilai atau melakukan normalisasi data pada model *frequency*, *monetary*, dan *tenure* (FMT). Dalam penelitian ini, digunakan komponen yang menjelaskan minimal lebih dari 80% variansi data, sehingga diperoleh jumlah komponen sebanyak tiga yang menjelaskan 89% dari keseluruhan informasi pada model FMT. Adapun diagram alir untuk proses reduksi dimensi menggunakan *Principal Component Analysis* (PCA) dapat dilihat pada Gambar 3.7.



Gambar 3.7 Diagram alir untuk tahap *Principal Component Analysis* (PCA)

Pada proses *clustering* dilakukan dengan memproses hasil data dari model FMT yang telah diterapkan analisis PCA. Hasil proses tersebut akan diolah terlebih dahulu menggunakan metode *Elbow* untuk menentukan jumlah kluster k yang optimal. Luaran kluster yang terbentuk akan diberi label atau nama untuk memudahkan perusahaan dalam mengingat karakteristik pelanggannya. Tahap pertama tersebut lalu dilanjutkan dengan menentukan titik *centroid* awal dalam penerapan algoritma *K-Means*. Selanjutnya dilakukan perhitungan untuk mengetahui jarak data ke tiap titik *centroid*. Data akan dikelompokkan ke titik *centroid* berdasarkan jarak minimum terdekat. Kemudian akan dilakukan pemeriksaan apakah ada data yang berubah posisi. Jika ada, maka akan dilakukan perhitungan nilai *centroid* baru menggunakan Persamaan 2.14 dan kembali ke proses perhitungan jarak data ke titik *centroid* baru. Iterasi ini akan terus

berlangsung hingga tidak ada perubahan posisi data. Adapun diagram alir untuk proses *clustering* menggunakan algoritma *K-Means* dapat dilihat pada Gambar 3.8.



Gambar 3.8 Diagram alir untuk tahap *K-Means clustering*

Pada tahap uji performa algoritma *K-Means* ini, akan diukur tingkat performa model hasil *clustering* yang telah terbentuk pada tahap sebelumnya. Pada uji performa ini akan diketahui seberapa baik kluster dikelompokkan dan seberapa dekat obyek yang berhubungan dalam satu kluster. Uji performa ini dilakukan secara internal yaitu dengan menggunakan perhitungan nilai jumlah kuadrat atau *Sum of Squared Errors* (SSE). Adapun jenis evaluasi SSE yang diterapkan pada penelitian ini yaitu *Within-Cluster Sum of Squares* (WCSS). WCSS sendiri merupakan ukuran kedekatan antarobyek dalam suatu kluster. Nilai WCSS yang kecil umumnya mengindikasikan kemiripan antarobyek dalam suatu kluster. Sehingga luaran yang didapatkan nantinya berupa nilai k yang memiliki penurunan drastis dan membentuk sudut siku (*elbow*) pada visualisasi grafik..

BAB IV

ANALISIS DAN PERANCANGAN SISTEM

4.1 Analisis Sistem yang Diusulkan

Dalam membangun sistem, perlu dilakukan proses analisis fungsional dan nonfungsional agar dapat mengetahui fungsi-fungsi pada sistem dan peralatan yang dibutuhkan. Pada dasarnya, pencatatan data transaksi maupun data pelanggan dalam perusahaan Starbucks sudah dilakukan secara komputerisasi. Namun data transaksi dan data pelanggan belum dimanfaatkan secara maksimal oleh perusahaan. Contoh pemanfaatan data tersebut yaitu dapat diolah untuk mengetahui pelanggan mana yang akan mendapatkan perlakuan khusus. Misalnya dengan memberikan diskon pada tiap transaksi terhadap pelanggan tersebut. Hal ini dilakukan agar perusahaan dapat mempertahankan hubungan yang baik terhadap pelanggan mereka.

Sistem yang dibangun oleh penulis ini akan digunakan untuk mengelompokkan segmen pelanggan Starbucks menggunakan algoritma *K-Means* dan *Principal Component Analysis* (PCA). Data pelanggan yang akan diproses diperoleh dari situs web Kaggle.com, karena situs web tersebut menyediakan *dataset* yang dibutuhkan dalam penelitian. Pengelompokkan segmen pelanggan dilakukan berdasarkan indikator *frequency* (frekuensi pembelian pelanggan), *monetary* (jumlah yang dibelanjakan oleh pelanggan), dan *tenure* (waktu pendaftaran pelanggan ke aplikasi Starbucks) yang telah diekstraksi dari *dataset* pelanggan Starbucks.

4.1.1 Analisis Fungsional

Analisis fungsional merupakan analisis yang terkait dengan kebutuhan fasilitas yang akan digunakan oleh sistem secara umum. Kebutuhan fungsional dari sistem yang akan dibangun oleh penulis meliputi masukan (*input*), proses, dan keluaran (*output*). Kebutuhan secara fungsional adalah sebagai berikut.

a. Kebutuhan *Input*

Input dataset: kebutuhan ini berupa data pelanggan Starbucks yang akan digunakan pada setiap proses yang ada dalam penelitian.

b. Kebutuhan Proses

1. Proses *preprocessing*, yaitu proses yang dilakukan untuk memahami, membersihkan, dan memformat tipe data dari *dataset* pelanggan Starbucks.
2. Proses *exploratory data analysis* (EDA), yaitu proses yang dilakukan untuk mengeksplorasi dan memvisualisasi *dataset* pelanggan Starbucks.
3. Proses pembobotan FMT, yaitu proses yang dilakukan untuk menghitung nilai untuk masing-masing variabel *frequency*, *monetary*, dan *tenure* (FMT), serta menormalisasi variabel pada *dataset* pelanggan Starbucks.
4. Proses *principal component analysis* (PCA), yaitu proses yang dilakukan untuk mengurangi dimensi (variabel) pada *dataset* pelanggan Starbucks tanpa menghilangkan informasi utama yang termuat dalam *dataset*.
5. Proses *clustering*, yaitu proses yang dilakukan untuk mengelompokkan segmen pelanggan Starbucks ke dalam klaster tertentu. Pada proses ini digunakan algoritma *K-Means* yang mengklaster data berdasarkan jarak obyek ke titik pusat obyek (*centroid*).
6. Proses validasi *clustering*, yaitu proses yang dilakukan untuk menguji seberapa baik performa hasil proses *clustering* dari algoritma *K-Means*. Pada proses ini digunakan metode *Elbow* yang berfungsi dalam penentuan jumlah klaster terbaik.

c. Kebutuhan *Output*

Output dataset: kebutuhan ini berupa hasil segmentasi pelanggan Starbucks dalam bentuk jumlah klaster dan karakteristik pelanggan pada tiap klaster.

4.1.2 Analisis Nonfungsional

Analisis nonfungsional adalah analisis mengenai kebutuhan alat pendukung sistem yang akan dibuat, bertujuan untuk memenuhi kebutuhan nonfungsional sistem. Kebutuhan secara nonfungsional tersebut meliputi

kebutuhan *hardware* (perangkat keras) dan *software* (perangkat lunak) yang digunakan dalam penelitian. Kebutuhan secara nonfungsional adalah sebagai berikut.

a. Perangkat Keras (*Hardware*)

Perangkat keras yang digunakan untuk membangun dan mengimplementasikan sistem yang dibuat adalah laptop dengan spesifikasi sebagai berikut.

1. ASUS X450CA-WX242D
2. Prosesor Intel Celeron 1007U
3. GPU Intel HD Graphics 4000
4. RAM 10 GB
5. HDD 500 GB

b. Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan untuk membangun dan mengimplementasikan sistem yang dibuat adalah laptop dengan spesifikasi sebagai berikut.

1. Sistem operasi Windows 10 64-bit
2. Bahasa pemrograman Python versi 3.9
3. Microsoft Edge Browser versi 103.0.1264.37
4. Google Colaboratory
5. Draw.io

4.2 Desain Sistem

4.2.1 Desain Logis

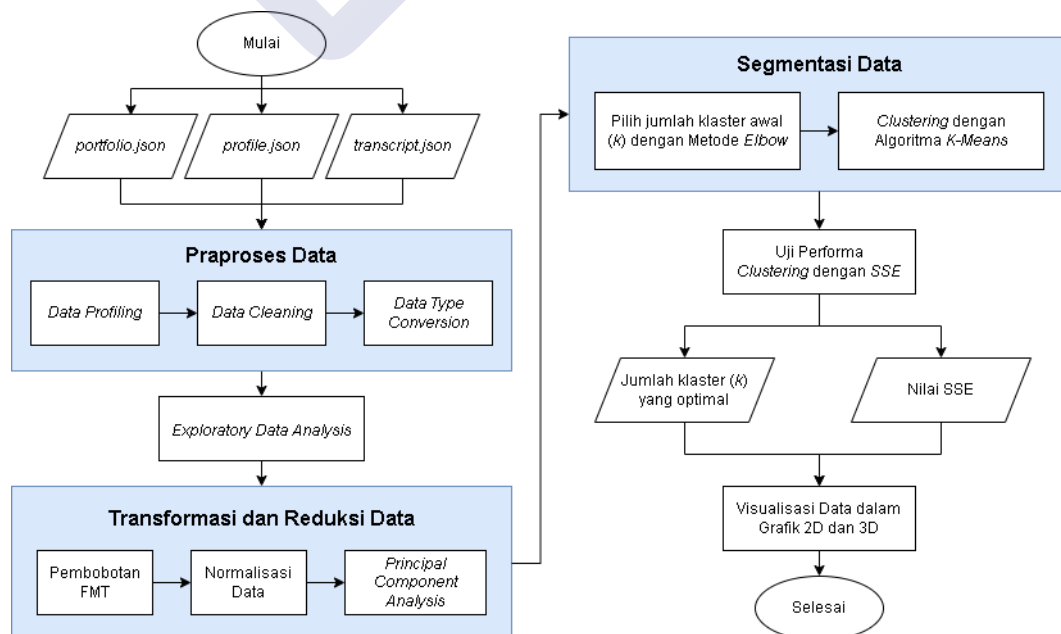
Perancangan sistem dapat diartikan sebagai gambaran atau sketsa dari alur proses sistem pengolahan *dataset*. Sistem ini didesain menggunakan perancangan UML (*Unified Modelling Language*) untuk menggambarkan bagaimana proses dari sistem ini bekerja. Desain perancangan ini, nantinya akan digunakan dalam tahap implementasi ke dalam bahasa pemrograman.

a. *Flowchart*

Dalam perancangan suatu sistem terdapat proses aliran sistem atau *flowchart* sistem. Perancangan sistem ini dilakukan sebelum proses

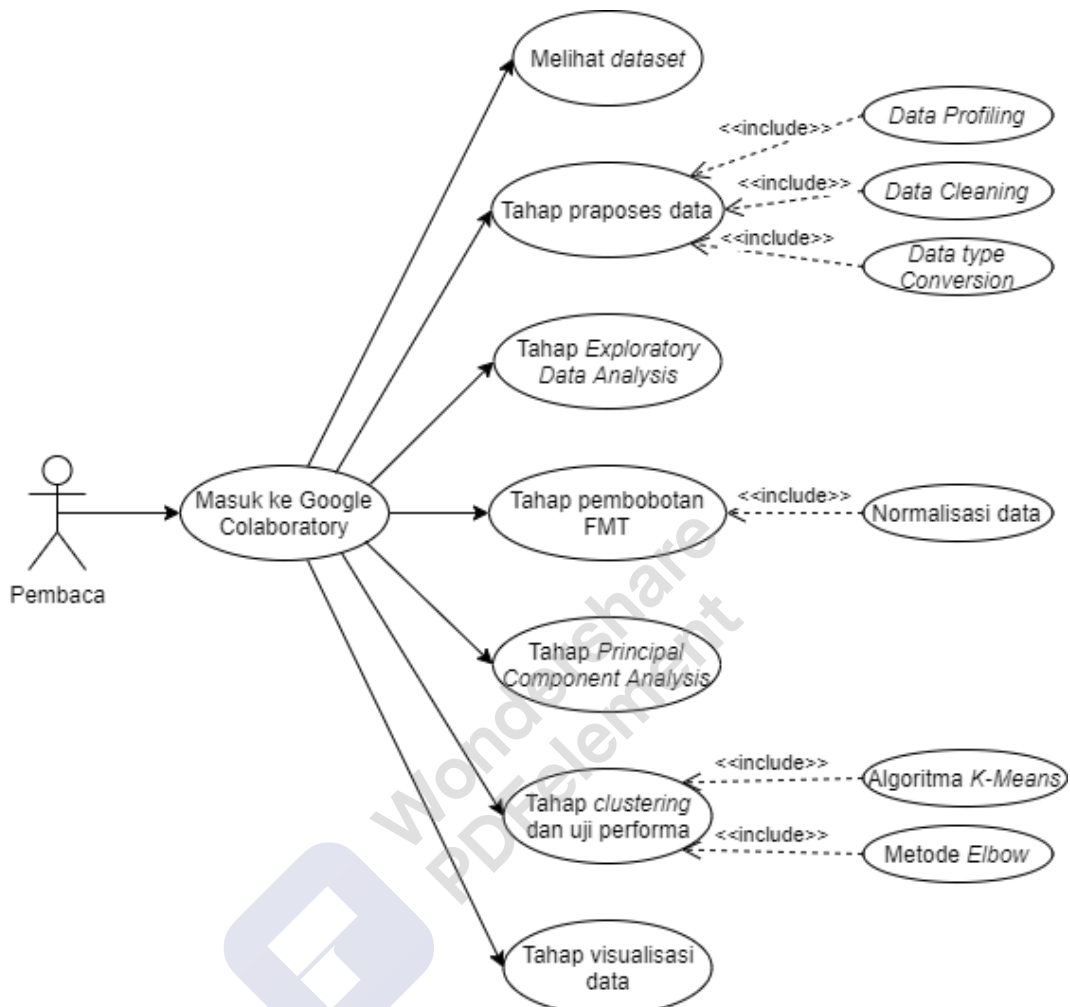
implementasi sistem. *Flowchart* akan menjabarkan metode yang digunakan pada pembuatan sistem segmentasi pelanggan Starbucks yaitu algoritma *K-Means* dan *Principal Component Analysis* (PCA). Algoritma dimulai dari memasukkan data pelanggan Starbucks dengan format *file* JSON, kemudian sistem mulai masuk ke tahap praproses data. Di tahap ini, data akan dilihat metadata-nya, dibersihkan, dan dikonversi tipe data dari beberapa variabel sebelum masuk ke proses berikutnya. Kemudian *dataset* hasil praproses data akan masuk ke proses *Exploratory Data Analysis* (EDA) untuk dieksplorasi fitur-fitur dan diketahui *business insight* seputar *dataset*. Selanjutnya data akan diintegrasikan dan ditransformasi menjadi nilai bobot tertentu pada proses pembobotan FMT, serta dilakukan normalisasi agar tidak tercipta *outlier* (data yang menyimpang).

Setelah itu, data akan masuk ke tahap *Principal Component Analysis* yang mana akan direduksi dimensi data (variabel) dan selanjutnya akan digunakan dalam proses *clustering*. Sebelum masuk ke tahap *clustering*, perlu ditentukan terlebih dahulu nilai k (jumlah kluster). Lalu data akan dilatih untuk bisa menemukan titik pusat data (*centroid*) terdekat dan akhirnya dilakukan proses *clustering* berdasarkan karakteristik yang mirip antardata. Setelah selesai, sistem akan menampilkan hasil *clustering* beserta nilai SSE sebagai nilai performa pada tiap jumlah kluster k . Adapun *flowchart* sistem dapat dilihat pada Gambar 4.1.



Gambar 4.1 Flowchart sistem yang akan dibangun

b. Use Case Diagram



Gambar 4.2 Use case diagram sistem yang akan dibangun

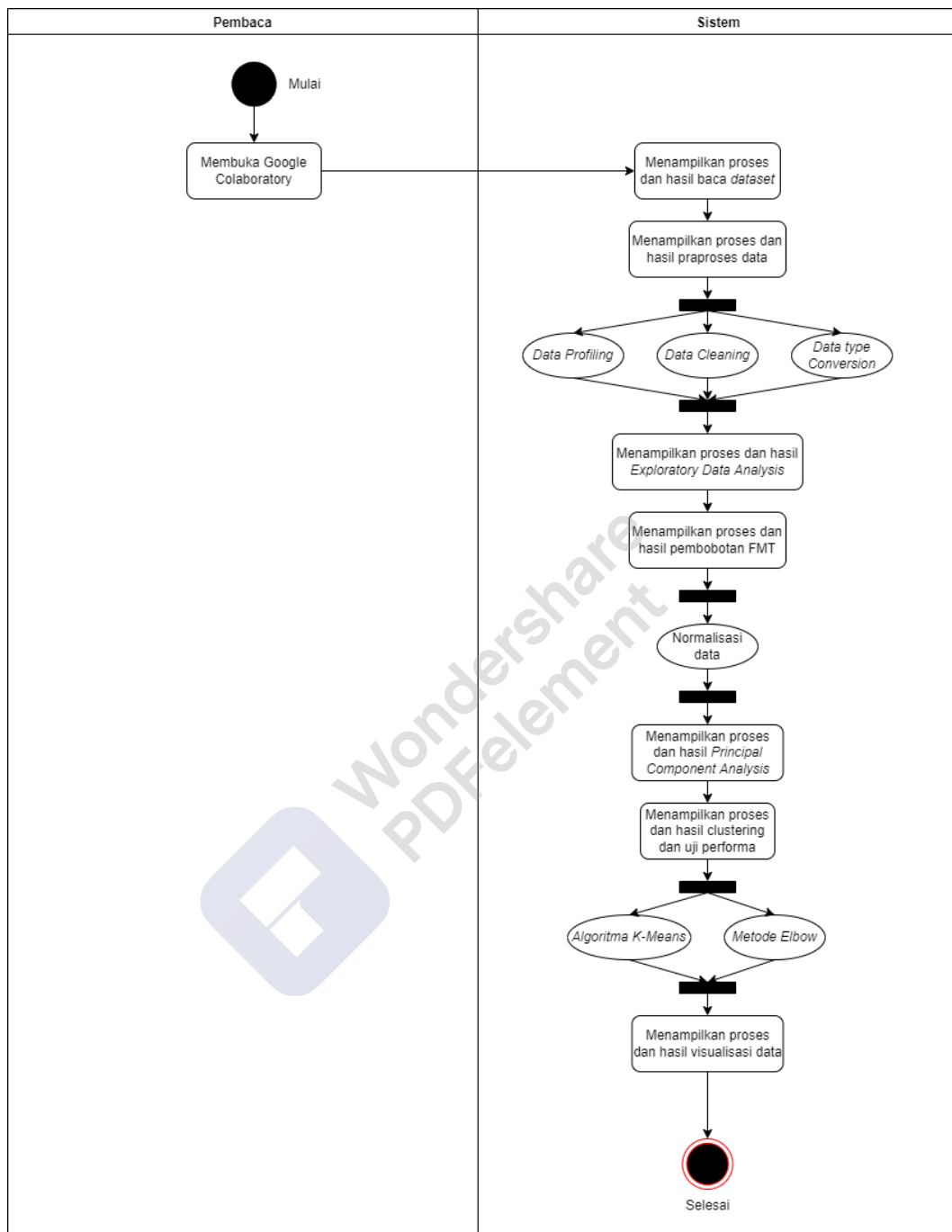
Use case diagram berfungsi untuk menggambarkan rangkaian proses yang saling terkait (berinteraksi) dan membentuk sistem secara teratur yang dilakukan oleh sebuah *actor*. Diagram ini menggambarkan proses system (kebutuhan sistem dari sudut pandang *user*). Pada Gambar 4.2 dapat diketahui bahwa hanya terdapat satu pengguna yang akan berperan dalam sistem yaitu pembaca. Pembaca memiliki beberapa peran yang bisa dilakukan antara lain:

1. Pembaca dapat melihat *dataset* yang diimpor dan dibaca oleh sistem.
2. Pembaca dapat melihat proses dan hasil dari tahap praproses data. Tahap ini memiliki tiga subproses meliputi *data profiling*, *data cleaning*, dan *data type conversion*.

3. Pembaca dapat melihat proses dan hasil dari tahap *Exploratory Data Analysis* (EDA).
4. Pembaca dapat melihat proses dan hasil dari tahap pembobotan FMT (*Frequency*, *Monetary*, dan *Tenure*). Tahap ini memiliki satu subproses yaitu normalisasi data.
5. Pembaca dapat melihat proses dan hasil dari tahap *Principal Component Analysis* (PCA).
6. Pembaca dapat melihat proses dan hasil dari tahap *clustering* dan uji performa. Tahap ini memiliki dua subproses meliputi *clustering* menggunakan algoritma *K-Means* dan uji performa sistem menggunakan metode *Elbow*.
7. Pembaca dapat melihat proses dan hasil dari tahap visualisasi data.

c. Activity Diagram

Activity diagram menggambarkan aktivitas-aktivitas yang terjadi dalam implementasi algoritma *K-Means* dengan *Principal Component Analysis* untuk segmentasi pelanggan aplikasi Starbucks. *Activity diagram* adalah representasi grafis dari seluruh tahapan alur kerja. Diagram ini mengandung aktivitas, pilihan tindakan, perulangan, dan hasil dari aktivitas tersebut. Pada pemodelan UML, diagram ini dapat digunakan untuk menjelaskan proses bisnis dan alur kerja operasional secara langkah demi langkah dari komponen suatu sistem. Adapun *activity diagram* dari sistem ini dapat dilihat pada Gambar 4.3.



Gambar 4.3 Activity diagram sistem yang akan dibangun

Pada Gambar 4.3 dapat dijelaskan bahwa alur aktivitas yang terjadi dimulai dari pembaca yang membuka proyek sistem segmentasi pelanggan aplikasi Starbucks melalui program Google Colaboratory. Selanjutnya proses akan dilanjutkan oleh sistem yang melakukan pembacaan *dataset*. Kemudian sistem akan melakukan praproses data pada *dataset* yang telah dibaca. Pada tahap

praproses data ini, *dataset* nantinya akan melewati tiga subproses meliputi *data profiling* (memberikan informasi singkat terkait *dataset*), *data cleaning* (membersihkan *outlier* pada *dataset*), dan *data type conversion* (pengubahan tipe data variabel pada *dataset*). Setelah melalui praproses data, lalu sistem akan melakukan *Exploratory Data Analysis* (EDA). Pada tahap ini, akan dicari dan diekstraksi *insight* atau informasi yang berguna untuk kebutuhan bisnis perusahaan.

Pada tahap berikutnya, sistem akan melakukan proses pembobotan model FMT (*Frequency*, *Monetary*, dan *Tenure*). Pada tahap ini *dataset* hasil eksplorasi EDA akan melalui proses normalisasi data untuk menstandarisasi satuan ukuran variabel. Selanjutnya *dataset* akan melalui proses *Principal Component Analysis* (PCA) untuk mereduksi jumlah dimensi pada *dataset*. Kemudian sistem akan melakukan *clustering* dengan algoritma *K-Means* dan uji performa sistem *clustering* dengan metode *Elbow*. Setelah data diklaster ke dalam sejumlah klaster tertentu, terakhir sistem akan memproses data tersebut untuk ditampilkan dalam bentuk visualisasi data seperti *histogram*, *scatter plot*, *count plot*, dan lain-lain.

4.2.2 Desain Fisik

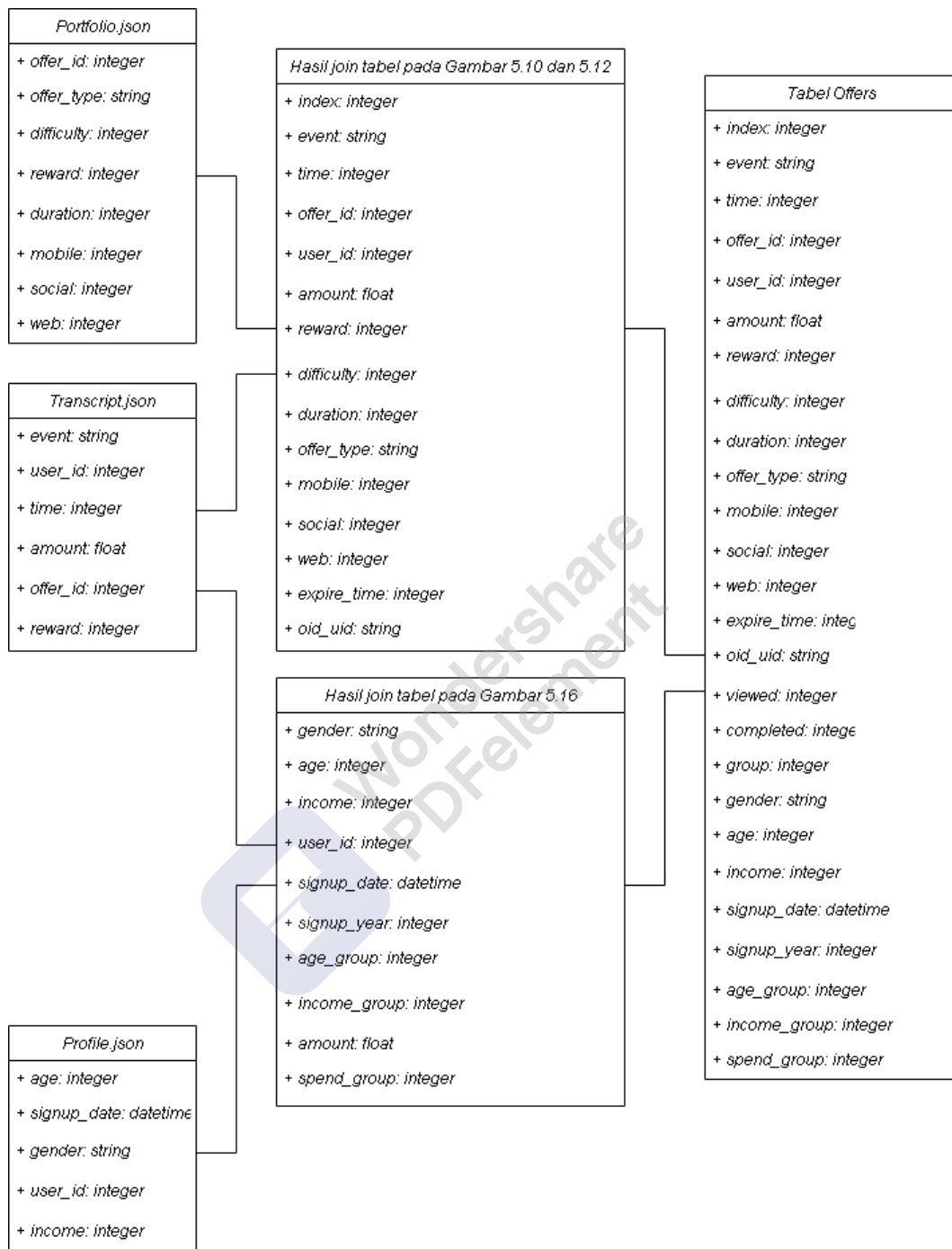
Pada penelitian ini, tidak dilakukan perancangan tampilan antarmuka sistem yang akan dibuat karena sistem berupa analisis yang memuat kode sintaksis pada bahasa pemrograman Python. Untuk itu, penulis merancang sebuah *class diagram* yang akan menjelaskan hubungan antar entitas tabel dari *dataset*. Kemudian untuk struktur *dataset* yang digunakan dalam penelitian ini berupa JSON (*JavaScript Object Notation*). JSON merupakan sebuah format penukaran data yang mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemrograman JavaScript. Format JSON adalah berbasis teks, dapat terbaca oleh manusia, digunakan untuk mempresentasikan struktur data sederhana, dan tidak bergantung dengan bahasa pemrograman apapun. Biasanya digunakan pada aplikasi AJAX (*Asynchronous JavaScript and XML*). Format JSON sering digunakan untuk mentransmisikan data terstruktur melalui koneksi jaringan. Secara umum, JSON

digunakan untuk mentransmisikan data antara server dan aplikasi web. Jenis media internet yang resmi untuk JSON adalah aplikasi/json.

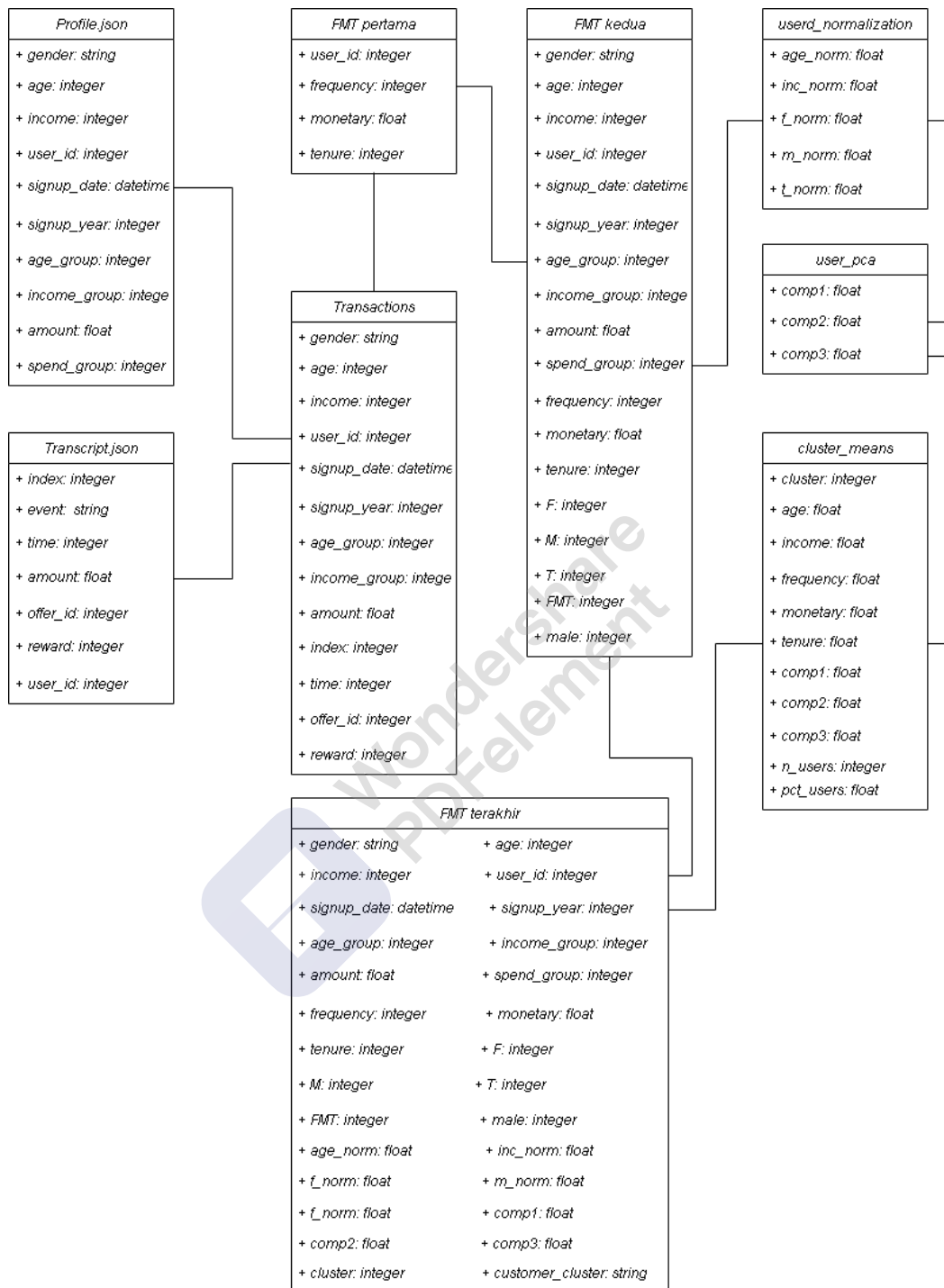
JSON terbuat dari dua struktur utama yaitu: pertama, kumpulan pasangan nama atau nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai obyek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel *hash* (*hash table*), daftar berkunci (*keyed list*), atau *associative array*. Kedua, daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*). Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemrograman modern mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data JSON mudah dipertukarkan dengan bahasa-bahasa pemrograman apa pun. Selain itu, JSON juga memiliki beberapa keunggulan dibandingkan dengan XML. Beberapa kelebihan JSON antara lain: JSON lebih ringkas, cepat, dan mudah; JSON tidak menggunakan tag penutup; JSON lebih cepat untuk dibaca dan ditulis; JSON dapat menggunakan *array*; dan JSON tidak perlu menggunakan *parser* (penerjemah) khusus untuk diubah menjadi sebuah obyek, melainkan hanya perlu menggunakan fungsi dasar dari JavaScript. Adapun struktur data JSON pada *dataset* yang digunakan dalam penelitian ini dapat dilihat pada Lampiran 5.

a. Class Diagram

Class diagram digunakan untuk menggambarkan kumpulan dari entitas tabel dan hubungannya. Diagram ini merupakan diagram yang paling umum ditemukan dalam pemodelan sistem berorientasi obyek. Entitas tabel memiliki tiga area utama yaitu nama entitas tabel, atribut, dan metode. Selain itu, setiap entitas tabel yang ada dapat menjadi sebuah formulir saat pembuatan sistem. Adapun *class diagram* untuk sistem segmentasi pelanggan aplikasi Starbucks menggunakan algoritma *K-Means* dengan *Principal Component Analysis* (PCA) dapat dilihat pada Gambar 4.4.



Gambar 4.4 Class diagram sistem yang akan dibangun



Lanjutan Gambar 4.4 Class diagram sistem yang akan dibangun

Pada Gambar 4.4 dapat diketahui alur proses dari awal (tahap pembacaan *dataset*) hingga akhir (tahap *Exploratory Data Analysis*). Entitas tabel awal terdiri atas tiga *dataset* meliputi *portfolio.json*, *profile.json*, dan *transcript.json*.

Kemudian pada tahap *Exploratory Data Analysis* dilakukan penggabungan tabel *portfolio.json* dengan *profile.json* yang nantinya akan menghasilkan tabel pada Gambar 5.10 dan Gambar 5.12. Selanjutnya, juga dilakukan penggabungan tabel *profile.json* dengan *transcript.json* yang nantinya akan menghasilkan tabel pada Gambar 5.16. Lalu kedua tabel hasil penggabungan tersebut kembali digabung sehingga menghasilkan entitas tabel akhir bernama *offers*.

Pada Gambar 4.5 dapat diketahui alur proses dari awal (tahap pembobotan model FMT) hingga akhir (tahap *clustering* dan uji performa). Entitas tabel awal terdiri atas dua *dataset* yaitu *profile.json* dan *transcript.json*. Berikutnya pada tahap pembobotan model FMT dilakukan penggabungan antara kedua tabel tersebut sehingga menghasilkan tabel bernama *transactions*. Setelah itu, dari tabel *transactions* dilakukan transformasi data melalui tahap pembobotan model FMT sehingga menghasilkan tabel bernama FMT pertama. Pada tahap ini nantinya akan dilakukan proses normalisasi data untuk menstandarisasi satuan ukuran. Namun sebelumnya akan dilakukan penggabungan tabel *profile.json* dengan FMT pertama sehingga menghasilkan tabel bernama FMT kedua.

Pada proses normalisasi data, tabel FMT kedua akan ditransformasi nilai datanya sehingga memiliki satuan ukuran yang sama. Satuan ukuran ini perlu diseragamkan karena dapat memengaruhi hasil *clustering*. Tabel hasil transformasi tersebut diberi nama *userd_normalization*. Kemudian data tersebut akan melalui proses *Principal Component Analysis* (PCA) yang berfungsi untuk mereduksi jumlah dimensi data. Tabel hasil transformasi pada proses ini bernama *user_pca*. Selanjutnya data akan masuk ke tahap *clustering* dan uji performa. Pada tahap ini data hasil segmentasi pelanggan aplikasi Starbucks diintegrasikan untuk diketahui informasi statistik secara sederhana. Tabel ini dinamai dengan *segment_means* karena menampilkan informasi nilai rata-rata pada tiap variabel yang digunakan oleh sistem. Lalu dilakukan penggabungan antara tabel FMT kedua dengan tabel *cluster_means* sehingga menghasilkan entitas tabel akhir bernama FMT terakhir.

BAB V IMPLEMENTASI DAN HASIL

5.1 Implementasi

5.1.1 Proses Data Preprocessing

Pada tahap *data preprocessing*, terdapat tiga proses utama yang akan dilakukan yaitu *data profiling* (memahami *dataset*), *data cleaning* (membersihkan *dataset*), dan *data type conversion* (mengkonversi tipe data pada *dataset*). *Data profiling* adalah suatu proses menganalisis data mentah dengan tujuan memahami informasi yang terkandung di dalam *dataset*. Luaran dari proses ini yaitu informasi deskriptif yang berguna untuk proses-proses manipulasi data berikutnya. Sebelum melakukan *data profiling*, penulis mengkonfigurasi beberapa *library* Python yang berfungsi agar sistem dapat berjalan dengan baik. Selain itu, dilakukan *read* pada *dataset* yang akan diproses untuk *data profiling*. Pada *source code* sistem, terdapat beberapa *library* Python yang digunakan oleh penulis untuk mengembangkan sistem segmentasi pelanggan antara lain: json, numpy, pandas, plotly, matplotlib, dan seaborn.

Setelah mengkonfigurasi *library* Python pada sistem, penulis melakukan *read* terhadap tiga *dataset* yang telah dipersiapkan. Tiga *dataset* tersebut antara lain: *portfolio.json*, *profile.json*, dan *transcript.json*. Kemudian dilakukan *data profiling* untuk mengetahui informasi tertentu yang terkandung dalam *dataset*. *Data profiling* membantu penulis menemukan, memahami, dan mengatur data sehingga sudah seharusnya menjadi bagian penting dalam menangani dan mengolah data. *Data profiling* merupakan langkah pertama yang perlu dilakukan jika ingin membuat keputusan yang lebih baik dengan data.

Setelah diketahui informasi yang terkandung dalam *dataset*, selanjutnya yaitu melakukan *data cleaning*. *Data cleaning* adalah sebuah proses menghapus atau memperbaiki data yang kosong, duplikat, tidak valid, atau salah format. Ada dua jenis metode yang bisa digunakan untuk membersihkan data yaitu menggunakan fungsi standar yang terdapat pada *library* pandas maupun

mendefinisikan fungsi baru agar dapat melakukan kustomisasi fitur. Pada penelitian ini, penulis mendefinisikan fungsi baru untuk membersihkan data karena ingin mengeksplorasi informasi lain yang tidak bisa dilakukan oleh fungsi standar.

Setelah membuat fungsi baru, penulis akan menerapkan fungsi tersebut untuk menampilkan jumlah data yang memiliki nilai kosong. Saat penulis melakukan *data profiling*, ditemukan bahwa hanya *dataset profile.json* yang memiliki nilai NaN (*Not a Number*) atau nilai yang hilang. Maka dari itu, penulis akan menerapkan fungsi baru hanya untuk *dataset profile.json*. Setelah diketahui jumlah data yang kosong, penulis akan membersihkan data dengan cara menghapus baris data (*record*) tersebut dengan fungsi standar Python. Namun dikarenakan ada beberapa ID pengguna dari *dataset profile.json* yang berkaitan dengan ID penawaran dari *dataset transcript.json*, maka ID penawaran tersebut juga akan dihapus.

Setelah melakukan pembersihan terhadap *dataset*, proses yang akan dilakukan berikutnya yaitu ekstraksi nilai *unique* dari *dataset*. Pada *dataset* yang digunakan dalam penelitian, terdapat nilai yang masih berada dalam bentuk *list* atau *dictionary* sehingga nilai tersebut perlu diekstraksi. Sebelum melakukan ekstraksi nilai *unique* dari *dataset*, penulis mendefinisikan dua fungsi yaitu *get_unique_values()* untuk mendapatkan nilai *unique* dari sebuah kolom yang memiliki data berupa *list* atau *dictionary*, dan *extract_from_iterable_col()* untuk mengekstraksi nilai dari sebuah kolom yang memiliki data berupa *list* atau *dictionary*. Setelah kedua fungsi dibuat, langkah selanjutnya yaitu menerapkan fungsi tersebut ke *dataset* yang memiliki data berupa *list* atau *dictionary*. Saat melakukan proses *data profiling* sebelumnya, diketahui bahwa *dataset portfolio.json* dan *transcript.json* yang memiliki data berupa *list* atau *dictionary*. Oleh karena itu, penulis akan menerapkan kedua fungsi yang telah dibuat terhadap dua *dataset* tersebut.

Setelah mendapatkan nilai *unique* dari *dataset* dan mengekstraksi data, selanjutnya adalah mengkonversi kolom ID pada ketiga *dataset* yang awalnya bertipe data *string* menjadi *integer* (numerik). Konversi tipe data ini diperlukan

karena operasi statistik akan lebih mudah jika data yang digunakan berupa angka. Sebelum melakukan konversi tipe data, penulis menyortir data pada *dataset portfolio.json* berdasarkan tingkat kesulitan penawaran. Lalu penulis akan membuat dua fungsi, *create_map_dict()* dan *map_ids_to_num()*, yang nantinya akan digunakan dalam proses konversi tipe data dari kolom ID terhadap tiga *dataset*. Langkah selanjutnya yang perlu dilakukan ialah menerapkan fungsi *mapping* dan konversi tipe data terhadap tiga *dataset* yang ada dalam penelitian ini. Pertama, penulis akan menerapkan fungsi *mapping* untuk membuat data bertipe data *dictionary* yang memuat kumpulan ID dari *dataset* tertentu. Kedua, penulis akan menerapkan fungsi konversi tipe data untuk mengubah tipe data dari kolom ID yang awalnya *string* menjadi *integer*.

Setelah melakukan banyak manipulasi data terhadap tiga *dataset* yang ada dalam penelitian ini, penulis akan memeriksa kembali apakah ada data yang kosong untuk menjaga kualitas *dataset*. Untuk *dataset* yang akan diperiksa kembali yaitu *transcript.json* karena memuat data peristiwa yang juga berkaitan dengan data penawaran dan data pengguna. Penulis akan memanfaatkan fungsi *show_missing_values()* yang telah dibuat sebelumnya. Selain itu, penulis juga akan menampilkan jumlah data yang kosong untuk setiap jenis peristiwa dari *dataset transcript.json*. Apabila data sudah ditampilkan, selanjutnya penulis akan mengisi data yang kosong tersebut dengan nilai 0 agar diketahui bahwa data tersebut kosong akibat tidak ada peristiwa yang terjadi, bukan karena kelalaian sistem.

Setelah memeriksa data yang kosong, proses yang akan dilakukan selanjutnya ialah mengkonversi beberapa kolom yang masih menggunakan tipe data yang kurang tepat, contohnya kolom *became_member_on* dari *dataset profile.json* yang bertipe data *integer*. Konversi tipe data ini merupakan salah satu hal penting dalam *data preprocessing* karena tipe data ini akan memengaruhi kelancaran manipulasi data yang akan dilakukan pada proses-proses selanjutnya. Namun pada proses ini, hanya *dataset portfolio.json* yang tidak akan dilakukan konversi tipe data karena tidak ada anomali pada struktur *dataset*.

Langkah terakhir yang akan dilakukan dalam tahap *data preprocessing* yaitu memeriksa data duplikat. Duplikasi merupakan salah satu masalah yang penting untuk diperhatikan karena dapat menyebabkan inkonsistensi pada *dataset*. Oleh karena itu, penulis akan menangani masalah dengan cara menghapus data duplikat tersebut. Dengan menghapus data duplikat tidak akan memengaruhi hilangnya informasi utama dari *dataset* sehingga hal ini aman untuk dilakukan. Mirip dengan proses konversi tipe data, pada proses menghapus data duplikat, hanya dua *dataset* yang digunakan yakni *profile.json* dan *transcript.json*.

5.1.2 Proses Exploratory Data Analysis

Pada tahap *Exploratory Data Analysis*, terdapat dua jenis eksplorasi yang akan dilakukan oleh penulis yakni pada data penawaran dan data pengguna. Eksplorasi pada data penawaran akan melalui empat proses utama meliputi mencari tahu berapa banyak penawaran yang dikirimkan dari tiap tipe penawaran, berapa banyak penawaran *reward* yang diselesaikan, berapa banyak penawaran informasional yang diikuti dengan transaksi pengguna, dan tipe penawaran apa yang memiliki tingkat penyelesaian paling tinggi. Untuk data pengguna, eksplorasi akan dilakukan melalui tiga proses utama meliputi mencari tahu bagaimana distribusi demografi pengguna, apakah ada pola tertentu pada *user spending*, dan apakah ada pola tertentu pada penyelesaian penawaran oleh pelanggan.

Pada proses pertama, penulis ingin mencari tahu berapa banyak jumlah penawaran yang telah dikirimkan berdasarkan tipe penawaran. Dalam *marketing channels* Starbucks, ada tiga tipe penawaran yang dikirimkan oleh tim *marketing* Starbucks antara lain penawaran *discount*, penawaran BOGO (*Buy One Get One*), dan penawaran informasional. Adapun penjelasan detail terkait tipe-tipe penawaran yang dikirimkan oleh tim *marketing* Starbucks disajikan pada Tabel 5.1.

Tabel 5.1 Deskripsi terkait tipe-tipe penawaran

No.	Tipe Penawaran	Deskripsi Penawaran
1.	<i>Discount</i>	Penawaran ini berisi iklan terkait diskon produk dengan persentase tertentu. Contoh: <i>Special Only on Monday, Regular Size with 10% Discount.</i>
2.	BOGO	Penawaran ini berisi iklan terkait beli satu produk dan dapatkan satu produk lagi secara gratis. Contoh: <i>Buy One Big Size, Get One Free Regular Size.</i>
3.	Informasional	Penawaran ini berisi iklan terkait informasi yang bersifat edukasi kepada pelanggan Starbucks. Contoh: <i>Coffee drinkers tend to live longer. Research has linking moderate consumption (about three to four cups per day) with a longer life span.....</i>

Untuk mencari tahu jumlah penawaran yang telah dikirimkan kepada pelanggan Starbucks, penulis membuat fungsi yang akan menggabungkan *dataset transcript.json* dengan *portfolio.json*, lalu akan diperiksa apakah peristiwa *offer received* ada dalam *dataset* hasil penggabungan. Setelah diketahui bahwa data tersebut ada, akan dilakukan perhitungan terhadap jumlah peristiwa *offer received* untuk mengetahui jumlah penawaran yang telah dikirimkan. Pada proses kedua, penulis ingin mencari tahu berapa banyak penawaran *reward* yang telah diselesaikan oleh pelanggan Starbucks. Penawaran *reward* di sini merujuk pada penawaran *discount* dan BOGO. Penawaran informasional tidak termasuk dalam penawaran *reward* karena tidak memberikan atribut *reward* setelah pelanggan menyelesaikan penawaran tersebut. Selain mencari tahu berapa banyak penawaran *reward* yang diselesaikan, penulis juga akan mencari tahu berapa banyak penawaran *reward* yang dilihat. Sebab terkadang pelanggan Starbucks hanya melihat penawaran tanpa menyelesaikannya, atau tidak melihat penawaran namun menyelesaikannya. Hal tersebut tentu akan menjadi *insight* tersendiri bagi perusahaan. Untuk memperoleh jumlah penawaran *reward*, penulis kembali

melakukan penggabungan *dataset transcript.json* dan *portfolio.json*, seperti yang dilakukan pada proses pertama namun dengan filter atribut yang berbeda. Setelah itu, penulis juga menghapus data penawaran informasional karena tidak dibutuhkan dalam proses perhitungan. Kemudian penulis membuat fungsi *concat_str_cols()* untuk menggabungkan nilai bertipe data *string* yang diperoleh dari beberapa kolom pada *dataset* tertentu.

Setelah membuat fungsi untuk penggabungan *string*, selanjutnya penulis akan membuat *lambda expression* untuk menghitung waktu kedaluwarsa dari penawaran *reward* yang telah dikirimkan kepada pelanggan. *Lambda expression* adalah sebuah blok kode pendek yang memuat parameter dan mengembalikan nilai tertentu. Dengan menghitung waktu kedaluwarsa penawaran *reward*, tim *marketing* Starbucks menjadi tahu seberapa efektif penawaran *reward* tersebut dapat memengaruhi daya beli pelanggan. Adapun *lambda expression with conditional statement* yang diterapkan untuk menghitung waktu kedaluwarsa penawaran *reward* dapat dilihat pada Tabel 5.2.

Tabel 5.2 *Lambda expression with conditional statement* untuk mengetahui waktu kedaluwarsa penawaran *reward*

No.	Kondisi	<i>Lambda Expression</i>
1.	IF peristiwa == “offer received”	$expire_time = duration \times 24 + time$
2.	ELSE	$expire_time = -1$

Keterangan:

duration = durasi penawaran sebelum kedaluwarsa (dalam satuan hari)

time = waktu yang dihitung sejak penawaran dilihat oleh pelanggan

Setelah waktu kedaluwarsa penawaran *reward* dihitung, selanjutnya penulis akan menyatukan kolom ID penawaran dan ID pengguna menggunakan fungsi *concat_str_cols()* yang telah dibuat sebelumnya. Lalu penulis membuat fungsi baru yaitu *fill_with()* yang akan digunakan untuk mengisi nilai apakah sebuah penawaran *reward* sudah dilihat atau sudah diselesaikan. Fungsi tersebut akan mengisi nilai dengan angka biner antara 0 atau 1, melalui persyaratan jika penawaran tersebut dilihat atau diselesaikan akan diisi dengan angka 1 dan berlaku sebaliknya. Setelah fungsi *fill_with()* dibuat, langkah selanjutnya yang

akan dilakukan adalah membuat fungsi *go_to_event()* yang berfungsi untuk proses iterasi dalam penentuan apakah sebuah penawaran *reward* sudah diterima, sudah dilihat, atau sudah diselesaikan. Lalu penulis menginisialisasi nilai awal dari variabel *viewed* dan *completed* dengan angka -1 sebelum masuk ke proses iterasi. Dalam proses iterasi, penulis akan menerapkan kedua fungsi yang telah dibuat.

Setelah dilakukan penentuan apakah sebuah penawaran *reward* telah dilihat (*viewed*) atau telah diselesaikan (*completed*), penulis akan menampilkan ringkasan data dari hasil proses tersebut. Lalu penulis membuat fungsi baru, *group_offer()* yang berfungsi untuk mengelompokkan penawaran *reward* berdasarkan masing-masing variabel *viewed* dan *completed*. Setelah fungsi dibuat, kemudian dilakukan penerapan fungsi sehingga terbentuk kolom baru bernama *group*. Untuk penjelasan detail terkait tiap grup penawaran *reward* akan disajikan pada Tabel 5.3.

Tabel 5.3 Penjelasan dari tiap grup penawaran *reward*

No.	Nama Grup	Penjelasan
1.	<i>Group 1</i>	Penawaran <i>reward</i> tidak dilihat dan tidak diselesaikan
2.	<i>Group 2</i>	Penawaran <i>reward</i> dilihat, tapi tidak diselesaikan
3.	<i>Group 3</i>	Penawaran <i>reward</i> diselesaikan, tapi tidak dilihat
4.	<i>Group 4</i>	Penawaran <i>reward</i> dilihat dan diselesaikan

Setelah data penawaran *reward* dikelompokkan ke dalam grup tertentu, selanjutnya ialah memvisualisasikan data tersebut ke dalam bentuk grafis agar menjadi lebih mudah dipahami oleh *stakeholder* terkait. Data akan disajikan dalam *stacked bar chart* dengan tujuan untuk membandingkan data antarvariabel kategoris. Panjang dari setiap *bar* merepresentasikan seberapa banyak data tersebut berkontribusi dari total keseluruhan data. Untuk membuat visualisasi data dalam bentuk *stacked bar chart*, penulis merancang fungsi baru yakni *show_group_count()*. Setelah mengetahui data penawaran *reward* yang dilihat dan diselesaikan, pada proses ketiga, penulis akan mencari tahu seberapa banyak penawaran informasional yang diikuti dengan sebuah transaksi.

Karena penawaran informasional tidak memberikan atribut *reward* setelah pelanggan menyelesaikan transaksi, maka perhitungannya akan sedikit berbeda dengan penawaran *reward*. Penawaran informasional akan dihitung valid jika penawaran tersebut memiliki jumlah minimum pembelian selama durasi penawaran berlangsung. Penulis akan kembali melakukan penggabungan *dataset transcript.json* dan *portfolio.json* namun dengan filter atribut yang berbeda. Kemudian beberapa kolom dari *dataset* hasil penggabungan tersebut dikonversi tipe datanya menjadi *integer* untuk memudahkan dalam manipulasi data. Lalu dilakukan seleksi data penawaran berdasarkan ID pengguna yang memiliki atribut peristiwa tidak sama dengan *transaction*, sehingga bisa diperoleh data penawaran informasional yang unik dari tiap pelanggan.

Setelah membuat melakukan beberapa proses manipulasi data pada proses sebelumnya, langkah berikutnya yang akan dilakukan yaitu membuat *lambda expression* untuk menghitung waktu kedaluwarsa dari penawaran informasional yang telah dikirimkan kepada pelanggan. *Lambda expression* adalah sebuah blok kode pendek yang memuat parameter dan mengembalikan nilai tertentu. Dengan menghitung waktu kedaluwarsa penawaran informasional, tim *marketing* Starbucks menjadi tahu seberapa efektif penawaran informasional tersebut dapat memengaruhi daya beli pelanggan. Adapun *lambda expression with conditional statement* yang diterapkan untuk menghitung waktu kedaluwarsa penawaran informasional dapat dilihat pada Tabel 5.4.

Tabel 5.4 *Lambda expression with conditional statement* untuk mengetahui waktu kedaluwarsa penawaran informasional

No.	Kondisi	<i>Lambda Expression</i>
1.	IF peristiwa == " <i>transaction</i> "	<i>expire_time</i> = -1
2.	ELSE	<i>expire_time</i> = <i>duration</i> × 24 + <i>time</i>

Keterangan:

duration = durasi penawaran sebelum kedaluwarsa (dalam satuan hari)

time = waktu yang dihitung sejak penawaran dilihat oleh pelanggan

Setelah waktu kedaluwarsa penawaran informasional dihitung, selanjutnya penulis akan menyatukan kolom ID penawaran dan ID pengguna menggunakan

fungsi `concat_str_cols()` yang telah dibuat sebelumnya. Lalu penulis menginisialisasi nilai awal dari variabel *viewed* dan *completed* dengan angka -1 sebelum masuk ke proses iterasi. Dalam proses iterasi, penulis akan menerapkan kedua fungsi yang telah dibuat. Setelah atribut *viewed* dan *completed* dari data penawaran informasional ditentukan, selanjutnya ialah mengelompokkan penawaran informasional ke dalam grup tertentu dan memvisualisasikan data tersebut ke dalam bentuk grafis agar menjadi lebih mudah dipahami oleh *stakeholder* terkait. Data akan disajikan dalam *stacked bar chart* dengan tujuan untuk membandingkan data antarvariabel kategoris. Panjang dari setiap *bar* merepresentasikan seberapa banyak data tersebut berkontribusi dari total keseluruhan data. Untuk membuat visualisasi data dalam bentuk *stacked bar chart*, penulis menerapkan fungsi yang telah dibuat sebelumnya yakni `show_group_count()`. Untuk penjelasan detail terkait tiap grup penawaran informasional dapat dilihat pada Tabel 5.5.

Tabel 5.5 Penjelasan dari tiap grup penawaran informasional

No.	Nama Grup	Penjelasan
1.	<i>Group 1</i>	Penawaran informasional tidak dilihat dan tidak diselesaikan
2.	<i>Group 2</i>	Penawaran informasional dilihat, tapi tidak diselesaikan
3.	<i>Group 3</i>	Penawaran informasional diselesaikan, tapi tidak dilihat
4.	<i>Group 4</i>	Penawaran informasional dilihat dan diselesaikan

Pada proses keempat, penulis akan mencari tahu tipe penawaran mana yang memiliki tingkat penyelesaian paling tinggi. Oleh karena itu, penulis menggabungkan kembali data penawaran *reward* dan penawaran informasional yang sebelumnya dipisah. Kemudian penulis melakukan filter untuk memperoleh data valid dengan mengabaikan data yang berasal dari *Group 3*, penawaran diselesaikan tapi tidak dilihat. Selanjutnya penulis juga melakukan filter dengan

hanya memilih data yang berasal dari *Group 4*, penawaran dilihat dan diselesaikan. Pada langkah berikutnya, penulis melakukan penggabungan untuk dua data hasil filtrasi tersebut yang akan menjadi data untuk perhitungan nantinya. Lalu penulis menghitung persentase tingkat penyelesaian penawaran dengan mengalikan jumlah penawaran yang diselesaikan (tiap tipe penawaran) dengan angka 100 lalu dibagi dengan jumlah total penawaran (tiap tipe penawaran).

Setelah mengetahui persentase tingkat penyelesaian penawaran, selanjutnya penulis akan melakukan penggabungan data tingkat penyelesaian penawaran dengan *dataset portfolio.json* untuk memperoleh metadata penawaran. Selain itu, penulis juga membentuk kolom baru yaitu *offer* (berisi tipe penawaran dan masa berlaku penawaran) dan *summary* (berisi jumlah penawaran yang diterima dan diselesaikan). Kedua kolom tersebut memperoleh masukan dari data tingkat penyelesaian penawaran dan memanfaatkan fungsi *concat_str_cols()* untuk penggabungan nilai bertipe data *string*. Lalu setelah semua data selesai dipersiapkan, penulis akan memvisualisasikan data tersebut ke dalam bentuk *horizontal bar plot*. Untuk membentuk sebuah diagram batang, akan digunakan fungsi *barplot()* yang terdapat pada *library* seaborn.

Setelah selesai melakukan tahap *Exploratory Data Analysis* pada data penawaran, selanjutnya penulis akan mengeksplorasi beberapa *insight* yang dapat diperoleh dari data pengguna. Untuk *insight* pertama yang bisa dieksplorasi yaitu mencari tahu distribusi demografi pengguna Starbucks. Pada langkah awal, penulis akan menambahkan kolom *signup_year* untuk mengetahui pada tahun berapa pelanggan tersebut mulai menjadi pengguna Starbucks. Kemudian penulis membuat fungsi baru untuk visualisasi data yakni *plot_user_demographic()*. Fungsi tersebut nantinya akan membentuk representasi data berupa histogram. Histogram merupakan salah satu jenis visualisasi data yang berfungsi untuk menampilkan distribusi dari sekelompok data. Dengan menggunakan histogram, penulis juga dapat mengetahui frekuensi data pada rentang tertentu. Frekuensi data yang ada pada tiap kelompok data direpresentasikan dengan bentuk grafik diagram batang. Adapun rentang dari data pengguna dapat dilihat pada Tabel 5.6.

Tabel 5.6 Rentang data demografi pengguna Starbucks

No.	Variabel	Rentang Data
1.	<i>Gender</i>	F (<i>Female</i>), M (<i>Male</i>), O (<i>Other</i>)
2.	<i>Age</i>	18 – 101 tahun
3.	<i>Signup Year</i>	2013 – 2018
4.	<i>Income</i>	USD30k – USD 120k

Setelah mengetahui distribusi demografi pengguna Starbucks, langkah selanjutnya yang akan dilakukan ialah mencari tahu apakah ada pola tertentu pada pembelian pelanggan. Untuk mengetahui pola tersebut, penulis melakukan pemotongan kuantil menggunakan fungsi *qcut()* untuk membagi data pengguna menjadi suatu kelompok yang memiliki karakteristik sama. Pada atribut *age_group* akan dilakukan pemotongan sebanyak lima kuantil, atribut *income_group* akan dilakukan pemotongan sebanyak lima kuantil, dan atribut *spend_group()* akan dilakukan pemotongan sebanyak sepuluh kuantil. Atribut *spend_group* diperoleh dengan melakukan manipulasi data pada *dataset transcript.json* dan *profile.json*. Lalu sebelum data ditampilkan, penulis menyortir data tersebut berdasarkan atribut *amount* untuk melihat pengguna mana yang melakukan pembelian dengan nominal terbesar.

Pada proses berikutnya, penulis akan memvisualisasikan data pendapatan pengguna berdasarkan atribut *age* dan *gender*. Kemudian penulis juga akan memvisualisasikan data jumlah total yang dibelanjakan pengguna berdasarkan atribut *age*, *gender*, dan *income*. Data yang digunakan untuk membuat visualisasi data yaitu menggunakan data pengguna dengan atribut *age_group* dan *income_group*. Lalu jenis diagram yang akan ditampilkan berupa *boxplot* (juga dikenal sebagai diagram *box-and-whisker*). *Boxplot* adalah salah satu penyajian data secara grafis dari data numerik melalui lima ukuran antara lain: nilai observasi terkecil, kuartil bawah (Q1), median (Me = Q2), kuartil atas (Q3), dan nilai observasi terbesar. *Boxplot* juga dapat memberikan gambaran tingkat kesimetrisan dari distribusi data (*skewness*) dan memuat *outlier* (data dengan nilai yang berbeda jauh dari suatu himpunan data). Penyajian *bloxpot* dapat diberikan

dalam dua cara yaitu secara horizontal dan vertikal. Namun bentuk yang paling sering digunakan pada umumnya adalah *boxplot* vertikal.

Setelah membuat visualisasi data dalam bentuk *boxplot*, penulis selanjutnya akan memvisualisasikan data pengguna dengan atribut *spend_group* untuk mengetahui bagaimana pola demografi memengaruhi pembelanjaan pengguna Starbucks. Pada proses visualisasi data yang kedua ini, penulis akan membuat diagram dalam bentuk *line chart*. Diagram garis (*line chart*) adalah salah satu bentuk visualisasi data yang sering digunakan untuk memperlihatkan perubahan kondisi dari variabel tertentu. Diagram garis mampu memberikan sekilas pemahaman tentang tren, pertumbuhan, maupun penurunan data dengan ringkas. Sebelum melakukan visualisasi, penulis membuat fungsi agregat untuk setiap variabel yang akan ditampilkan dalam *line chart* yang dapat dilihat pada Tabel 5.7.

Tabel 5.7 Variabel pada fungsi agregat

No.	Variabel Awal	Proses Agregasi	Variabel Akhir
1.	<i>Amount</i>	<i>Mean</i>	<i>avg_spent</i>
2.	<i>User ID</i>	<i>Count</i>	<i>n_users</i>
3.	<i>Signup Year</i>	<i>Median</i>	<i>median_signup</i>
4.	<i>Gender</i>	$\text{mean} \times 100$	<i>pct_male, pct_female</i>
5.	<i>Age</i>	<i>Mean</i>	<i>avg_age</i>
6.	<i>Income</i>	<i>Mean</i>	<i>avg_income</i>

Setelah mengetahui bagaimana pola demografi pengguna yang dikelompokkan berdasarkan atribut tertentu, selanjutnya penulis akan mencari tahu bagaimana pola demografi pengguna dalam menyelesaikan tiap tipe penawaran. Oleh karena itu, penulis melakukan penggabungan *dataset profile.json* dengan data penawaran yang telah dieksplorasi sebelumnya. Lalu dilakukan ekstraksi data untuk memperoleh data penawaran yang valid yaitu selain penawaran yang diselesaikan tapi belum dilihat. Kemudian akan dibentuk fungsi baru, *plot_demographic_completion()* untuk melakukan visualisasi data dalam bentuk diagram batang (*bar plot*). Setelah itu, dilakukan iterasi atribut

untuk mengetahui bagaimana pola demografi pengguna dapat memengaruhi tingkat penyelesaian dari tiap tipe penawaran.

5.1.3 Proses Pembobotan FMT

Sebelum melakukan beberapa proses pada tahap pembobotan FMT (*Frequency, Monetary, Tenure*), penulis kembali melakukan konfigurasi *library* Python dan *read dataset* karena implementasi sistem berada di *file* yang berbeda. Setelah *dataset* dibaca, akan dilakukan *data profiling* untuk mengetahui struktur dan informasi detail tentang *dataset* tersebut. *Data profiling* kembali dilakukan karena *dataset* yang dibaca pada tahap ini merupakan *dataset* hasil dari beberapa proses manipulasi data yang dilakukan sebelumnya. *Library* Python yang ditambahkan pada tahap ini ada empat macam meliputi: *joblib* (untuk menyimpan model hasil *K-Means clustering*), *sklearn* (untuk mengaktifasi fungsi pada proses normalisasi data, PCA, dan *K-Means clustering*), *yellowbrick* (untuk visualisasi data pada metode *Elbow*), dan *google.colab* (untuk menyimpan *file* model hasil *K-Means clustering*).

Setelah konfigurasi *library* Python dan *read dataset* dilakukan, langkah berikutnya ialah melakukan pembobotan terhadap variabel FMT (*Frequency, Monetary, Tenure*). Untuk itu, penulis melakukan ekstraksi pada *dataset transcript.pkl* yang memiliki atribut peristiwa berupa transaksi dan menghapus kolom peristiwa. Proses tersebut dilakukan untuk memperoleh data dengan variabel *frequency of transaction* (F) dan *monetary of transaction* (M). Selanjutnya akan dilakukan penggabungan data transaksi dengan *dataset profile.pkl* untuk memperoleh data dengan variabel *tenure of user* (T). Kemudian untuk menentukan pembobotan FMT dilakukan menerapkan *lambda expression* dan fungsi agregat seperti yang dilakukan pada proses EDA. Adapun operasi matematis yang akan diterapkan dalam *lambda expression* dapat dilihat pada Tabel 5.8.

Tabel 5.8 Operasi matematis untuk pembobotan FMT

No.	Nama Variabel	Operasi Matematis	Tipe Data
1.	<i>Frequency</i>	$\text{SUM}(\text{amount} \geq 1 \text{ dollar})$	<i>integer</i>
2.	<i>Monetary</i>	$\text{SUM}(\text{amount})$	<i>float</i>
3.	<i>Tenure</i>	$(\text{reference date} - \text{signup date}) // 30$	<i>integer</i>

Keterangan:

amount = jumlah total pembelian (nominal dalam USD)

reference date = tanggal akhir yang ditentukan sebagai batas perhitungan

signup date = tanggal pendaftaran pengguna

// = operator pembagian dengan pembulatan pada hasilnya

Setelah diperoleh hasil pembobotan untuk variabel FMT, penulis selanjutnya akan melakukan visualisasi data dalam bentuk histogram. Histogram adalah salah satu jenis diagram yang sering digunakan untuk menampilkan distribusi data. Dengan mengetahui distribusi data pada variabel FMT, penulis menjadi tahu apakah perlu atau tidaknya dilakukan proses normalisasi data. Selain itu, dengan memvisualisasikan data dalam bentuk histogram, dapat diketahui kemencengan distribusi data (*skewness*). *Skewness* adalah ukuran ketidaksimetrisan kurva pada suatu distribusi data. Distribusi data yang simetris mempunyai rata-rata, median, dan modus yang sama besar. Kurva yang simetris umumnya ditandai dengan histogram yang menyerupai bentuk lonceng.

Langkah berikutnya yang dilakukan adalah melakukan pemotongan kuantil pada variabel FMT. Kuantil adalah nilai-nilai yang membagi rentang distribusi data menjadi bagian yang sama banyaknya. Pemotongan kuantil ini nantinya akan menghasilkan variabel FMT baru dengan proporsi nilai yang lebih kecil namun sejatinya memiliki informasi yang sama dengan variabel FMT lama. Untuk ukuran kuantil yang diterapkan dalam pemotongan data variabel FMT meliputi: *frequency* (enam kuantil), *monetary* (delapan kuantil), dan *tenure* (tiga kuantil). Kemudian dilakukan penggabungan nilai pada ketiga variabel hasil pemotongan kuantil dan di-*casting* ke dalam tipe data *string*. Penggabungan nilai pada ketiga variabel tersebut diperlukan untuk proses pembuatan fungsi

segment_means() yang berada setelah *K-Means clustering* dilakukan. Selanjutnya penulis melakukan penggabungan data pemotongan kuantil pada variabel FMT dengan *dataset* pengguna untuk memperoleh atribut *gender*. Lalu atribut *gender* di-casting ke tipe data *integer* yang mewakili jenis kelamin pengguna. Pengguna pria di-casting menjadi 1 dan pengguna selain pria di-casting menjadi 0.

Setelah dilakukan pemotongan kuantil dan mengetahui bahwa distribusi data tidak simetris, maka selanjutnya penulis akan melakukan normalisasi data. Normalisasi data perlu dilakukan karena rentang data pada masing-masing variabel FMT tidak seragam. Hal tersebut disebabkan karena satuan data (contoh: USD dan bulan) yang berbeda. Perbedaan rentang data ini cukup berpengaruh terhadap hasil proses manipulasi data yang akan dilakukan selanjutnya. Oleh karena itu, penulis menerapkan fungsi *MinMaxScaler()* untuk melakukan normalisasi data pada variabel demografi pengguna yang meliputi: *age*, *income*, *frequency*, *monetary*, dan *tenure*. Fungsi tersebut sama dengan salah satu jenis normalisasi data yaitu *Min-Max normalization*, yang mana memiliki rentang data antara 0 – 1.

5.1.4 Proses Principal Component Analysis

Pada tahap *Principal Component Analysis* (PCA), penulis akan melakukan reduksi dimensi terhadap data hasil proses normalisasi. Data tersebut perlu dilakukan reduksi dimensi karena memiliki jumlah dimensi yang cukup tinggi. Permasalahan ini pada ilmu data sering disebut dengan *curse of dimensionality* (kutukan dimensi). *Curse of dimensionality* merupakan permasalahan yang terjadi akibat adanya peningkatan jumlah dimensi data sehingga dapat menyebabkan turunnya performa algoritma *machine learning*. Salah satu metode yang dapat dilakukan untuk mengatasi permasalahan ini yaitu reduksi dimensi menggunakan PCA. Dari jumlah variabel awal sebanyak lima, melalui proses uji coba PCA, akan diketahui jumlah variabel optimal yang akan digunakan dalam proses utama PCA. Jumlah variabel optimal dapat diketahui dengan menghitung variansi data kumulatif pada masing-masing jumlah komponen.

Setelah mengimplementasikan metode PCA pada data hasil normalisasi, langkah berikutnya ialah memvisualisasikan hasil proses PCA. Visualisasi PCA perlu dilakukan agar dapat diketahui distribusi data dan korelasi antarkomponen dengan variabel awal. Untuk distribusi data, penulis akan memvisualisasikannya dalam bentuk histogram. Untuk korelasi antarkomponen dengan variabel awal, penulis akan memvisualisasikannya dalam bentuk *heatmap*. *Heatmap* merupakan salah satu bentuk visualisasi data yang menggunakan representasi warna dan matriks korelasi. Umumnya, semakin besar nilai matriks korelasi pada sebuah *heatmap*, semakin gelap pula warna yang akan ditampilkan.

5.1.5 Proses Clustering dan Uji Performa

Setelah dilakukan proses PCA untuk mereduksi dimensi data, selanjutnya penulis akan melakukan proses *clustering* untuk mengetahui segmen pelanggan Starbucks. Sebelum melakukan *clustering*, penulis perlu menentukan jumlah klaster yang akan dijadikan nilai k pada algoritma *K-Means*. Penulis memanfaatkan *library yellowbrick* untuk menentukan jumlah klaster yang optimal. Kemudian penulis membuat fungsi *segment_means()* untuk mengetahui jumlah dan persentase pengguna dari tiap klaster. Fungsi tersebut memanfaatkan variabel FMT hasil pemotongan kuantil yang dilakukan ketika tahap normalisasi data. Dengan mengetahui persentase pengguna dari tiap klaster, penulis menjadi tahu seberapa baik distribusi data setelah proses *clustering* dilakukan.

Selain itu, fungsi tersebut juga menampilkan rata-rata dari tiap variabel demografi pengguna dan variabel hasil proses PCA. Setelah fungsi dibuat, penulis menerapkan algoritma *K-Means* dan fungsi baru tersebut. Untuk implementasi algoritma *K-Means*, digunakan data hasil proses PCA. Sedangkan untuk implementasi fungsi *segment_means()*, digunakan data FMT dan data hasil proses *clustering*. Setelah dilakukan proses *clustering*, akan terbentuk variabel baru yakni *cluster*. Variabel *cluster* ini selanjutnya akan menjadi label untuk membantu dalam penerapan fungsi *segment_means()*. Terakhir, akan dilakukan *string mapping* dalam menetapkan segmen pelanggan Starbucks.

Setelah diketahui segmen pelanggan menggunakan algoritma *K-Means*, proses selanjutnya yang perlu dilakukan ialah mengukur performa dari proses *clustering*. Metode yang akan digunakan untuk mencari tahu performa proses *clustering* yaitu metode *Elbow*. Metode *Elbow* akan menunjukkan perbandingan nilai klaster pertama dengan nilai klaster ke- n melalui penggambaran grafik yang menyerupai siku pada tangan (*elbow*). Untuk mendapatkan perbandingannya, penulis perlu menghitung *sum of squared errors* (SSE) dari masing-masing nilai klaster. Karena semakin besar nilai klaster k , maka nilai SSE akan semakin kecil.

SSE juga memiliki beberapa jenis perhitungan *error* yang dapat digunakan. Jenis SSE yang akan digunakan oleh penulis adalah *Within-Cluster Sum of Squares* (WCSS). WCSS adalah ukuran kedekatan antarobyek dalam suatu klaster. Nilai WCSS yang kecil mengindikasikan kemiripan antarobyek dalam suatu klaster memang valid. Selanjutnya akan dilakukan visualisasi data dalam bentuk *lineplot* agar perbandingan nilai WCSS antarklaster lebih jelas. Selain menampilkan visualisasi dalam bentuk grafik, penulis juga menampilkan iterasi nilai WCSS secara langsung agar dapat diketahui nilai klaster mana yang memiliki penurunan nilai WCSS paling signifikan.

5.1.6 Proses Visualisasi Data

Langkah berikutnya yang akan dilakukan yaitu memvisualisasikan data hasil *K-Means clustering* ke dalam berbagai jenis diagram. Visualisasi data ini dapat membantu pembaca menjadi lebih memahami makna di balik data. Selain itu dengan memvisualisasikan data, diharapkan muncul wawasan (*insight*) bagi *stakeholder* terkait sehingga mampu memberikan keputusan bisnis yang baik terhadap perusahaan. Jenis diagram yang pertama kali akan dibuat ialah *heatmap*. Melalui *heatmap*, penulis ingin mencari tahu korelasi variabel demografi pengguna pada masing-masing klaster. Sehingga penulis menjadi memiliki ide untuk memberikan nama segmen yang tepat untuk setiap klaster pelanggan. Kemudian penulis juga merancang fungsi *snake_plot()* untuk memvisualisasikan data dalam bentuk *snake plot*. *Snake plot* adalah salah satu jenis diagram yang sering digunakan untuk menggambarkan persepsi pelanggan terhadap suatu

produk. Pada penelitian ini, *snake plot* digunakan untuk melihat pengaruh suatu variabel terhadap klaster tertentu.

Setelah membuat fungsi *snake_plot()*, penulis akan menerapkan fungsi tersebut sehingga grafik *snake plot* bisa ditampilkan. Lalu akan dilakukan visualisasi data dalam bentuk diagram pencar (*scatter plot*) 2 dimensi dan 3 dimensi. Pada dasarnya, diagram pencar merupakan suatu alat interpretasi data yang digunakan untuk mengetahui bagaimana hubungan antara dua variabel. Selain itu, diagram pencar juga bisa digunakan untuk menentukan apakah jenis hubungan antara dua variabel tersebut positif, negatif, atau tidak ada hubungan sama sekali. Diagram pencar positif biasanya ditandai dengan nilai-nilai besar pada variabel sumbu X berhubungan terhadap nilai-nilai besar pada variabel sumbu Y. Sedangkan diagram pencar negatif memiliki pola yang berkebalikan dengan diagram pencar positif. Sementara itu, diagram pencar yang tidak ada hubungan ditandai dengan tidak ada kecenderungan pola tertentu pada kedua variabel.

Setelah menampilkan visualisasi data dalam bentuk *scatter plot*, selanjutnya penulis akan memvisualisasikan data dalam bentuk *count plot* untuk melihat frekuensi data pada masing-masing segmen pelanggan. Dengan mengetahui persentase pelanggan pada tiap klaster, dapat diketahui juga apakah persebaran data cukup baik setelah dilakukan *K-Means clustering*. Lalu penulis juga membuat fungsi *plot_user_demographics()* untuk memvisualisasikan data demografi pengguna pada tiap klaster. Dengan mengetahui demografi pengguna pada tiap klaster, diharapkan ada wawasan (*insight*) yang berguna untuk keperluan bisnis perusahaan. Data demografi pengguna akan divisualisasikan dalam bentuk *histogram* dengan variabel meliputi: *gender*, *age*, *income*, *frequency*, *monetary*, dan *tenure*. Kemudian penulis juga melakukan filter dan *data profiling* pada masing-masing segmen pelanggan Starbucks, sehingga karakteristik pelanggan menjadi lebih mudah dipahami.

5.2 Hasil

5.2.1 Hasil Data Preprocessing

Setelah melakukan serangkaian proses implementasi sistem menggunakan bahasa pemrograman Python, selanjutnya ialah membahas bagaimana hasil dari *source code* dan analisis yang telah diterapkan pada sistem. Pada tahap *data preprocessing* terdapat beberapa proses yang dilakukan meliputi: konfigurasi *library*, *read dataset*, *data profiling*, menangani data yang kosong, ekstraksi data dalam *dictionary*, dan konversi tipe data. Pada proses konfigurasi *library*, tidak ada *output* tertentu dikarenakan proses ini hanya berupa persiapan agar *source code* dapat berjalan dengan baik. Kemudian pada proses *read dataset* dilakukan untuk membaca *file dataset* yang berformat JSON (*JavaScript Object Notation*). Adapun hasil proses *read* untuk ketiga *dataset* dapat dilihat pada Gambar 5.1, Gambar 5.2, dan Gambar 5.3.

	reward	channels	difficulty	duration	offer_type	id
0	10	[email, mobile, social]	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd
1	10	[web, email, mobile, social]	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0
2	0	[web, email, mobile]	0	4	informational	3f207df678b143eea3cee63160fa8bed
3	5	[web, email, mobile]	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9
4	5	[web, email]	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7

Gambar 5.1 Hasil proses *read dataset portfolio.json*

	gender	age	id	became_member_on	income
0	None	118	68be06ca386d4c31939f3a4f0e3dd783	20170212	NaN
1	F	55	0610b486422d4921ae7d2bf64640c50b	20170715	112000.0
2	None	118	38fe809add3b4fcf9315a9694bb96ff5	20180712	NaN
3	F	75	78afa995795e4d85b5d9ceeca43f5fef	20170509	100000.0
4	None	118	a03223e636434f42ac4c3df47e8bac43	20170804	NaN

Gambar 5.2 Hasil proses *read dataset profile.json*

	person	event	value	time
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebfe6a799e6d9'}	0
1	a03223e636434f42ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c272da2e1d7'}	0
2	e2127556f4f64592b11af22de27a7932	offer received	{'offer id': '2906b810c7d4411798c6938adc9daaa5'}	0
3	8ec6ce2a7e7949b1bf142def7d0e0586	offer received	{'offer id': 'fafdc668e3743c1bb461111dcafc2a4'}	0
4	68617ca6246f4fbc85e91a2a49552598	offer received	{'offer id': '4d5c57ea9a6940dd891ad53e9dbe8da0'}	0

Gambar 5.3 Hasil proses *read dataset transcript.json*

Setelah hasil proses *read dataset* diperoleh, selanjutnya yaitu melihat struktur dan informasi pada *dataset* melalui proses *data profiling*. *Data profiling* merupakan komponen yang penting saat persiapan data. *Data profiling* juga dapat membantu perusahaan untuk mengidentifikasi data yang berkualitas serta menjadikannya acuan untuk pemrosesan data berikutnya. Selain itu, perusahaan dapat menggunakan wawasan atau *insight* dari proses *data profiling* untuk terus meningkatkan kualitas data ke depannya. Pada penelitian ini, hasil *data profiling* pada *dataset portfolio.json* diperoleh bahwa *dataset* terdiri atas enam variabel dengan jumlah data sebanyak 10 baris. Sedangkan hasil *data profiling* pada *dataset profile.json* diperoleh bahwa *dataset* terdiri atas lima variabel dengan jumlah data sebanyak 17.000 baris. Kemudian hasil *data profiling* pada *dataset transcript.json* diperoleh bahwa *dataset* terdiri atas empat variabel dengan jumlah data sebanyak 306.534 baris. Adapun hasil proses *data profiling* untuk ketiga *dataset* secara detail dapat dilihat pada Tabel 5.9 hingga Tabel 5.11.

Tabel 5.9 Hasil proses *data profiling* untuk *dataset portfolio.json*

No.	Kolom	Data Kosong	Tipe Data
1.	<i>reward</i>	<i>non-null</i>	<i>integer</i>
2.	<i>channels</i>	<i>non-null</i>	<i>object</i>
3.	<i>difficulty</i>	<i>non-null</i>	<i>integer</i>
4.	<i>duration</i>	<i>non-null</i>	<i>integer</i>
5.	<i>offer_type</i>	<i>non-null</i>	<i>object</i>
6.	<i>id</i>	<i>non-null</i>	<i>object</i>

Tabel 5.10 Hasil proses *data profiling* untuk *dataset profile.json*

No.	Kolom	Data Kosong	Tipe Data
1.	<i>gender</i>	<i>non-null</i>	<i>object</i>
2.	<i>age</i>	<i>non-null</i>	<i>integer</i>
3.	<i>id</i>	<i>non-null</i>	<i>object</i>
4.	<i>became_member_on</i>	<i>non-null</i>	<i>integer</i>
5.	<i>income</i>	<i>non-null</i>	<i>float</i>

Tabel 5.11 Hasil proses *data profiling* untuk *dataset transcript.json*

No.	Kolom	Data Kosong	Tipe Data
1.	<i>person</i>	<i>non-null</i>	<i>object</i>
2.	<i>event</i>	<i>non-null</i>	<i>object</i>
3.	<i>value</i>	<i>non-null</i>	<i>object</i>
4.	<i>time</i>	<i>non-null</i>	<i>integer</i>

Setelah hasil proses *data profiling* diperoleh, selanjutnya penulis akan menampilkan hasil proses menangani data yang kosong. Pada penelitian ini, *dataset profile.json* telah memiliki nilai spesifik yang menandakan bahwa data tersebut kosong atau tidak diisi oleh pelanggan Starbucks. Data yang kosong telah ditandai dengan nilai “118” pada atribut *age*. Dikarenakan data telah memiliki nilai khusus sehingga proses pencarian data yang kosong menjadi lebih mudah. Penulis memanfaatkan fungsi *show_missing_value()* untuk menampilkan jumlah dan persentase data yang kosong. Lalu dilakukan pengecekan kembali menggunakan fungsi *info()* untuk memastikan bahwa kolom lain yang memiliki data kosong berkorelasi dengan kolom *age*. Kemudian dilakukan filter untuk memisahkan data dari *dataset profile.json* yang memiliki data kosong. Dan juga, dilakukan filter untuk mengetahui data dari *dataset transcript.json* yang berhubungan dengan data dari *dataset profile.json* yang memiliki data kosong. Data pada *dataset transcript.json* memiliki hubungan dengan *dataset profile.json* karena setiap peristiwa atau transaksi yang dilakukan oleh pelanggan Starbucks pasti berkaitan dengan kolom *user_id*. Setelah kedua data difilter, penulis melakukan penghapusan data pada kedua *dataset* tersebut sesuai dengan filter yang diterapkan. Adapun jumlah dan persentase data yang kosong, hasil proses pengecekan korelasi data, serta dimensi *dataset* setelah penghapusan data yang kosong dapat dilihat pada Tabel 5.12, Tabel 5.13, dan Tabel 5.14.

Tabel 5.12 Jumlah dan persentase data yang kosong di *dataset profile.json*

No.	Kolom	Jumlah Data	Persentase
1.	<i>Gender</i>	2175	12,79%
2.	<i>Age</i>	2175	12,79%
3.	<i>Id</i>	0	0%
4.	<i>became_member_on</i>	0	0%
5.	<i>Income</i>	2175	12,79%

Tabel 5.13 Hasil proses pengecekan korelasi data yang kosong

No.	Kolom	Jumlah Data Non-Null	Korelasi
1.	<i>gender</i>	0	Ya
2.	<i>age</i>	2175	Tidak
3.	<i>id</i>	2175	Tidak
4.	<i>became_member_on</i>	2175	Tidak
5.	<i>income</i>	0	Ya

Tabel 5.14 Dimensi *dataset* setelah penghapusan data yang kosong

No.	Nama Dataset	Dimensi Data Awal	Dimensi Data Akhir
1.	<i>profile.json</i>	17.000×5	14.825×5
2.	<i>transcript.json</i>	306.534×4	272.726×4

Setelah diperoleh hasil dari proses penanganan data yang kosong, selanjutnya penulis akan menampilkan hasil proses ekstraksi data. Pada penelitian ini, terdapat satu *dataset* yang memiliki variabel bertipe data *list* dan satu *dataset* yang memiliki variabel bertipe data *dictionary*. Tipe data *list* adalah sebuah tipe data yang bisa menampung banyak nilai sekaligus. Lalu tipe data *dictionary* adalah sebuah tipe data di mana setiap anggotanya tersusun atas dua atribut yaitu *key* dan *value*. Berbeda dengan tipe data *list*, *key* ini merupakan nama atribut unik yang menampung satu nilai tertentu. Sedangkan *value* merupakan suatu nilai dengan tipe data apa pun (*string*, *integer*, *float*, dan lainnya) yang terkandung pada *key*. Pada satu variabel *dictionary*, bisa disimpan banyak *key* selama nama *key* tersebut berbeda satu sama lain. Oleh karena itu, perlu dilakukan ekstraksi

data untuk mengeluarkan nilai-nilai yang ada di dalam *list* dan atribut-atribut yang ada di dalam *dictionary*. Penulis telah merancang dua fungsi meliputi *get_unique_values()* untuk mendapatkan nilai atau *key* unik pada *list* dan *dictionary*, serta *extract_from_iterable_col()* untuk mengekstraksi data pada nilai atau *key* tersebut. Adapun hasil proses dari fungsi *get_unique_values()* dapat dilihat pada Tabel 5.15. Kemudian hasil proses dari fungsi *extract_from_iterable_col()* dapat dilihat pada Gambar 5.4 dan Gambar 5.5.

Tabel 5.15 Hasil proses dari fungsi *get_unique_values()*

No.	Nama Dataset	Nama Variabel	Hasil Fungsi <i>get_unique_values()</i>
1.	<i>portfolio.json</i>	<i>channels</i>	<i>email, mobile, social, web</i>
2.	<i>transcript.json</i>	<i>value</i>	<i>amount, offer_id, reward</i>

	reward	difficulty	duration	offer_type		id	mobile	social	web
0	10	10	7	bogo	ae264e3637204a6fb9bb56bc8210ddfd		1	1	0
1	10	10	5	bogo	4d5c57ea9a6940dd891ad53e9dbe8da0		1	1	1
2	0	0	4	informational	3f207df678b143eea3cee63160fa8bed		1	0	1
3	5	5	7	bogo	9b98b8c7a33c4b65b9aebfe6a799e6d9		1	0	1
4	5	20	10	discount	0b1e1539f2cc45b7b9fa7c272da2e1d7		0	0	1

Gambar 5.4 Hasil proses dari fungsi *extract_from_iterable_col()* untuk dataset *portfolio.json*

	person	event	time	amount		offer_id	reward
0	78afa995795e4d85b5d9ceeca43f5fef	offer received	0	NaN	9b98b8c7a33c4b65b9aebfe6a799e6d9	NaN	
2	e2127556f4f64592b11af22de27a7932	offer received	0	NaN	2906b810c7d4411798c6938adc9daaa5	NaN	
5	389bc3fa690240e798340f5a15918d5c	offer received	0	NaN	f19421c1d4aa40978ebbb69ca19b0e20d	NaN	
7	2eeac8d8feae4a8cad5a6af0499a211d	offer received	0	NaN	3f207df678b143eea3cee63160fa8bed	NaN	
8	aa4862eba776480b8bb9c68455b8c2e1	offer received	0	NaN	0b1e1539f2cc45b7b9fa7c272da2e1d7	NaN	

Gambar 5.5 Hasil proses dari fungsi *extract_from_iterable_col()* untuk dataset *transcript.json*

Setelah hasil proses ekstraksi data diperoleh, langkah berikutnya yang akan dilakukan yaitu menampilkan hasil proses konversi tipe data untuk kolom ID dari *string* menjadi *integer*. Kolom ID yang ada pada dataset *portfolio.json* dan *profile.json* perlu dikonversi karena nilai pada kolom tersebut berupa UUID. UUID adalah sebuah kombinasi sebanyak 32 karakter (*string*) yang dibuat secara acak menggunakan algoritma tertentu. Sebelum melakukan konversi tipe data,

penulis melakukan sortir untuk mengurutkan *dataset portfolio.json* berdasarkan tingkat kesulitan penawaran. Hasil dari proses sortir tersebut diperoleh bahwa urutan jenis penawaran berdasarkan tingkat kesulitan adalah penawaran informasional, penawaran *discount*, dan penawaran BOGO (*Buy One Get One*). Penawaran informasional menjadi jenis penawaran paling mudah karena tidak perlu adanya minimal pembelian untuk menyelesaikan penawaran.

Lalu penulis memanfaatkan fungsi *create_map_dict()* untuk membuat *mapping* terhadap kolom ID berupa tipe data *dictionary*. Pada fungsi inilah proses konversi tipe data pada kolom ID terjadi. Setiap ID akan menjadi *key* dan setiap nilai *index* iterasi akan menjadi *value* dalam *dictionary*. Kemudian penulis juga memanfaatkan fungsi *map_ids_to_num()* untuk mengeluarkan nilai *mapping* pada *dataset* yang dipilih. Setelah kolom ID berhasil dikonversi, nantinya akan dibuat nama variabel baru agar lebih mudah dimengerti. Maka kolom ID akan dihapus karena sudah tidak akan digunakan lagi. Pada proses konversi tipe data ini, penulis akan menerapkan fungsi terhadap tiga *dataset*. Adapun hasil proses konversi tipe data *string* menjadi *integer* untuk kolom ID dari ketiga *dataset* dapat dilihat pada Gambar 5.6 hingga Gambar 5.8.

	reward	difficulty	duration	offer_type	mobile	social	web	offer_id
0	0	0	4	informational	1	0	1	1
1	0	0	3	informational	1	1	0	2
2	3	7	7	discount	1	1	1	3
3	2	10	10	discount	1	1	1	4
4	2	10	7	discount	1	0	1	5
5	5	20	10	discount	0	0	1	6
6	5	5	7	bogo	1	0	1	7
7	5	5	5	bogo	1	1	1	8
8	10	10	7	bogo	1	1	0	9
9	10	10	5	bogo	1	1	1	10

Gambar 5.6 Hasil konversi tipe data pada *dataset portfolio.json*

	gender	age	became_member_on	income	user_id
1	F	55	20170715	112000.0	1
3	F	75	20170509	100000.0	2
5	M	68	20180426	70000.0	3
8	M	65	20180209	53000.0	4
12	M	58	20171111	51000.0	5

Gambar 5.7 Hasil konversi tipe data pada *dataset profile.json*

	event	time	amount	offer_id	reward	user_id
0	offer received	0	NaN	7.0	NaN	2
2	offer received	0	NaN	5.0	NaN	3
5	offer received	0	NaN	8.0	NaN	4
7	offer received	0	NaN	1.0	NaN	5
8	offer received	0	NaN	6.0	NaN	6

Gambar 5.8 Hasil proses konversi tipe data pada *dataset transcript.json*

Setelah hasil proses konversi tipe data *string* menjadi *integer* diperoleh, diketahui bahwa ternyata ada nilai NaN (*Not a Number*) pada *dataset transcript.json*. Nilai NaN adalah istilah yang menandakan bahwa *dataset* tersebut memiliki data yang kosong. Oleh karena itu, penulis kembali melakukan pemeriksaan data yang kosong menggunakan fungsi *show_missing_values()*. Selain melakukan pemeriksaan dasar terkait jumlah dan persentase data yang kosong, penulis juga melakukan iterasi pada variabel *event* untuk mengetahui secara pasti apakah nilai kosong ini murni karena tidak ada transaksi atau disebabkan oleh hal lain. Pada variabel *event* terdapat empat jenis peristiwa yang terjadi antara lain: *offer received*, *offer viewed*, *transaction*, dan *offer completed*. Adapun hasil proses pemeriksaan data yang kosong menggunakan fungsi *show_missing_values()* dapat dilihat pada Tabel 5.16.

Tabel 5.16 Jumlah dan persentase data yang kosong di *dataset transcript.json*

No.	Kolom	Jumlah Data	Persentase
1.	<i>time</i>	0	0%
2.	<i>amount</i>	148.805	54,55%
3.	<i>offer_id</i>	123.957	45,45%
4.	<i>reward</i>	240.318	88,11%
5.	<i>user_id</i>	0	0%

Setelah dilakukan proses iterasi variabel *event* di *dataset transcript.json* untuk mengetahui alasan data yang kosong, diperoleh fakta bahwa tidak ada masalah serius terkait data yang kosong. Data yang kosong tersebut murni disebabkan karena tidak adanya peristiwa atau transaksi yang terjadi. Pada *event offer received* dan *offer viewed*, tidak ada peristiwa penawaran yang dilakukan sehingga tidak ada nilai untuk variabel *amount* dan *reward*. Pada *event transaction*, tidak ada penawaran yang berkaitan dengan transaksi sehingga tidak ada nilai untuk variabel *offer_id* dan *reward*. Pada *event offer completed*, tidak ada penyelesaian penawaran yang dilakukan sehingga tidak ada nilai untuk variabel *amount*. Dikarenakan variabel tersebut kosong karena tidak peristiwa atau transaksi yang terjadi, untuk menjaga kualitas *dataset*, penulis akan mengisi semua data yang kosong tersebut dengan nilai 0. Adapun hasil proses iterasi variabel *event* di *dataset transcript.json* dapat dilihat pada Tabel 5.17.

Tabel 5.17 Hasil proses iterasi variabel *event* di *dataset transcript.json*

No.	Nama Event	Jumlah Data	Variabel Terkait	Jumlah Data yang Kosong
1.	<i>offer received</i>	66.501	<i>amount</i>	66.501
			<i>Reward</i>	66.501
2.	<i>offer viewed</i>	49.860	<i>amount</i>	49.860
			<i>Reward</i>	49.860
3.	<i>transaction</i>	123.957	<i>offer_id</i>	123.957
			<i>Reward</i>	123.957
4.	<i>offer completed</i>	32.444	<i>amount</i>	32.444

Proses berikutnya yang dilakukan pada tahap *data preprocessing* adalah mengkonversi beberapa variabel yang masih menggunakan tipe data yang kurang sesuai. Tipe data ini merupakan hal yang cukup penting untuk diperhatikan karena dapat memengaruhi proses-proses yang akan dilakukan di tahap selanjutnya. Pada proses konversi tipe data ini, ada empat variabel yang akan dikonversi meliputi: *income*, *become_member_on*, *offer_id*, dan *reward*. Variabel *income*, *offer_id*, dan *reward* perlu dikonversi karena nilai data berupa bilangan bulat. Sedangkan variabel *become_member_on* perlu dikonversi untuk mempermudah proses pembobotan model FMT nantinya. Selain dilakukan konversi tipe data, nama variabel *become_member_on* diubah menjadi *signup_date* agar lebih ringkas. Adapun hasil proses konversi tipe data lanjutan dapat dilihat pada Tabel 5.18.

Tabel 5.18 Hasil proses konversi tipe data (lanjutan)

No.	Nama Variabel	Tipe Data Awal	Tipe Data Akhir
1.	<i>income</i>	<i>float</i>	<i>integer</i>
2.	<i>become_member_on</i>	<i>integer</i>	<i>datetime</i>
3.	<i>offer_id</i>	<i>float</i>	<i>integer</i>
4.	<i>reward</i>	<i>float</i>	<i>integer</i>

Hal terakhir yang dilakukan pada tahap ini yaitu menampilkan hasil proses pencarian data yang terduplikasi. Data duplikat atau data ganda merupakan suatu kondisi di mana data entitas yang seharusnya sama, tapi tercatat lebih dari satu kali. Data duplikat bisa disebabkan oleh banyak hal, mulai dari rancangan sistem yang tidak memiliki validasi hingga kesalahan manusia. Data duplikat dapat menimbulkan kerugian yang sangat besar bagi sebuah perusahaan jika tidak diatasi dengan baik. Pada beberapa kasus, data duplikat menyebabkan kekacauan penanganan stok, analisis data yang tidak akurat, dan gagalnya rekonsiliasi data finansial. Pada penelitian ini, penulis menemukan data duplikat pada *dataset transcript.json*. *Dataset* tersebut cukup rentan terjadi redundansi data karena memuat banyak peristiwa dan transaksi yang dilakukan oleh pelanggan. Setelah dilakukan pencarian pada *dataset transcript.json*, ada duplikasi sebanyak 374 data. Sehingga total baris setelah terjadi duplikasi ada sebanyak 747 baris. Lalu

ditemukan juga bahwa jenis *event* yang duplikat adalah *offer completed*. Menurut pendapat penulis, tidak mungkin bahwa pelanggan yang sama menyelesaikan jenis penawaran yang sama dalam waktu yang sama. Maka dari itu, data duplikat tersebut selanjutnya akan dihapus dari *dataset transcript.json*. Adapun hasil proses pencarian data yang duplikat dapat dilihat pada Gambar 5.9.

	event	time	amount	offer_id	reward	user_id
66122	offer completed	168	0.0	5	2	1271
66123	offer completed	168	0.0	5	2	1271
66782	offer completed	168	0.0	9	10	3872
66783	offer completed	168	0.0	9	10	3872
67613	offer completed	168	0.0	7	5	7462
...
304756	offer completed	708	0.0	6	5	6340
305550	offer completed	714	0.0	4	2	2041
305551	offer completed	714	0.0	4	2	2041
306455	offer completed	714	0.0	6	5	13967
306456	offer completed	714	0.0	6	5	13967

Gambar 5.9 Hasil proses pencarian data duplikat di *dataset transcript.json*

5.2.2 Hasil Exploratory Data Analysis

Pada tahap *exploratory data analysis* (EDA) terdapat dua proses utama yang dilakukan yaitu eksplorasi pada data penawaran dan data pengguna. Eksplorasi pada data penawaran terbagi menjadi empat proses antara lain: mencari tahu berapa banyak penawaran dari tiap tipe yang telah dikirimkan, mencari tahu berapa banyak penawaran *reward* yang telah diselesaikan, mencari tahu berapa banyak penawaran informasional yang dilanjutkan dengan transaksi, dan mencari tahu tipe penawaran mana yang memiliki tingkat penyelesaian paling tinggi. Eksplorasi pada data pengguna terbagi menjadi tiga proses antara lain: mencari tahu bagaimana distribusi pada demografi pengguna, mencari tahu apakah ada pola tertentu pada *user spending*, dan mencari tahu apakah ada pola demografi tertentu pada *offer completion*.

Setelah melalui serangkaian proses pada tahap *data preprocessing* sehingga data menjadi siap untuk dieksplorasi. Hasil proses eksplorasi pertama yang akan ditampilkan adalah informasi mengenai berapa banyak penawaran dari tiap tipe yang telah dikirimkan kepada pelanggan Starbucks. Penulis memanfaatkan fungsi *filter_for_events()* untuk memfilter penawaran berdasarkan peristiwa *offer received* untuk memperoleh data penawaran yang telah dikirimkan. Lalu penulis menggunakan fungsi *shape* untuk menampilkan jumlah total penawaran yang telah dikirimkan. Hasil fungsi tersebut diperoleh bahwa ada sebanyak 66.501 penawaran yang telah dikirimkan kepada pelanggan Starbucks. Kemudian penulis juga ingin mengetahui jumlah penawaran yang telah dikirimkan berdasarkan tipe penawaran dan ID penawaran. Dengan memanfaatkan fungsi *value_counts()*, penulis memperoleh data tersebut seperti yang dapat dilihat pada Tabel 5.19 dan Tabel 5.20.

Tabel 5.19 Jumlah dan persentase penawaran berdasarkan tipe

No.	Tipe Penawaran	Jumlah Penawaran	Persentase
1.	<i>Discount</i>	26.664	40,09%
2.	<i>BOGO</i>	26.537	39,90%
3.	<i>Informational</i>	13.300	19,99%

Tabel 5.20 Jumlah penawaran berdasarkan ID penawaran

No.	ID Penawaran	Tipe Penawaran	Jumlah Penawaran
1.	6	<i>Discount</i>	6.726
2.	7	<i>BOGO</i>	6.685
3.	9	<i>BOGO</i>	6.683
4.	1	<i>Informational</i>	6.657
5.	3	<i>Discount</i>	6.655
6.	4	<i>Discount</i>	6.652
7.	2	<i>Informational</i>	6.643
8.	5	<i>Discount</i>	6.631
9.	10	<i>BOGO</i>	6.593
10.	8	<i>BOGO</i>	6.576

Hasil proses eksplorasi kedua yang akan ditampilkan adalah informasi mengenai berapa banyak penawaran *reward* yang telah diselesaikan oleh pelanggan. Penawaran *reward* merupakan istilah yang digunakan untuk menunjukkan tipe penawaran *discount* dan BOGO (*Buy One Get One*). Kedua tipe penawaran tersebut disebut penawaran *reward* karena jika seorang pelanggan menyelesaikan penawaran, maka pelanggan akan memperoleh *reward* berupa diskon. Untuk memperoleh data penawaran *reward* yang telah diselesaikan, penulis nantinya akan membuat dua atribut *viewed* dan *completed* untuk menunjukkan apakah sebuah penawaran telah dilihat atau diselesaikan.

Langkah awal yang dilakukan penulis untuk memperoleh data penawaran *reward* yang telah diselesaikan ialah melakukan filter terhadap data penawaran. Data penawaran yang digunakan akan terbatas pada dua tipe penawaran yaitu penawaran *discount* dan *BOGO*. Setelah data diperoleh, penulis memanfaatkan fungsi *concat_str_cols()* untuk menggabungkan nilai ID penawaran dengan ID pengguna. Kemudian penulis juga menerapkan *lambda expression* untuk memperoleh nilai *expire time* dari sebuah penawaran. Nilai *expire time* ini adalah waktu yang dibutuhkan oleh sebuah penawaran untuk mencapai kedaluwarsa. Namun nilai *expire time* ini hanya berlaku untuk penawaran dengan jenis peristiwa *offer received*. Perhitungan nilai *expire time* pada data penawaran *reward* ini menggunakan rumus yang berada di Tabel 5.2. Sebab jenis peristiwa tersebut menandakan bahwa sebuah penawaran itu belum dilihat dan belum diselesaikan. Adapun hasil proses penggabungan *string* dan perhitungan nilai *expire time* untuk data penawaran *reward* dapat dilihat pada Gambar 5.10.

	index	event	time	offer_id	user_id	amount	reward	difficulty	duration	offer_type	mobile	social	web	expire_time	oid_uid
0	91	offer received	0	10	100	0.0	10	10	5	bogo	1	1	1	120	10_100
1	15575	offer viewed	6	10	100	0.0	10	10	5	bogo	1	1	1	-1	10_100
2	151484	offer received	408	10	1000	0.0	10	10	5	bogo	1	1	1	528	10_1000
3	182714	offer viewed	444	10	1000	0.0	10	10	5	bogo	1	1	1	-1	10_1000
4	185051	offer completed	450	10	1000	0.0	10	10	5	bogo	1	1	1	-1	10_1000

Gambar 5.10 Hasil penggabungan *string* dan perhitungan nilai *expire time* untuk data penawaran *reward*

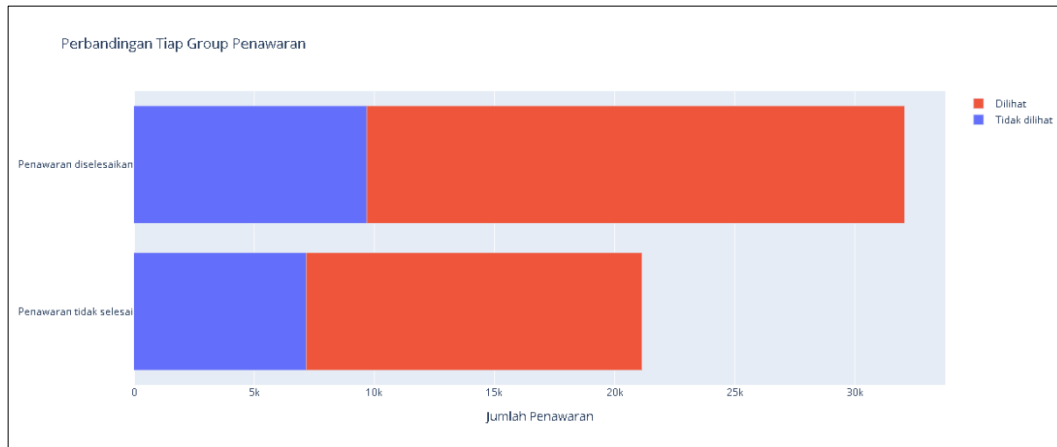
Setelah memperoleh data penawaran *reward*, selanjutnya penulis akan melakukan iterasi untuk memberikan nilai pada atribut *viewed* dan *completed*.

Sebelum melalui proses iterasi, akan diinisialisasi nilai awal -1 pada atribut *viewed* dan *completed*. Proses iterasi ini akan memanfaatkan fungsi *fill_with()* untuk mengisi nilai dan *go_to_event()* untuk melompat menuju ke peristiwa tertentu. Setelah melalui proses iterasi, nantinya nilai pada atribut *viewed* dan *completed* akan berisi nilai biner antara 0 atau 1. Nilai 0 berarti belum dilihat atau belum diselesaikan, sedangkan nilai 1 berarti sudah dilihat atau sudah diselesaikan.

Kemudian penulis akan mengelompokkan setiap penawaran berdasarkan karakteristiknya menggunakan fungsi *group_offer()*. Untuk penjelasan detail dari tiap grup penawaran telah dijelaskan pada Tabel 5.3. Lalu akan dihitung jumlah penawaran *reward* yang telah dikirimkan, dilihat, maupun diselesaikan oleh pelanggan. Data jumlah penawaran tersebut juga akan divisualisasikan dalam bentuk *horizontal bar plot* untuk melihat perbandingan jumlah pada tiap grup penawaran. Hasilnya, dari 21.131 penawaran *reward* yang tidak diselesaikan, 7.165 (sekitar sepertiga) bahkan tidak dilihat. Dari 32.070 penawaran *reward* selesai yang tercatat, hanya 22.382 penawaran *reward* yang benar-benar selesai (penawaran dilihat terlebih dahulu sebelum diselesaikan). Sisanya 9.688 penawaran *reward* tidak dapat benar-benar dianggap selesai karena penawaran tersebut diselesaikan tanpa dilihat oleh pengguna atau pengguna melihat penawaran setelah durasi penawaran selesai. Adapun jumlah penawaran *reward* dan hasil visualisasi data dalam bentuk *horizontal bar plot* dapat dilihat pada Tabel 5.21 dan Gambar 5.11.

Tabel 5.21 Jumlah dan persentase penawaran *reward* pada tiap grup

No.	Nama Grup	Jumlah Data	Persentase
1.	<i>Group 1</i>	7.165	33,90%
2.	<i>Group 2</i>	13.966	66,09%
3.	<i>Group 3</i>	9.688	30,20%
4.	<i>Group 4</i>	22.382	69,79%
Total penawaran <i>reward</i> yang telah dikirimkan			53.201
Total penawaran <i>reward</i> yang telah diselesaikan			32.070
Total penawaran <i>reward</i> yang tidak diselesaikan			21.131



Gambar 5.11 Hasil visualisasi data (*bar plot*) untuk jumlah penawaran *reward*

Setelah memperoleh hasil eksplorasi pada data penawaran *reward*, selanjutnya penulis akan mencari tahu hasil eksplorasi pada data penawaran informasional. Karena penawaran informasional tidak memberikan *reward* kepada pelanggan dan atribut penyelesaian penawaran, maka data penawaran informasional akan dieksplorasi dengan metode yang sedikit berbeda dengan data penawaran *reward*. Pada eksplorasi ini, penulis ingin memperoleh data penawaran informasional yang diikuti dengan transaksi pembelian produk. Namun transaksi tersebut perlu dilakukan dalam rentang waktu tertentu (durasi penawaran). Sehingga jika transaksi tersebut dilakukan di luar rentang waktu yang ada, maka data tersebut tidak akan dihitung.

Langkah pertama yang dilakukan ialah memfilter data untuk memperoleh data penawaran informasional melalui penggabungan *dataset portfolio.json* dengan *transcript.json*. Kemudian penulis juga memfilter data penawaran informasional yang tidak memiliki peristiwa berupa *transaction*. Hasil dari proses filter data tersebut diperoleh data sebanyak 98.029 baris. Lalu akan diterapkan kembali *lambda expression* untuk menghitung nilai *expire time* dari penawaran informasional. Perhitungan nilai *expire time* ini menggunakan rumus yang sedikit berbeda dengan yang telah dilakukan sebelumnya. Pada data penawaran informasional, terdapat kondisi tertentu yang menjadi syarat agar data tersebut valid untuk dihitung. Sehingga perhitungan nilai *expire time* untuk data penawaran informasional akan menggunakan rumus yang berada di Tabel 5.4.

Selain itu, juga dilakukan penggabungan *string* pada variabel ID penawaran dan ID pengguna. Adapun hasil perhitungan nilai *expire time* dan penggabungan *string* untuk data penawaran informasional dapat dilihat pada Gambar 5.12.

	index	event	time	offer_id	user_id	amount	reward	difficulty	duration	offer_type	mobile	social	web	expire_time	oid_uid
0	20282	transaction	18	0	1	21.51	-1	-1	-1	-1	-1	-1	-1	-1	0_1
1	49501	transaction	144	0	1	32.28	-1	-1	-1	-1	-1	-1	-1	-1	0_1
2	201571	offer received	504	1	1	0.00	0	0	4	Informational	1	0	1	600	1_1
3	227842	transaction	528	0	1	23.22	-1	-1	-1	-1	-1	-1	-1	-1	0_1
4	47582	transaction	132	0	2	19.89	-1	-1	-1	-1	-1	-1	-1	-1	0_2

Gambar 5.12 Hasil penggabungan *string* dan perhitungan nilai *expire time* untuk data penawaran informasional

Setelah memperoleh data penawaran informasional, selanjutnya penulis akan melakukan iterasi untuk memberikan nilai pada atribut *viewed* dan *completed*. Sebelum melalui proses iterasi, akan diinisialisasi nilai awal -1 pada atribut *viewed* dan *completed*. Proses iterasi ini akan memanfaatkan fungsi *fill_with()* untuk mengisi nilai dan *go_to_event()* untuk melompat menuju ke peristiwa tertentu. Setelah melalui proses iterasi, nantinya nilai pada atribut *viewed* dan *completed* akan berisi nilai biner antara 0 atau 1. Nilai 0 berarti belum dilihat atau belum diselesaikan, sedangkan nilai 1 berarti sudah dilihat atau sudah diselesaikan. Kemudian penulis akan mengelompokkan setiap penawaran berdasarkan karakteristiknya menggunakan fungsi *group_offer()*. Untuk penjelasan detail dari tiap grup penawaran telah dijelaskan pada Tabel 5.5. Lalu akan dihitung jumlah penawaran informasional yang telah dikirimkan, dilihat, maupun diselesaikan oleh pelanggan. Data jumlah penawaran tersebut juga akan divisualisasikan dalam bentuk *horizontal bar plot* untuk melihat perbandingan pada tiap grup penawaran.

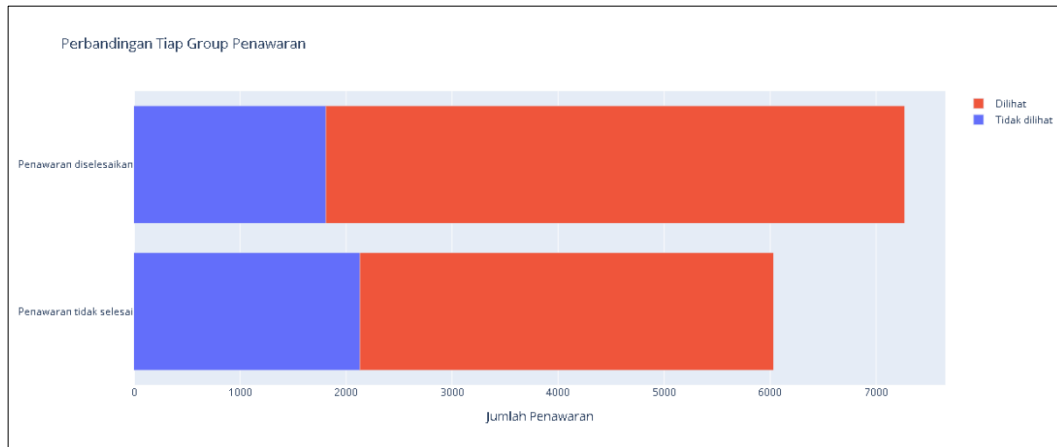
Untuk penawaran informasional, atribut *completed* berarti pengguna melakukan transaksi setidaknya 2,5 dolar selama durasi penawaran. Jumlah dolar minimum adalah sebuah asumsi bahwa penawaran informasional adalah kampanye periklanan dengan biaya setidaknya 2,5 dolar. Penulis memutuskan pada angka 2,5 dolar karena ini adalah setengah dari jumlah kesulitan terendah pada semua tipe penawaran. Minimum pembelian ini mengabaikan transaksi kecil yang kemungkinan besar tidak terkait dengan penawaran. Penulis juga membuat

asumsi bahwa transaksi di atas ambang batas valid untuk dikaitkan dengan penawaran. Selain itu, penulis juga tidak memiliki cara untuk mengetahui dengan pasti karena tidak ada peristiwa *offer completed* untuk mengonfirmasinya. Jika penawaran dilihat, durasi akan dimulai setelah dilihat karena saat itulah pengguna mengetahui penawaran tersebut. Jika penawaran tidak dilihat, durasinya dimulai setelah diterima. Dalam kasus ini, pengguna tidak mengetahui penawaran tersebut dan tetap membelanjakan uang mereka.

Hasil dari proses iterasi dan penerapan fungsi *show_group_count()* diperoleh bahwa dari 6.031 penawaran informasional yang tidak diselesaikan, 2.131 penawaran bahkan tidak dilihat. Dari 7.269 penawaran *reward* selesai yang tercatat, hanya 5.460 penawaran informasional yang benar-benar selesai (penawaran dilihat terlebih dahulu sebelum diselesaikan). Sisanya 4.709 penawaran informasional tidak dapat benar-benar dianggap selesai karena penawaran tersebut diselesaikan tanpa dilihat oleh pengguna atau pengguna melihat penawaran setelah durasi penawaran selesai. Adapun jumlah dan persentase penawaran informasional di tiap grup serta hasil visualisasi data dalam bentuk *horizontal bar plot* dapat dilihat pada Tabel 5.22 dan Gambar 5.13.

Tabel 5.22 Jumlah dan persentase penawaran informasional pada tiap grup

No.	Nama Grup	Jumlah Data	Persentase
1.	<i>Group 1</i>	2.131	35,33%
2.	<i>Group 2</i>	3.900	64,66%
3.	<i>Group 3</i>	1.809	24,88%
4.	<i>Group 4</i>	5.460	75,11%
Total penawaran informasional yang telah dikirimkan			13.300
Total penawaran informasional yang telah diselesaikan			7.269
Total penawaran informasional yang tidak diselesaikan			6.031



Gambar 5.13 Hasil visualisasi data untuk jumlah penawaran informasional

Eksplorasi terakhir pada data penawaran adalah mencari tahu tipe penawaran mana yang memiliki tingkat penyelesaian paling tinggi. Langkah pertama yang perlu dilakukan yaitu melakukan penggabungan data penawaran *reward* dan informasional hasil filter pada proses sebelumnya. Kemudian akan dilakukan filter data lagi untuk memisahkan data penawaran yang valid (kecuali *group 3*) dan data penawaran pada *group 4*. Lalu akan dilakukan penggabungan terhadap dua *dataset* tersebut untuk memperoleh data tingkat penyelesaian penawaran. Variabel *pct_completed* akan dibentuk sebagai persentase dari jumlah penawaran yang telah diselesaikan dengan jumlah total penawaran. Selanjutnya akan dilakukan penggabungan data tingkat penyelesaian penawaran dengan *dataset portfolio.json* untuk memperoleh metadata penawaran. Pada langkah berikutnya, dilakukan penggabungan *string* menggunakan fungsi *concat_str_cols()* untuk membentuk variabel *offer* (berisi metadata penawaran) dan *summary* (berisi informasi terkait tingkat penyelesaian penawaran). Dan akhirnya, penulis memvisualisasikan data tingkat penyelesaian penawaran ke dalam bentuk *bar plot*. Adapun hasil proses eksplorasi dan visualisasi data penawaran yang memiliki tingkat penyelesaian paling tinggi dapat dilihat pada Tabel 5.23 dan Gambar 5.14.

Tabel 5.23 Tingkat penyelesaian penawaran pada tiap tipe penawaran

No.	ID Penawaran	Jumlah Penyelesaian	Jumlah Penawaran	Persentase Penyelesaian
1.	4	4.316	6.011	71,80%
2.	3	4.101	5.909	69,40%
3.	8	3.354	5.856	57,27%
4.	2	3.077	6.130	50,20%
5.	9	2.554	5.633	45,34%
6.	10	2.719	6.031	45,08%
7.	1	2.383	5.361	44,45%
8.	7	2.020	4.564	44,26%
9.	5	2.024	4.795	42,21%
10.	6	1.294	4.714	27,45%

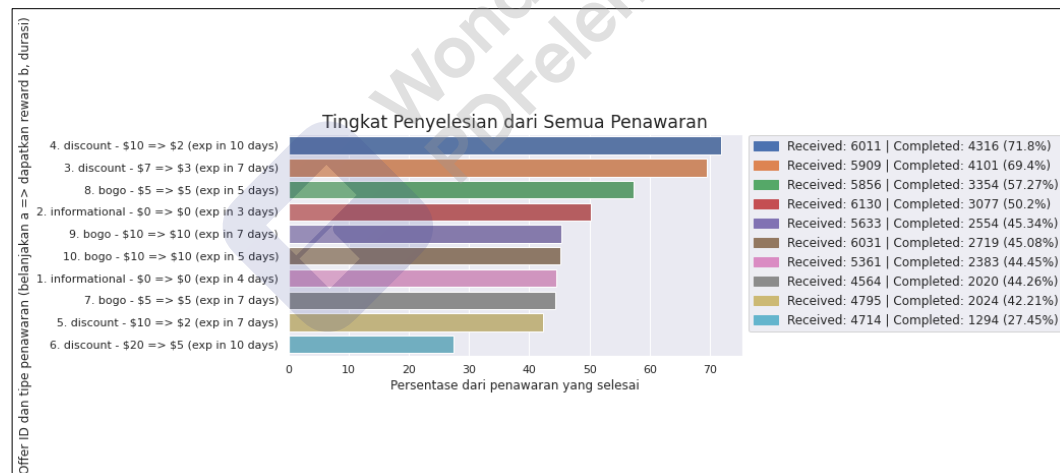
**Gambar 5.14** Hasil visualisasi untuk data tingkat penyelesaian penawaran

Diagram batang (*bar plot*) di atas menunjukkan 10 tipe penawaran yang berbeda dalam *dataset* meliputi: empat penawaran BOGO, empat penawaran *discount*, dan dua penawaran informasional. Variabel *completed* untuk penawaran informasional yang ditunjukkan di atas adalah di mana pengguna melihat penawaran dan melakukan transaksi minimal 2,50 dolar dalam durasi tertentu. *Offer 4 (discount)* dengan rincian belanjaan 10 dolar untuk mendapatkan diskon 2 dolar yang berakhir dalam 10 hari, memiliki tingkat penyelesaian tertinggi

sebesar 71,80% (4.316 dari 6.011 penawaran). *Offer 6* dengan rincian belanjaan 20 dolar untuk mendapatkan diskon 5 dolar yang berakhir dalam 10 hari, memiliki tingkat penyelesaian terendah sebesar 27,45% (1.294 dari 4.714 penawaran).

Penulis menyimpulkan bahwa variabel *reward* bukan merupakan faktor utama dalam menentukan tingkat penyelesaian penawaran. Awalnya, penulis berasumsi bahwa penawaran BOGO akan memiliki tingkat penyelesaian tertinggi karena pelanggan mendapatkan *reward* sebesar 100% atas uang yang mereka keluarkan. Tapi ternyata data berbicara hal yang sebaliknya. Ini kemungkinan besar karena penawaran BOGO secara inheren lebih sulit untuk diselesaikan. Di sisi lain, penawaran *discount* memungkinkan pelanggan untuk mengakumulasi jumlah pembelian dengan beberapa transaksi. Misalnya, *Offer 4* (belanjaan 10 dolar untuk mendapatkan diskon 2 dolar) akan selesai jika pelanggan melakukan empat transaksi kecil masing-masing 3 dolar, karena transaksi ini akan terakumulasi menjadi lebih dari 10 dolar selama durasi penawaran.

Setelah memperoleh beberapa *insight* dari eksplorasi data penawaran, selanjutnya penulis akan menggali beberapa *insight* dari data pengguna. Pertama, penulis ingin mencari tahu bagaimana distribusi pada demografi pengguna. Dengan mengetahui distribusi pada demografi pengguna, dapat diperoleh gambaran umum atau karakteristik dari pelanggan Starbucks. Langkah pertama yang dilakukan adalah menambahkan variabel *signup year* pada *dataset profile.json*. Untuk menambahkannya, penulis hanya perlu mengambil nilai *year* pada variabel *signup date*. Lalu dilakukan *data profiling* singkat untuk mengetahui informasi statistik pada *dataset profile.json*. Kemudian penulis membuat fungsi *plot_user_demographics()* yang nantinya akan digunakan untuk memvisualisasikan data demografi pengguna dalam bentuk histogram.

Untuk eskplorasi data demografi pengguna pada *dataset profile.json*, penulis menggunakan variabel demografi pengguna meliputi: *gender*, *age*, *signup year*, dan *income*. Dari hasil visualisasi data diperoleh bahwa untuk variabel *gender*, ada lebih banyak pengguna berjenis kelamin pria daripada wanita, dan sebagian kecil pengguna yang tidak diketahui (*other*). Untuk variabel *age*, usia pengguna berkisar antara 18 hingga 100 tahun, dengan distribusi tertinggi antara

usia 50 dan 65 tahun. Untuk variabel *signup year*, histogram menunjukkan ada peningkatan jumlah pendaftaran pengguna dari 2013 ke 2017, lalu agak turun di 2018. Untuk variabel *income*, pengguna memiliki pendapatan berkisar dari USD 30k per tahun hingga USD 120k per tahun, dengan distribusi tertinggi antara USD 50k per tahun hingga USD 75k per tahun. Adapun hasil visualisasi dari distribusi demografi pengguna dapat dilihat pada Gambar 5.15.



Gambar 5.15 Histogram untuk distribusi demografi pengguna

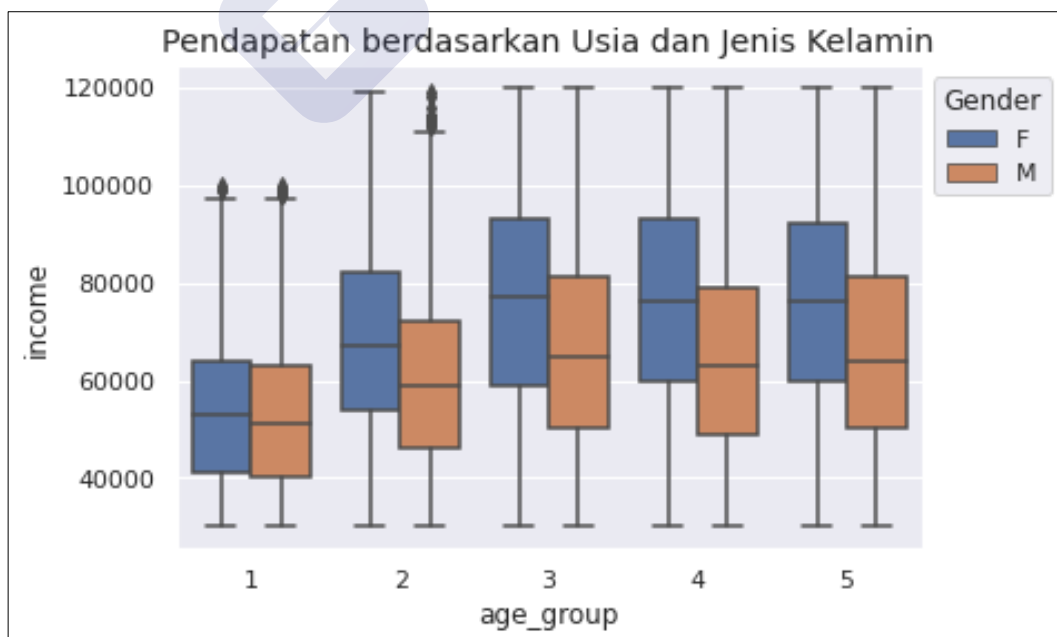
Setelah memperoleh gambaran umum tentang distribusi demografi pengguna, penulis selanjutnya ingin mencari tahu apakah ada pola tertentu dalam aktivitas pembelian yang dilakukan oleh pengguna Starbucks. Sebelum melakukan eksplorasi data, penulis melakukan pemotongan kuantil terhadap *dataset profile.json* untuk membagi pengguna ke beberapa kelompok tertentu. Pemotongan kuantil ini didasarkan pada variabel demografi tertentu antara lain: *age*, *income*, dan *amount*. Untuk variabel *amount*, data diperoleh dengan mengekstraksi dan menjumlahkan variabel *amount* pada *dataset transcript.json*. Kemudian dilakukan penggabungan data hasil ekstraksi dengan *dataset profile.json*. Pemotongan kuantil yang dilakukan akan membagi variabel *age* menjadi lima *age_group*, variabel *income* menjadi lima *income_group*, dan variabel *amount* menjadi sepuluh *spend_group*. Adapun hasil pemotongan kuantil

terhadap variabel *age*, *income*, dan *amount* di *dataset profile.json* dapat dilihat pada Gambar 5.16.

	gender	age	income	user_id	signup_date	signup_year	age_group	income_group	amount	spend_group
12951	M	77	97000	12952	2017-09-26	2017	5	5	1608.69	10
11767	M	32	64000	11768	2018-06-19	2018	1	3	1365.66	10
9404	F	36	71000	9405	2017-05-21	2017	1	4	1327.74	10
9293	M	53	103000	9294	2015-08-25	2015	3	5	1321.42	10
8987	M	50	86000	8988	2016-05-12	2016	2	5	1319.97	10

Gambar 5.16 Hasil pemotongan kuantil pada *dataset profile.json*

Langkah berikutnya yang akan dilakukan penulis adalah memvisualisasikan data hasil pemotongan kuantil ke dalam bentuk *box plot*. *Box plot* adalah salah satu bentuk diagram yang digunakan untuk memahami karakteristik dalam distribusi data. Untuk *box plot* pertama, penulis membandingkan *gender* dan *age group* terhadap variabel *income*. Dengan begitu, penulis dapat mengetahui pendapatan rata-rata dari masing-masing *gender* dan *age group*. Sebagai contoh, pada *age group 1*, pengguna berjenis kelamin pria memiliki pendapatan rata-rata sekitar USD 40k sampai USD 63k. Adapun visualisasi data berbentuk *box plot* dan interpretasi lebih lanjut terkait *boxplot* dapat dilihat pada Gambar 5.17 dan Tabel 5.24.

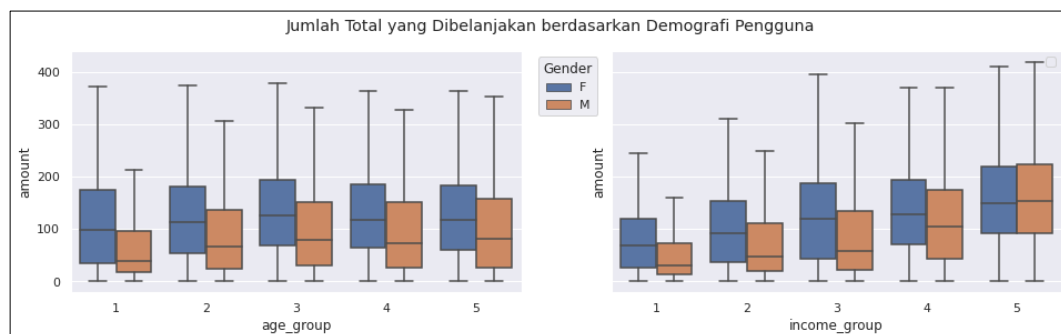


Gambar 5.17 Hasil visualisasi data untuk pendapatan pelanggan berdasarkan kelompok usia dan jenis kelamin

Tabel 5.24 Interpretasi untuk hasil visualisasi data pada Gambar 5.64

No.	Nama Grup	Jenis Kelamin	Rentang Usia	Rata-rata <i>Income</i>
1.	<i>Age Group 1</i>	Pria	18 – 39	40.000 – 63.000
		Wanita		41.000 – 64.000
2.	<i>Age Group 2</i>	Pria	40 – 51	46.000 – 72.000
		Wanita		54.000 – 82.000
3.	<i>Age Group 3</i>	Pria	52 – 59	50.000 – 81.00
		Wanita		59.000 – 93.000
4.	<i>Age Group 4</i>	Pria	60 – 69	49.000 – 79.000
		Wanita		60.000 – 93.000
5.	<i>Age Group 5</i>	Pria	70 – 101	50.000 – 81.000
		Wanita		60.000 – 92.000

Selain membandingkan *gender* dan *age group* terhadap variabel *income*, penulis juga membandingkan *gender* dan *age group* terhadap variabel *amount*. Dengan begitu, penulis dapat mengetahui total pembelian dari masing-masing *gender* dan *age group*. Sebagai contoh, pada *age group 1*, pengguna berjenis kelamin pria memiliki total pembelian sekitar USD 17,79 sampai USD 95,80. Lalu penulis juga membandingkan *gender* dan *income group* terhadap variabel *amount*. Dengan begitu, penulis dapat mengetahui total pembelian dari masing-masing *gender* dan *income group*. Sebagai contoh, pada *age group 1*, pengguna berjenis kelamin pria memiliki total pembelian sekitar USD 14,45 sampai USD 72,65. Adapun visualisasi data berbentuk *box plot* dan interpretasi lebih lanjut terkait *boxplot* dapat dilihat pada Gambar 5.18, Tabel 5.25, dan Tabel 5.26.

**Gambar 5.18** Jumlah total pembelian berdasarkan demografi pengguna

Tabel 5.25 Interpretasi untuk hasil visualisasi data pada Gambar 5.65

No.	Nama Grup	Jenis Kelamin	Rentang Usia	Total Pembelian
1.	<i>Age Group 1</i>	Pria	18 – 39	17,79 – 95,80
		Wanita		34,22 – 175,18
2.	<i>Age Group 2</i>	Pria	40 – 51	23,39 – 136,95
		Wanita		52,85 – 182,07
3.	<i>Age Group 3</i>	Pria	52 – 59	30,46 – 151,45
		Wanita		68,03 – 193,05
4.	<i>Age Group 4</i>	Pria	60 – 69	25,39 – 150,92
		Wanita		65,24 – 185,70
5.	<i>Age Group 5</i>	Pria	70 – 101	27,19 – 157,63
		Wanita		60,92 – 183,18

Tabel 5.26 Interpretasi untuk hasil visualisasi data pada Gambar 5.65 (lanjutan)

No.	Nama Grup	Jenis Kelamin	Rentang Gaji	Total Pembelian
1.	<i>Income Group 1</i>	Pria	30k – 45k	14,45 – 72,65
		Wanita		27,01 – 119,06
2.	<i>Income Group 2</i>	Pria	46k – 57k	19,08 – 111,24
		Wanita		37,74 – 154,07
3.	<i>Income Group 3</i>	Pria	58k – 70k	23,10 – 135,00
		Wanita		43,96 – 187,47
4.	<i>Income Group 4</i>	Pria	71k – 85k	43,83 – 175,65
		Wanita		71,82 – 194,87
5.	<i>Income Group 5</i>	Pria	86k – 120k	91,62 – 224,24
		Wanita		91,27 – 220,32

Setelah menampilkan visualisasi data tentang hasil pemotongan kuantil *age group* dan *income group*, selanjutnya akan dilakukan visualisasi data untuk hasil pemotongan kuantil *spend group*. Sebelum itu, penulis ingin menggali informasi statistik dari tiap *spend group*. Penulis merancang fungsi agregat untuk memperoleh informasi tersebut dari variabel *amount*, *user_id*, *signup_year*, *gender*, *age*, dan *income*. Fungsi statistik yang diterapkan pada masing-masing variabel telah dijelaskan pada Tabel 5.7. Lalu dari keenam variabel tersebut nantinya akan divisualisasikan dalam bentuk *line plot* untuk mengetahui pola tertentu pada aktivitas pembelian pelanggan. Pola pembelian tersebut tentunya juga didasarkan pada *spend group* yang menjadi informasi tambahan dalam menilai aktivitas pembelian pelanggan. Adapun hasil dari fungsi agregat dan visualisasi data dalam bentuk *line plot* dapat dilihat pada Tabel 5.27 dan Gambar 5.19.

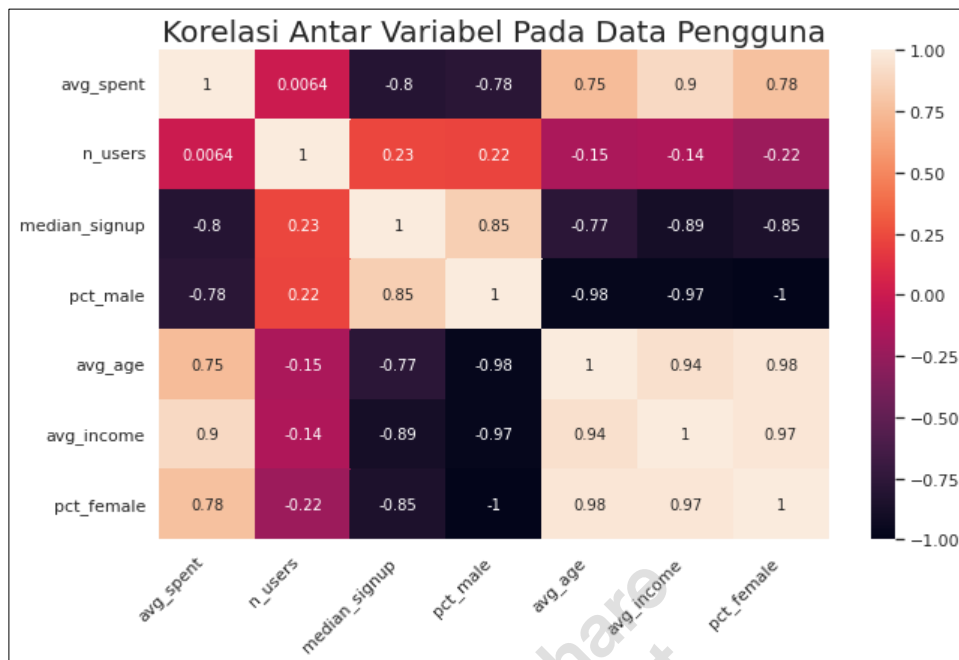
Tabel 5.27 Hasil penerapan fungsi agregat pada *spend group*

<i>spend group</i>	<i>avg spent</i>	<i>n_users</i>	<i>median signup</i>	<i>pct male</i>	<i>avg_age</i>	<i>avg income (per year)</i>	<i>pct female</i>
1	6	1483	2018	74%	51	53.942	26%
2	18	1484	2018	75%	49	53.427	25%
3	31	1482	2017	71%	50	55.855	29%
4	50	1481	2017	62%	52	60.079	38%
5	74	1483	2017	55%	55	65.341	45%
6	100	1483	2017	49%	56	68.162	51%
7	129	1481	2017	48%	56	69.490	52%
8	161	1483	2016	47%	56	71.810	53%
9	205	1482	2016	44%	56	76.542	56%
10	391	1483	2016	44%	57	79.405	56%



Gambar 5.19 Line plot pola demografi pengguna pada tiap *spend group*

Setelah memvisualisasikan hasil pemotongan kuantil pada variabel *spend group*, selanjutnya penulis ingin mencari tahu korelasi antar variabel demografi pengguna. Pengguna dikelompokkan menjadi sepuluh *spend group* berdasarkan jumlah total yang mereka belanjakan. Ada korelasi yang tinggi pada variabel pembelanjaan pengguna dengan beberapa fitur demografis pengguna. Saat pembelanjaan (*avg_spent*) meningkat, kita dapat melihat korelasi untuk setiap variabel seperti tahun pendaftaran (*median_signup*) dan persentase pengguna pria (*pct_male*) menurun. Sementara itu, pada variabel lain seperti usia (*avg_age*) dan pendapatan (*avg_income*) meningkat. Agar lebih mudah dalam melihat korelasi antarvariabel, penulis merancang diagram *heatmap*. Adapun hasil visualisasi data dalam bentuk *heatmap* dapat dilihat pada Gambar 5.20.

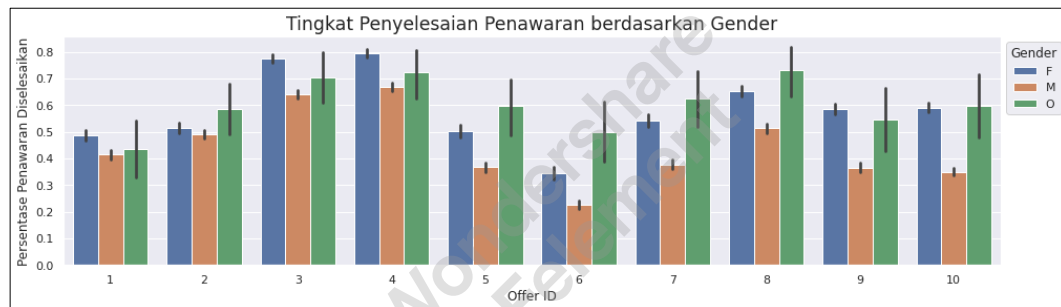


Gambar 5.20 Heatmap untuk korelasi antarvariabel demografi pengguna

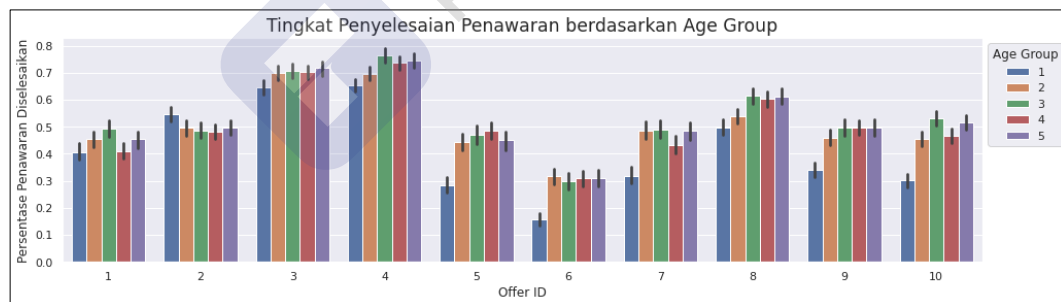
Dari diagram *heatmap* tersebut, penulis membuat beberapa kesimpulan tentang bagaimana pengguna menghabiskan uang mereka untuk membeli produk-produk Starbucks. Untuk variabel *signup_year*, pengguna yang telah menjadi pelanggan lama Starbucks cenderung menghabiskan uang lebih banyak. Untuk variabel *income*, pengguna yang memiliki pendapatan tinggi cenderung membelanjakan uang lebih banyak. Untuk variabel *age*, pembelanjaan pengguna meningkat seiring bertambahnya usia pengguna hingga ke usia tertentu. Kemudian pembelanjaan tersebut konsisten di paruh usia (sekitar usia 50-an). Untuk variabel *gender*, pengguna wanita cenderung menghabiskan lebih banyak uang daripada pengguna pria. Pola ini juga terlihat pada *box plot* yang telah dibuat sebelumnya.

Eksplorasi terakhir yang akan dilakukan pada data pengguna ialah mencari tahu apakah ada pola tertentu dalam penyelesaian penawaran pengguna. Dengan mengetahui pola demografi dalam penyelesaian penawaran pengguna, dapat diperoleh *insight* terkait kelompok pengguna mana yang memiliki kemauan yang besar untuk menyelesaikan penawaran. Sehingga tim *marketing* perusahaan bisa mengirimkan tipe penawaran yang efektif dan sesuai dengan karakteristik pengguna tersebut. Langkah pertama yang dilakukan yaitu menggabungkan

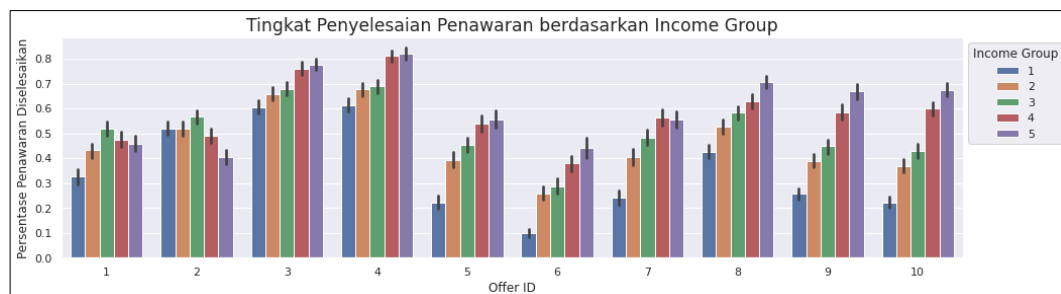
dataset profile.json dengan data penawaran hasil eksplorasi data sebelumnya. Kemudian dilakukan filter untuk memperoleh penawaran yang valid dengan peristiwa adalah *offer received* dan mengecualikan *offer group 3* (penawaran diselesaikan, tapi tidak dilihat). Lalu penulis juga merancang fungsi *plot_demographic_completion()* yang berfungsi memvisualisasikan pola tertentu dalam penyelesaian penawaran pengguna. Dan juga, visualisasi data ini didasarkan pada beberapa variabel demografi pengguna meliputi: *gender*, *age_group*, *income_group*, *signup_year*, dan *spend_group*. Adapun hasil visualisasi untuk pola demografi pengguna dalam penyelesaian penawaran dapat dilihat pada Gambar 5.21 hingga Gambar 5.75.



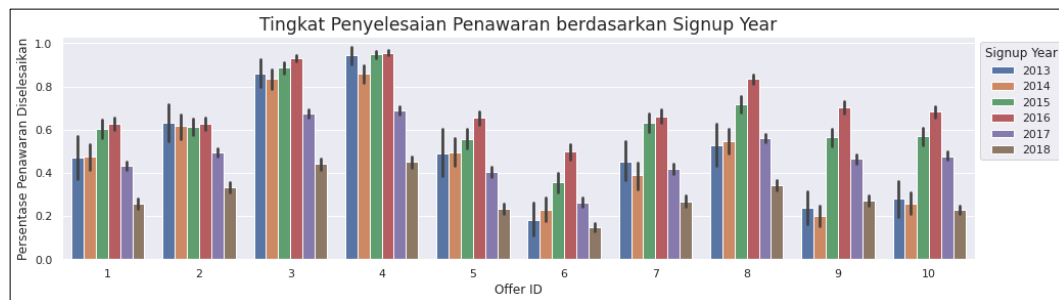
Gambar 5.21 Tingkat penyelesaian penawaran berdasarkan *gender*



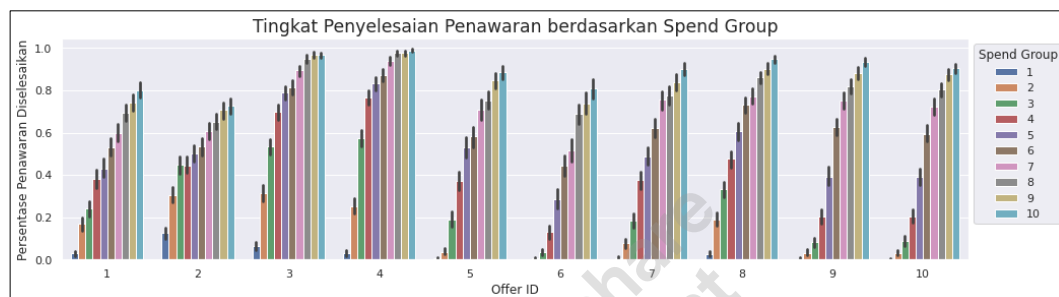
Gambar 5.22 Tingkat penyelesaian penawaran berdasarkan *age group*



Gambar 5.23 Tingkat penyelesaian penawaran berdasarkan *income group*



Gambar 5.24 Tingkat penyelesaian penawaran berdasarkan *signup year*



Gambar 5.25 Tingkat penyelesaian penawaran berdasarkan *spend group*

Tipe penawaran diurutkan berdasarkan seberapa sulit pengguna bisa menyelesaikannya meliputi: dua penawaran informasional, empat penawaran *discount*, dan empat penawaran BOGO. Penawaran *discount* lebih mudah diselesaikan daripada penawaran BOGO karena total pembeliannya dapat diakumulasikan dengan lebih dari satu transaksi. Dari beberapa *bar plot* yang telah ditampilkan, penulis menarik beberapa kesimpulan yang dapat menjadi *insight* terkait pola demografi pengguna dalam penyelesaian penawaran. Variabel *gender*, pengguna wanita memiliki tingkat penyelesaian yang lebih tinggi daripada pengguna pria untuk semua tipe penawaran. Variabel *age_group*, kelompok usia termuda (*age group 1*) memiliki tingkat penyelesaian terendah daripada keempat kelompok lainnya. Perbedaan pada masing-masing kelompok usia juga dipengaruhi oleh tingkat kesulitan penawaran.

Variabel *income_group*, untuk sebagian besar, ada hubungan kuat antara pendapatan tinggi dengan tingkat penyelesaian penawaran. Seperti yang terjadi pada variabel *age_group*, perbedaan terbesar antara *income group 1* dengan keempat kelompok lainnya adalah tingkat kesulitan penawaran. Karena pengguna dalam *age group 1* memiliki penghasilan yang lebih sedikit, maka mereka

membelanjakan lebih sedikit dan cenderung tidak menyelesaikan penawaran dengan tingkat kesulitan yang lebih tinggi. Variabel *signup_year*, pengguna yang mendaftar pada tahun 2016 memiliki tingkat penyelesaian tertinggi (diikuti oleh pengguna tahun 2015) dan pengguna yang mendaftar pada tahun 2018 memiliki tingkat penyelesaian terendah. Pengguna baru sepertinya sedikit ragu untuk membelanjakan banyak uang dibandingkan dengan pengguna yang lebih lama. Selain itu, diketahui bahwa mayoritas pendaftar dari tahun 2018 adalah pengguna yang berusia muda (yang juga tidak menghabiskan banyak uang).

5.2.3 Hasil Pembobotan FMT

Setelah memperoleh *insight* dari beberapa proses dalam tahap *exploratory data analysis* (EDA), tahap selanjutnya yang akan dilakukan adalah menampilkan hasil dari tahap pembobotan FMT. Sebelum itu, dikarenakan tahap pembobotan FMT dilakukan di *notebook* yang berbeda, maka akan dilakukan proses *read dataset* dan *data profiling* lagi. Perbedaannya, *dataset* yang dibaca pada tahap ini merupakan *dataset* hasil proses manipulasi dan eksplorasi yang telah dilakukan sebelumnya. Oleh karena itu, terdapat sedikit perbedaan struktur hingga jumlah baris antara *dataset* ini dengan *dataset* awal. Untuk *dataset profile*, setelah melalui berbagai proses EDA, dimensi *dataset*-nya menjadi 14.825 baris dan 10 kolom. Untuk *dataset transcript* memiliki dimensi *dataset* yaitu 272.388 baris dan 7 kolom. Sedangkan *dataset offers*, awalnya merupakan *dataset portfolio*, setelah dilakukan manipulasi dan eksplorasi data, kini terdapat 148.431 baris dan 26 kolom. Adapun hasil proses *data profiling* dapat dilihat pada Tabel 5.28 hingga Tabel 5.31.

Tabel 5.28 Hasil proses *data profiling* untuk *dataset profile* (lanjutan)

No.	Kolom	Data Kosong	Tipe Data
1.	<i>gender</i>	<i>non-null</i>	<i>object</i>
2.	<i>Age</i>	<i>non-null</i>	<i>integer</i>
3.	<i>income</i>	<i>non-null</i>	<i>integer</i>
4.	<i>user_id</i>	<i>non-null</i>	<i>integer</i>
5.	<i>signup_date</i>	<i>non-null</i>	<i>datetime</i>
6.	<i>signup_year</i>	<i>non-null</i>	<i>integer</i>
7.	<i>age_group</i>	<i>non-null</i>	<i>integer</i>
8.	<i>income_group</i>	<i>non-null</i>	<i>integer</i>
9.	<i>amount</i>	<i>non-null</i>	<i>float</i>
10.	<i>spend_group</i>	<i>non-null</i>	<i>integer</i>

Tabel 5.29 Hasil proses *data profiling* untuk *dataset transcript* (lanjutan)

No.	Kolom	Data Kosong	Tipe Data
1.	<i>index</i>	<i>non-null</i>	<i>integer</i>
2.	<i>event</i>	<i>non-null</i>	<i>object</i>
3.	<i>Time</i>	<i>non-null</i>	<i>integer</i>
4.	<i>amount</i>	<i>non-null</i>	<i>float</i>
5.	<i>offer_id</i>	<i>non-null</i>	<i>integer</i>
6.	<i>reward</i>	<i>non-null</i>	<i>integer</i>
7.	<i>user_id</i>	<i>non-null</i>	<i>integer</i>

Tabel 5.30 Hasil proses *data profiling* untuk *dataset offers*

No.	Kolom	Data Kosong	Tipe Data
1.	<i>index</i>	<i>non-null</i>	<i>integer</i>
2.	<i>event</i>	<i>non-null</i>	<i>object</i>
3.	<i>Time</i>	<i>non-null</i>	<i>integer</i>
4.	<i>offer_id</i>	<i>non-null</i>	<i>integer</i>
5.	<i>user_id</i>	<i>non-null</i>	<i>integer</i>
6.	<i>amount</i>	<i>non-null</i>	<i>float</i>
7.	<i>reward</i>	<i>non-null</i>	<i>integer</i>

Tabel 5.31 Hasil proses *data profiling* untuk *dataset offers* (lanjutan)

No.	Kolom	Data Kosong	Tipe Data
8.	<i>difficulty</i>	<i>non-null</i>	<i>integer</i>
9.	<i>duration</i>	<i>non-null</i>	<i>integer</i>
10.	<i>offer_type</i>	<i>non-null</i>	<i>object</i>
11.	<i>mobile</i>	<i>non-null</i>	<i>integer</i>
12.	<i>social</i>	<i>non-null</i>	<i>integer</i>
13.	<i>Web</i>	<i>non-null</i>	<i>integer</i>
14.	<i>expire_time</i>	<i>non-null</i>	<i>integer</i>
15.	<i>oid_uid</i>	<i>non-null</i>	<i>object</i>
16.	<i>viewed</i>	<i>non-null</i>	<i>integer</i>
17.	<i>completed</i>	<i>non-null</i>	<i>integer</i>
18.	<i>group</i>	<i>non-null</i>	<i>integer</i>
19.	<i>gender</i>	<i>non-null</i>	<i>object</i>
20.	<i>Age</i>	<i>non-null</i>	<i>integer</i>
21.	<i>income</i>	<i>non-null</i>	<i>integer</i>
22.	<i>signup_date</i>	<i>non-null</i>	<i>datetime</i>
23.	<i>signup_year</i>	<i>non-null</i>	<i>integer</i>
24.	<i>age_group</i>	<i>non-null</i>	<i>integer</i>
25.	<i>income_group</i>	<i>non-null</i>	<i>integer</i>
26.	<i>Spend_group</i>	<i>non-null</i>	<i>integer</i>

Setelah diperoleh hasil *read dataset* dan *data profiling*, selanjutnya penulis melakukan pemfilteran terhadap *dataset transcript* untuk memperoleh data transaksi pelanggan. Kemudian dilakukan penggabungan data transaksi pelanggan dengan *dataset profile*. Lalu penulis membuat variabel baru yaitu *gte1* untuk memperoleh data transaksi pelanggan dengan total minimal 1 dolar. Selanjutnya penulis melakukan proses agregasi dengan menghitung banyaknya data pada variabel *gte* untuk memperoleh data variabel *frequency*, serta menjumlahkan total transaksi pada variabel *amount* untuk memperoleh data variabel *monetary*. Sementara itu, untuk memperoleh data variabel *tenure*, penulis menerapkan

lambda expression pada Tabel 5.8. Adapun sekilas hasil proses pembobotan model dan analisis statistik sederhana untuk variabel FMT (*frequency*, *monetary*, dan *tenure*) dapat dilihat pada Tabel 5.32 dan Tabel 5.33.

Tabel 5.32 Sekilas hasil pembobotan model untuk variabel FMT

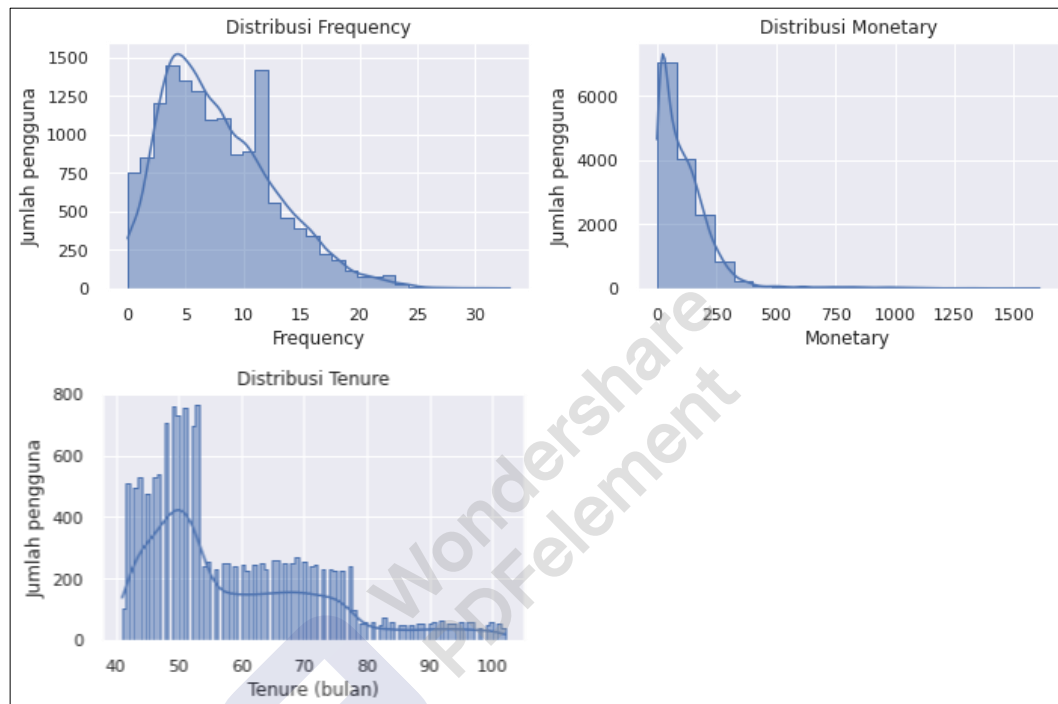
<i>user_id</i>	<i>frequency</i>	<i>monetary (USD)</i>	<i>tenure (month)</i>
1	3	77,01	54
2	7	159,27	56
3	3	57,73	44
4	3	36,43	47
5	3	15,62	50

Tabel 5.33 Analisis statistik sederhana untuk model variabel FMT

Fungsi Statistik	<i>frequency</i>	<i>monetary (usd)</i>	<i>tenure (month)</i>
<i>count</i>	14.825	14.825	14.825
<i>mean</i>	7,83	117,02	58,73
<i>std</i>	4,82	129,96	13,97
<i>min</i>	0	0	41
25%	4	31,45	48
50%	7	87,04	53
75%	11	160,90	68
<i>max</i>	33	1608,69	102

Langkah berikutnya yang dilakukan adalah memvisualisasikan distribusi data pada model FMT. Dengan melihat distribusi data, penulis menjadi tahu jenis kemencengan (*skewness*) yang terjadi pada model FMT. Penulis memanfaatkan fungsi *fmt_distributionplot()* untuk menampilkan visualisasi data dalam bentuk histogram. Setelah histogram ditampilkan, dapat diketahui bahwa distribusi data pada model FMT mengalami kemencengan positif (*positively skewed*). *Skewness* yang bernilai positif berarti sebagian besar distribusi data berada di nilai rendah. Dikarenakan distribusi data pada model FMT tidak simetris, maka nantinya penulis akan melakukan proses normalisasi data. Distribusi data telah menjadi salah satu faktor penting dalam konsep statistika peluang. Distribusi data yang

simetris dianggap penting karena beberapa alasan, mulai dari meningkatkan obyektivitas penilaian hingga mengelompokkan entitas ke dalam satu kriteria yang sama sehingga dapat menghindari terjadinya bias (penilaian yang condong di satu kategori saja). Adapun hasil visualisasi data dalam bentuk histogram untuk distribusi model FMT dapat dilihat pada Gambar 5.26.



Gambar 5.26 Hasil visualisasi untuk distribusi data model FMT

Setelah mengetahui bahwa distribusi data model FMT tidak simetris, selanjutnya penulis akan melakukan proses normalisasi data. Namun sebelum itu, penulis melakukan pemotongan kuantil pada variabel FMT untuk keperluan proses agregasi yang ada di tahap *clustering*. Dengan pemotongan kuantil, maka rentang nilai pada variabel FMT juga berubah sesuai dengan jumlah pemotongannya. Untuk variabel *frequency*, dilakukan pemotongan sebanyak enam bagian. Untuk variabel *monetary*, dilakukan pemotongan sebanyak delapan bagian. Sedangkan untuk variabel *tenure*, dilakukan pemotongan sebanyak tiga bagian. Lalu dilakukan penggabungan *string* pada variabel hasil pemotongan kuantil, yang mana akan menjadi variabel acuan dalam proses agregasi nantinya. Setelah dilakukan penggabungan *string*, data tersebut akan digabungkan ke

dataset profile. Dan juga, dilakukan pembuatan variabel *male* melalui konversi biner pada variabel *gender* untuk memudahkan algoritma dalam identifikasi jenis kelamin pengguna. Adapun sekilas hasil pemotongan kuantil pada model FMT dapat dilihat pada Tabel 5.34.

Tabel 5.34 Sekilas hasil pemotongan kuantil pada model FMT

<i>user_id</i>	<i>frequency</i>	<i>monetary (USD)</i>	<i>tenure (month)</i>	<i>F</i>	<i>M</i>	<i>T</i>	<i>FMT</i>
1	3	77,01	54	1	4	2	142
2	7	159,27	56	3	6	2	362
3	3	57,73	44	1	4	1	141
4	3	36,43	47	1	3	1	131
5	3	15,62	50	1	2	1	121

Setelah memperoleh hasil pemotongan kuantil, selanjutnya penulis akan menerapkan normalisasi data pada model FMT. Normalisasi data ini dilakukan dengan tujuan mengatasi permasalahan distribusi data tak simetris yang disebabkan oleh rentang data yang berbeda dari tiap variabel. Rentang data ini nantinya akan disamakan namun tetap memuat informasi yang sama dengan rentang sebelum dilakukan normalisasi data. Pada penelitian ini, penulis memanfaatkan *Min-Max Normalization* sebagai metode dalam melakukan normalisasi data. Rentang data yang akan dipakai dalam normalisasi yaitu antara 0 hingga 1. Selain menerapkan normalisasi data pada model FMT, ada dua variabel demografi pengguna yaitu *age* dan *income* agar model yang dihasilkan menjadi lebih komprehensif. Adapun hasil normalisasi data dapat dilihat pada Tabel 5.35.

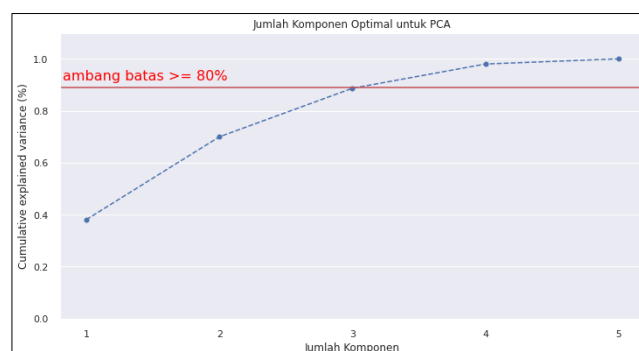
Tabel 5.35 Hasil normalisasi data menggunakan *Min-Max Normalization*

No.	<i>age_norm</i>	<i>inc_norm</i>	<i>f_norm</i>	<i>m_norm</i>	<i>t_norm</i>
1.	0,445783	0,911111	0,090909	0,047871	0,213115
2.	0,686747	0,777778	0,212121	0,099006	0,245902
3.	0,602410	0,444444	0,090909	0,035886	0,049180
4.	0,566265	0,255556	0,090909	0,022646	0,098361
5.	0,481928	0,233333	0,090909	0,009710	0,147541

5.2.4 Hasil Principal Component Analysis

Setelah memperoleh hasil pembobotan dan normalisasi pada model FMT, selanjutnya akan dilakukan tahap *principal component analysis* (PCA). Tahap PCA ini bertujuan mereduksi jumlah dimensi pada data hasil normalisasi. Jumlah dimensi pada sebuah data perlu direduksi karena dengan adanya jumlah dimensi yang tinggi dapat menyebabkan turunnya performa algoritma *machine learning*. Selain itu, indera penglihatan manusia juga hanya mampu digunakan untuk melihat obyek hingga pada batasan tiga dimensi. Oleh karena itu, jika tetap dipaksakan menggunakan jumlah dimensi yang tinggi sebagai *input* pada proses *clustering*, maka akan timbul permasalahan yang biasa disebut *curse of dimensionality*.

Langkah pertama yang dilakukan pada tahap PCA ialah menentukan jumlah dimensi (komponen) yang optimal. Pada dasarnya, penentuan ini tidak hanya memilih jumlah dimensi terkecil, namun juga harus tetap memerhatikan variansi atau banyaknya informasi yang dimuat dalam sebuah komponen. Untuk menentukan jumlah dimensi yang optimal, penulis membuat visualisasi data dalam bentuk *line plot* yang akan membandingkan masing-masing komponen PCA dengan variansi kumulatif data. Data *input* yang digunakan dalam proses PCA adalah data hasil normalisasi. Hasilnya diperoleh bahwa jumlah dimensi yang optimal yaitu tiga komponen dengan variansi data sebesar 89%. Adapun hasil visualisasi data dalam bentuk *line plot* dapat dilihat pada Gambar 5.27.



Gambar 5.27 Hasil visualisasi untuk jumlah komponen PCA yang optimal

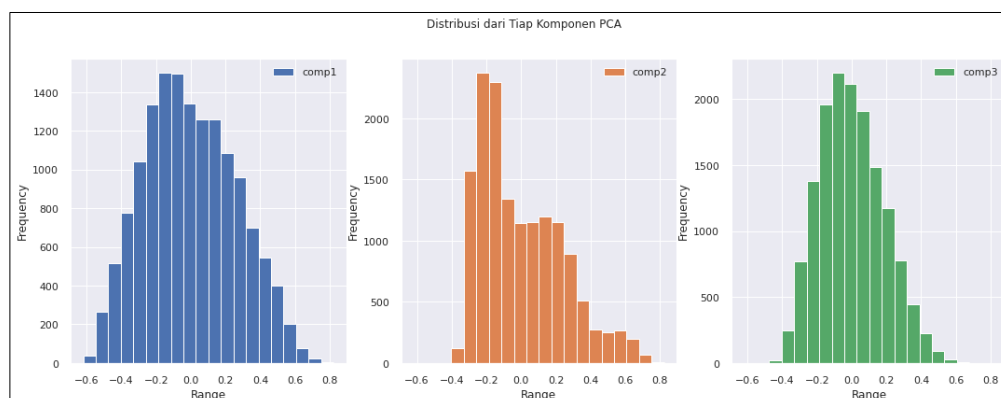
Setelah memperoleh jumlah dimensi yang optimal, maka langkah selanjutnya adalah menerapkan PCA menggunakan jumlah dimensi yang

didapatkan dari hasil visualisasi data. Lalu akan dilakukan analisis statistik sederhana untuk mengetahui gambaran umum tentang data hasil proses PCA. Kemudian penulis melakukan visualisasi data lagi dalam bentuk histogram untuk melihat distribusi data. Hasilnya diperoleh bahwa kemencengan distribusi data berbentuk simetris (menyerupai bentuk lonceng), meskipun untuk komponen PCA kedua memiliki bentuk yang agak *positively skewed*.

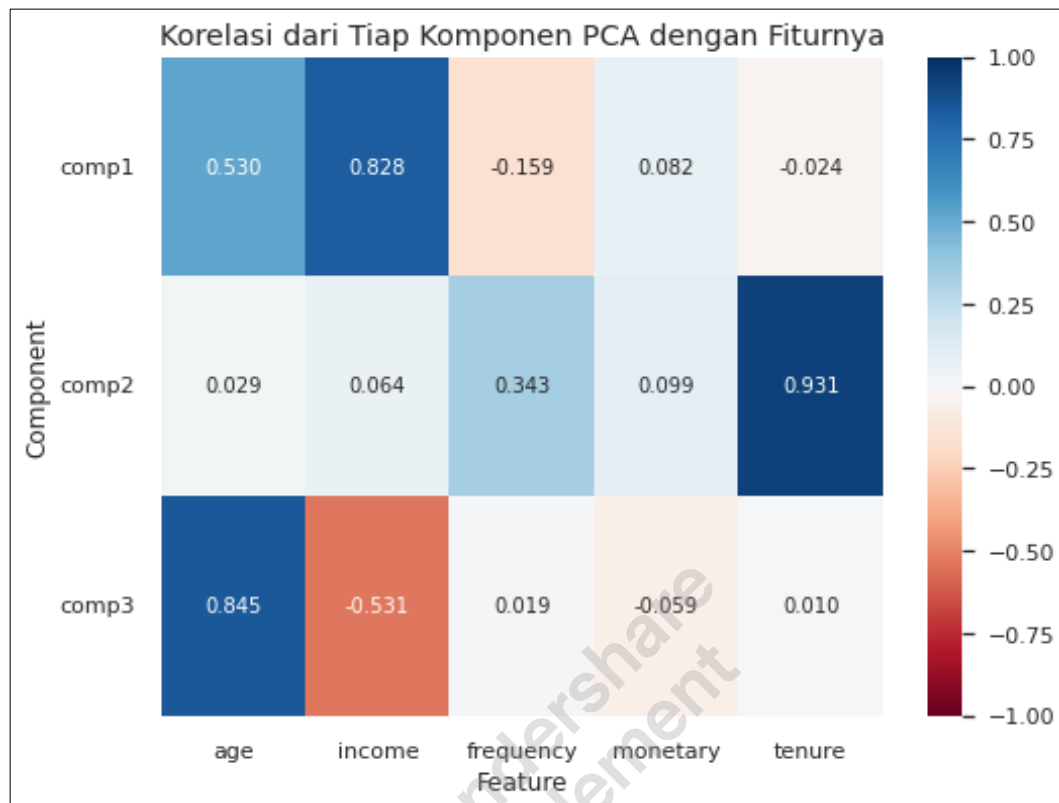
Selain itu, penulis juga memvisualisasikan data dalam bentuk *heatmap* untuk melihat korelasi dan dominasi variabel terhadap masing-masing komponen PCA. Hasilnya diperoleh bahwa komponen pertama berkorelasi dengan pengguna berusia paruh baya dan berpendapatan tinggi. Untuk komponen kedua berkorelasi dengan pengguna yang memiliki frekuensi pembelian sedang dan merupakan pengguna lama. Yang terakhir adalah komponen ketiga berkorelasi dengan pengguna berusia tua dan berpendapatan rendah. Adapun sekilas hasil penerapan PCA serta visualisasi data dalam bentuk histogram dan *heatmap* dapat dilihat pada Tabel 5.36, Gambar 5.28, dan Gambar 5.29.

Tabel 5.36 Sekilas hasil penerapan reduksi dimensi dengan PCA

No.	<i>comp1</i>	<i>comp2</i>	<i>comp3</i>
1.	0,455803	-0,091492	-0,270981
2.	0,457330	-0,015919	0,003068
3.	0,155220	-0,270855	0,108401
4.	-0,022680	-0,239550	0,179482
5.	-0,088051	-0,198884	0,121273



Gambar 5.28 Hasil visualisasi untuk distribusi data komponen PCA



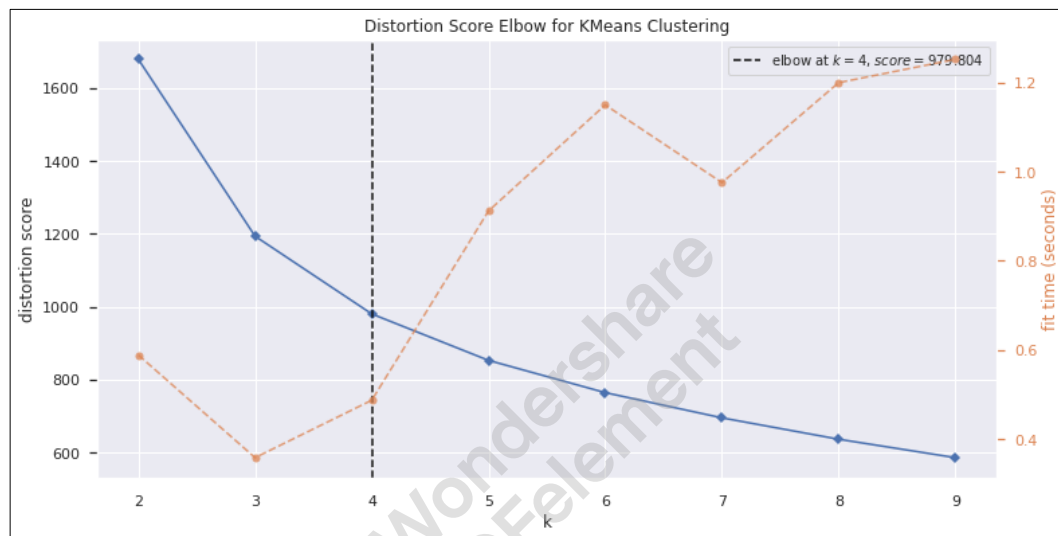
Gambar 5.29 Hasil visualisasi untuk korelasi data komponen PCA

5.2.5 Hasil Proses Clustering dan Uji Performa

Setelah melalui tahap PCA untuk mereduksi jumlah dimensi data, langkah berikutnya adalah melakukan *clustering* menggunakan algoritma *K-Means*. Algoritma *K-Means* merupakan salah satu algoritma *machine learning* yang *unsupervised* yang paling sering digunakan untuk mengelompokkan data. Algoritma *K-Means* termasuk dalam algoritma *unsupervised* karena dalam melakukan pengelompokkan data, algoritma ini tidak memerlukan label atau data latih untuk mempelajari data. Pada penelitian ini, sebelum menerapkan algoritma *K-Means* untuk mengklaster data, penulis perlu menentukan jumlah klaster (k) sebagai salah satu *input* agar algoritma dapat berjalan dengan baik.

Maka dari itu, metode *Elbow* digunakan untuk membantu penulis dalam penentuan jumlah klaster yang optimal. Metode ini membandingkan nilai *error* dari tiap jumlah klaster, mencari jumlah klaster yang mengalami penurunan terbesar, serta tetap menjaga supaya jumlah klaster tidak terlalu tinggi. Kemudian penulis juga memanfaatkan *library* Python bernama *kelbow visualizer* untuk

memvisualisasikan jumlah kluster yang optimal dalam bentuk *elbow plot*. Hasilnya diperoleh bahwa empat adalah jumlah kluster optimal karena menjadi titik siku pada visualisasi *elbow plot*. Selain itu, pada grafik *line plot* juga diperoleh waktu 0,5 detik untuk melakukan *K-means clustering* dengan jumlah kluster yaitu empat. Adapun hasil visualisasi grafik berupa *line plot* untuk menampilkan jumlah kluster yang optimal dapat dilihat pada Gambar 5.30.



Gambar 5.30 Hasil visualisasi untuk jumlah kluster yang optimal

Setelah diperoleh jumlah kluster yang optimal, penulis melanjutkan proses pada tahap *clustering* dengan menerapkan algoritma *K-Means* ke dalam sistem. Dengan *input* berupa jumlah kluster dan data hasil proses PCA, nantinya akan diperoleh data dengan masing-masing kluster yang memiliki karakteristik serupa. Selain itu, penulis juga membuat fungsi *segment_means()* untuk melakukan proses agregasi data hasil *clustering*. Proses agregasi ini akan menjelaskan gambaran umum dari data di masing-masing kluster. Proses agregasi ini akan menerapkan fungsi rata-rata pada beberapa variabel yang diperoleh dari tahap-tahap sebelumnya. Beberapa variabel tersebut antara lain: *age*, *income*, *frequency*, *monetary*, *tenure*, *comp1*, *comp2*, dan *comp3*. Penulis juga melakukan perhitungan jumlah dan persentase pengguna dari masing-masing kluster dengan memanfaatkan data hasil pemotongan kuantil pada model FMT. Adapun data hasil penerapan fungsi *segment_means()* dapat dilihat pada Gambar 5.31.

	age	income	frequency	monetary	tenure	comp1	comp2	comp3	n_users	pct_users
cluster										
0	62.98	92378.19	6.33	173.24	58.47	0.31	0.01	-0.07	4191	28.27
1	34.75	51393.33	7.70	80.53	50.91	-0.25	-0.14	-0.12	3628	24.47
2	52.02	56974.54	11.87	135.00	78.51	-0.12	0.34	0.03	3221	21.73
3	65.73	56143.20	6.18	74.47	49.70	-0.00	-0.16	0.17	3785	25.53

Gambar 5.31 Data hasil penerapan fungsi *segment_means()*

Setelah berhasil mengelompokkan data menjadi empat klaster, penulis selanjutnya akan melakukan *string mapping* untuk memberi nama pada tiap klaster. Pemilihan nama ini secara relatif bergantung pada kebijakan perusahaan. Namun dikarenakan penulis tidak bekerja sama secara langsung dengan perusahaan, lantas penulis mengajukan ide terkait nama pada tiap klaster. Penamaan klaster dalam penelitian ini mengacu pada tingkat pembelian dari tiap pengguna. Jika pengguna tersebut memiliki tingkat pembelian yang tinggi, maka akan dikelompokkan ke klaster tertinggi. Ide nama-nama klaster yang diajukan oleh penulis antara lain: *bronze customers*, *silver customers*, *gold customers*, dan *diamond customers*. Adapun hasil penamaan klaster dalam *string mapping* dapat dilihat pada Tabel 5.37.

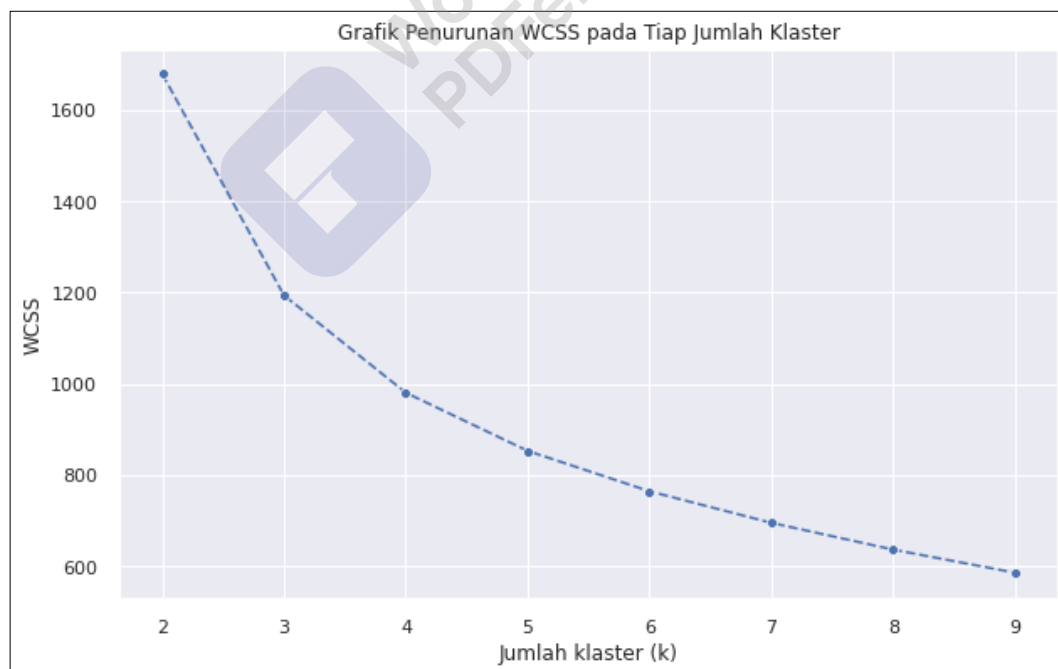
Tabel 5.37 Hasil penamaan klaster dalam *string mapping*

No.	Nomor Klaster	Nama Klaster	Definisi Klaster
1.	<i>Cluster 0</i>	<i>Gold Customers</i>	Pelanggan Berharga
2.	<i>Cluster 1</i>	<i>Silver Customers</i>	Pelanggan Cukup Berharga
3.	<i>Cluster 2</i>	<i>Diamond Customers</i>	Pelanggan Paling Berharga
4.	<i>Cluster 3</i>	<i>Bronze Customers</i>	Pelanggan Kurang Berharga

Setelah dilakukan *string mapping* untuk pemberian nama pada tiap klaster, langkah berikutnya yaitu menguji performa algoritma *K-Means* dalam melakukan pengelompokkan data. Uji performa algoritma *K-Means* dilakukan menggunakan metode *Elbow*. Metode ini akan menilai skor jumlah kuadrat (*Sum of Squares*) dari tiap klaster dan mencari klaster yang membentuk sudut siku (*elbow*) pada hasil visualisasinya. Pada metode *Elbow* terdapat beberapa jenis evaluasi performa yang bisa digunakan. Penulis memanfaatkan jenis evaluasi jumlah

kuadrat bernama *Within-Cluster Sum of Squares* (WCSS). Jenis evaluasi ini merupakan evaluasi yang mengukur kedekatan antarobyek dalam suatu kluster. Nilai WCSS yang kecil mengindikasikan kemiripan antarobyek dalam suatu kluster. Perhitungan WCSS tiap kluster akan dihitung menggunakan Persamaan 2.16.

Kemudian penulis memvisualisasikan hasil perhitungan WCSS dari tiap kluster agar dapat dilihat kluster mana yang membentuk sudut siku (*elbow*). Dari sembilan kluster yang dilakukan uji performa, grafik visualisasi menunjukkan bahwa penurunan terbesar terjadi pada kluster ketiga menuju kluster keempat. Sehingga pada kluster kelima mulai mengalami penurunan yang landai atau memiliki selisih nilai WCSS yang kecil. Hasilnya diperoleh bahwa jumlah kluster sebanyak empat memang merupakan kluster terbaik karena juga telah membentuk sudut siku (*elbow*). Adapun hasil visualisasi data serta perhitungan nilai WCSS dari masing-masing kluster pengguna dapat dilihat pada Gambar 5.32 dan Tabel 5.38.



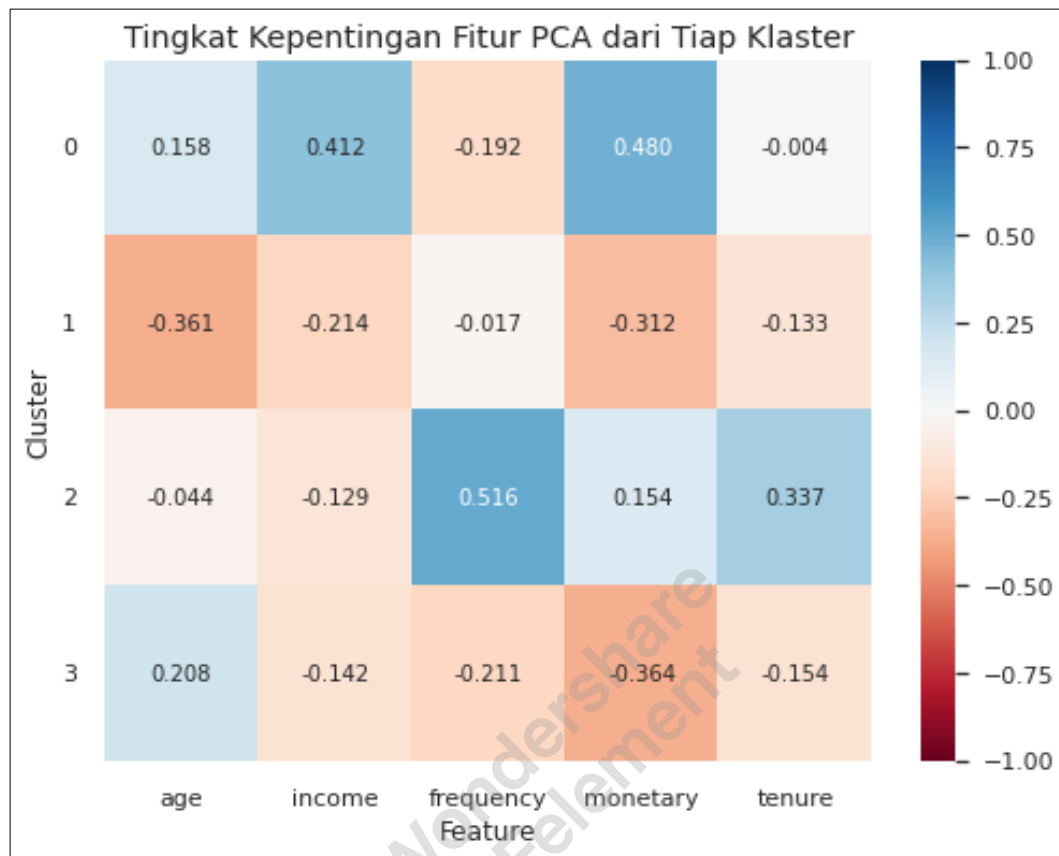
Gambar 5.32 Hasil visualisasi data untuk perhitungan nilai WCSS

Tabel 5.38 Hasil perhitungan dan selisih nilai WCSS dari tiap klaster

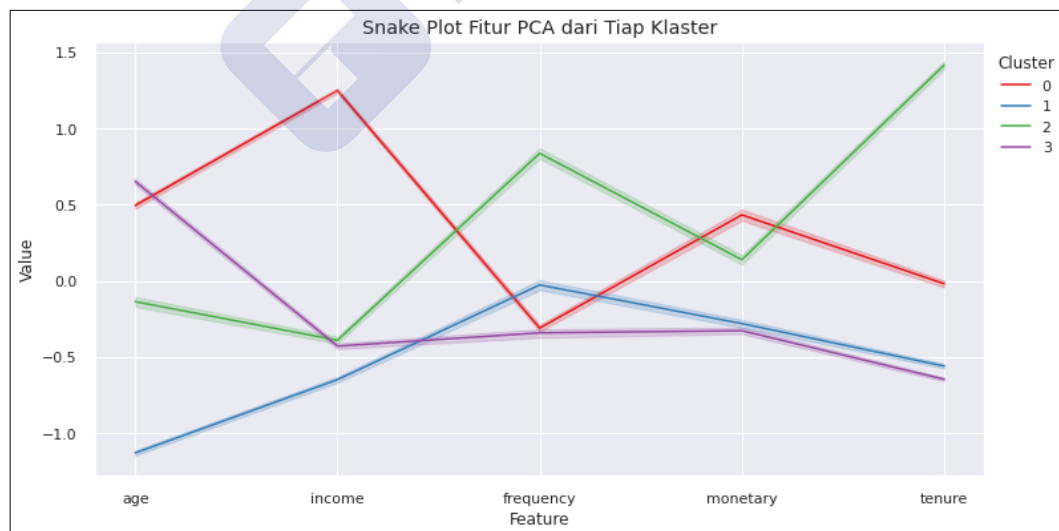
No.	Jumlah Klaster	Nilai WCSS	Selisih Nilai WCSS
1.	$k = 2$	1.678,80	0
2.	$k = 3$	1.193,21	485,59
3.	$k = 4$	979,80	213,41
4.	$k = 5$	852,33	127,47
5.	$k = 6$	763,74	88,59
6.	$k = 7$	694,53	69,21
7.	$k = 8$	635,82	58,71
8.	$k = 9$	585,20	50,62

5.2.6 Hasil Visualisasi Data

Setelah memperoleh hasil uji performa dari algoritma *K-Means*, selanjutnya penulis akan menampilkan data hasil proses *clustering* ke dalam berbagai bentuk diagram. Penulis memvisualisasikan data dalam banyak grafik dengan tujuan agar data hasil proses *clustering* dapat lebih mudah dipahami, bahkan oleh seseorang yang awam di bidang teknologi. Pada diagram pertama, penulis membuat visualisasi data dalam bentuk *heatmap* untuk menggambarkan tingkat kepentingan variabel demografi PCA dari masing-masing klaster. Selain itu, penulis membuat diagram kedua dalam bentuk *snake plot* yang juga menunjukkan hubungan tiap klaster dengan variabel demografi pengguna. Hasilnya diperoleh bahwa pada klaster pertama (*gold customers*) memiliki hubungan yang kuat dengan variabel *high income* dan *high monetary*. Pada klaster kedua (*silver customers*) terdapat hubungan yang kuat untuk variabel *low age* dan *low monetary*. Lalu pada klaster ketiga (*diamond customers*) berkorelasi kuat dengan variabel *high frequency* dan *high tenure*. Sedangkan pada klaster keempat (*bronze customers*) mempunyai korelasi yang kuat untuk variabel *high age* dan *low monetary*. Adapun hasil visualisasi data dalam bentuk diagram *heatmap* dan *snake plot* dapat dilihat pada Gambar 5.33 dan Gambar 5.34.



Gambar 5.33 Heatmap untuk tingkat korelasi variabel pada tiap kluster

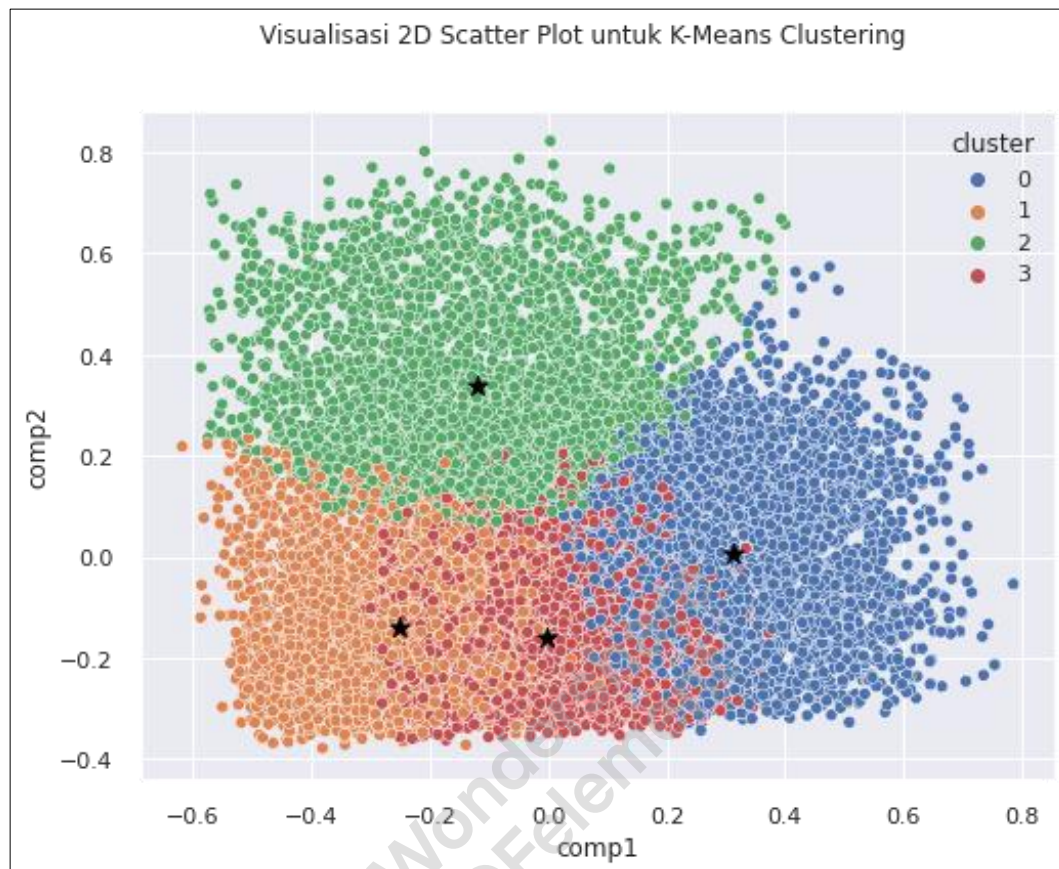


Gambar 5.34 Snake plot untuk tingkat korelasi variabel pada tiap kluster

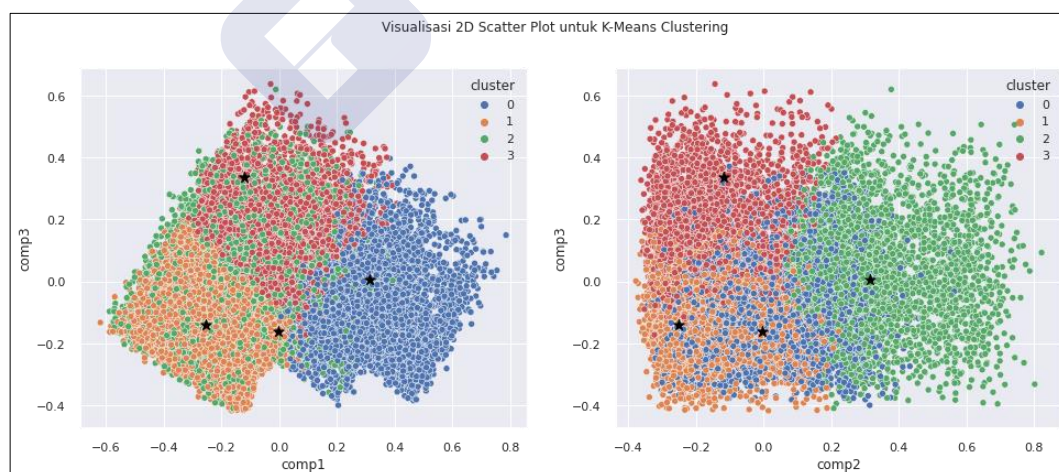
Setelah memvisualisasikan hubungan antara variabel demografi pengguna dengan masing-masing kluster, langkah berikutnya yang akan dilakukan ialah

memvisualisasikan hasil proses *clustering* dalam bentuk *scatter plot* dua dimensi dan tiga dimensi. *Scatter plot* dua dimensi adalah diagram yang digunakan untuk menjelaskan hubungan antara dua variabel pada sebuah data. *Scatter plot* tiga dimensi adalah diagram yang digunakan untuk menjelaskan hubungan antara tiga variabel pada sebuah data. Pada penelitian ini, penulis ingin mengetahui apakah variabel komponen yang diperoleh dari proses PCA berkorelasi dengan baik. Jika kedua variabel tersebut saling berkorelasi kuat, pada *scatter plot* akan tercipta hasil pengelompokkan data yang jelas dan dapat dibedakan antartitik *centroid*. Namun jika kedua variabel tersebut tidak berkorelasi kuat, maka akan terjadi hal sebaliknya yaitu hasil pengelompokkan data yang kurang teratur dan sedikit berantakan.

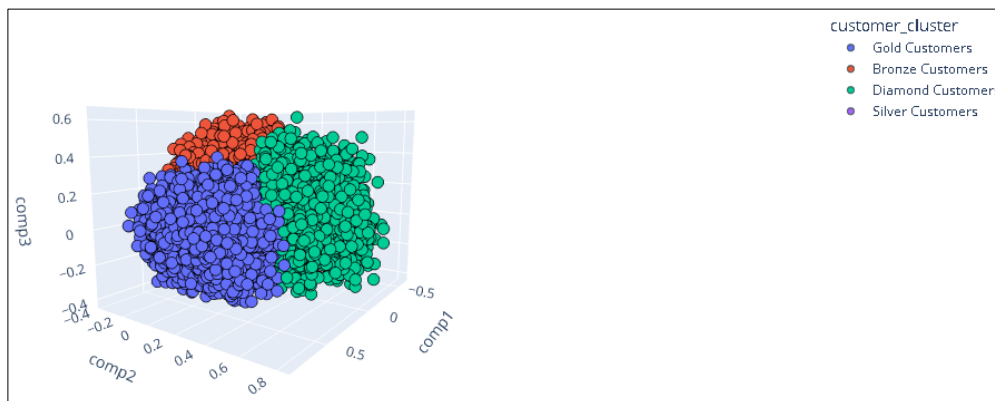
Hasilnya, pada *scatter plot* dua dimensi pertama yang memuat variabel *comp1* dan *comp2*, tercipta hasil pengelompokkan data yang cukup bagus. Titik data mendekati masing-masing *centroid* secara stabil sehingga batasan antarklaster terlihat jelas. Sedangkan, pada *scatter plot* dua dimensi kedua yang memuat variabel *comp1* dan *comp3* serta *scatter plot* dua dimensi ketiga yang memuat variabel *comp2* dan *comp3*, tercipta pengelompokkan data yang sedikit berantakan. Pada *scatter plot* kedua, klaster dua, tiga, dan empat menampilkan data yang saling berdekatan satu sama lain sehingga menampilkan ketidakteraturan. Begitu juga yang terjadi pada *scatter plot* ketiga, klaster satu, dua, dan empat mengalami pencampuran sehingga batasan antarklaster menjadi tidak jelas. Sedangkan *scatter plot* tiga dimensi menunjukkan hasil pengelompokkan yang cukup impresif. Titik data antarklaster mendekati *centroid* secara sempurna sehingga hasil visualisasi dalam bentuk *scatter plot* terlihat dengan jelas dan seragam. Adapun hasil visualisasi data dalam bentuk *scatter plot* dua dimensi dan tiga dimensi dapat dilihat pada Gambar 5.35 hingga Gambar 5.37.



Gambar 5.35 Scatter plot dua dimensi untuk hubungan *comp1* dengan *comp2*

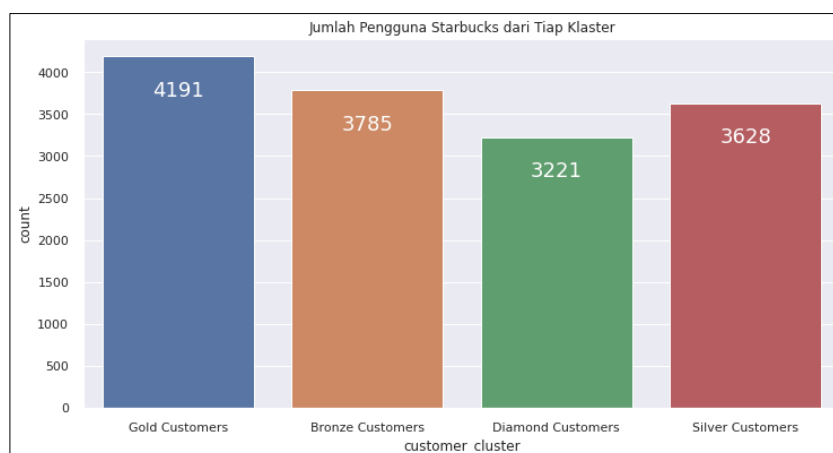


Gambar 5.36 Scatter plot dua dimensi untuk hubungan *comp1* dengan *comp3* serta hubungan *comp2* dengan *comp3*



Gambar 5.37 Scatter plot tiga dimensi untuk *comp1*, *comp2*, dan *comp3*

Setelah memvisualisasikan hasil *clustering* dalam bentuk *scatter plot*, langkah selanjutnya adalah membuat visualisasi data dalam bentuk *count plot*. Diagram ini akan memberikan gambaran terkait jumlah pengguna yang ada di masing-masing klaster. Dengan mengetahui informasi tersebut, penulis menjadi tahu bagaimana persebaran data dari tiap klaster pengguna. Selain itu, proporsi pengguna yang seimbang di tiap klaster bisa mengindikasikan bahwa hasil *clustering* menggunakan algoritma *K-Means* dilakukan dengan baik. Pada hasil *count plot* diperoleh informasi bahwa dari 14.825 data pengguna, klaster pertama (*gold customers*) memiliki persentase pengguna sebanyak 28,26% atau 4.191 orang, klaster kedua (*silver customers*) memiliki 24,47% atau 3.628 pengguna, klaster ketiga (*diamond customers*) memiliki 21,72% atau 3.221 orang, dan klaster keempat (*bronze customers*) memiliki 25,53% atau 3.785 orang. Adapun visualisasi dalam bentuk *count plot* dapat dilihat pada Gambar 5.38.

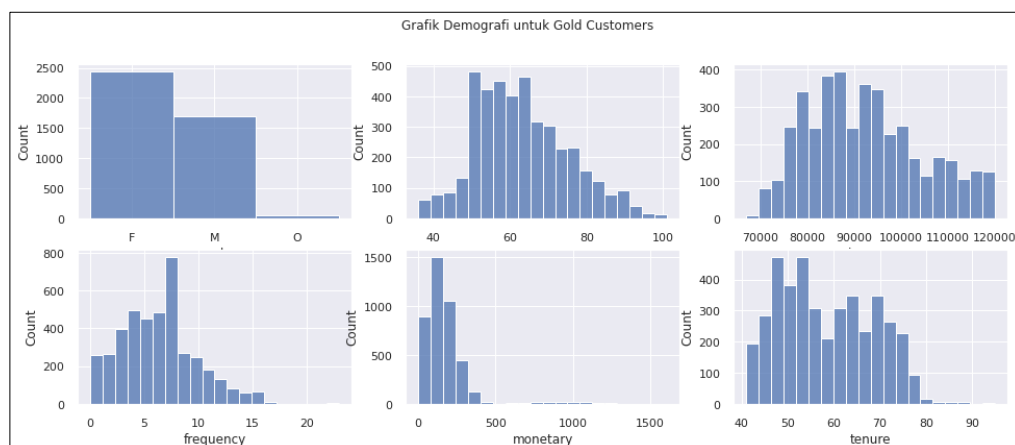


Gambar 5.38 Hasil visualisasi untuk jumlah pengguna tiap klaster

Visualisasi data terakhir yang akan dibuat adalah histogram yang akan menjelaskan demografi pengguna pada tiap kluster. Variabel demografi pengguna yang digunakan pada proses ini meliputi: *male*, *age*, *income*, *frequency*, *monetary*, dan *tenure*. Kemudian penulis membuat fungsi *plot_user_demographic()* yang bertujuan untuk memvisualisasikan data demografi pengguna tiap kluster ke dalam bentuk histogram. Lalu penulis juga melakukan filter data untuk melakukan *data profiling* ke masing-masing kluster pengguna. *Data profiling* ini perlu dilakukan untuk memperoleh angka detail pada variabel demografi pengguna sebelum divisualisasikan ke dalam histogram. Adapun informasi statistik, visualisasi data, dan penjelasan dari tiap kluster pengguna dapat dilihat pada Tabel 5.39 hingga 5.41 serta Gambar 5.39 hingga 5.41.

Tabel 5.39 Informasi statistik untuk kluster *gold customers*

No.	Fungsi Statistik	<i>age</i>	<i>income</i>	<i>frequency</i>	<i>monetary</i>	<i>Tenure</i>
1.	<i>count</i>	4.191	4.191	4.191	4.191	4.191
2.	<i>mean</i>	62,98	92.378,19	6,33	173,24	58,46
3.	<i>std</i>	12,25	12.303,03	3,49	160,88	10,16
4.	<i>min</i>	36	67.000	0	0	41
5.	25%	54	83.000	4	88,15	50
6.	50%	61	91.000	6	143,22	57
7.	75%	71	100.000	9	209,99	67
8.	<i>max</i>	101	120.000	23	1608,69	95

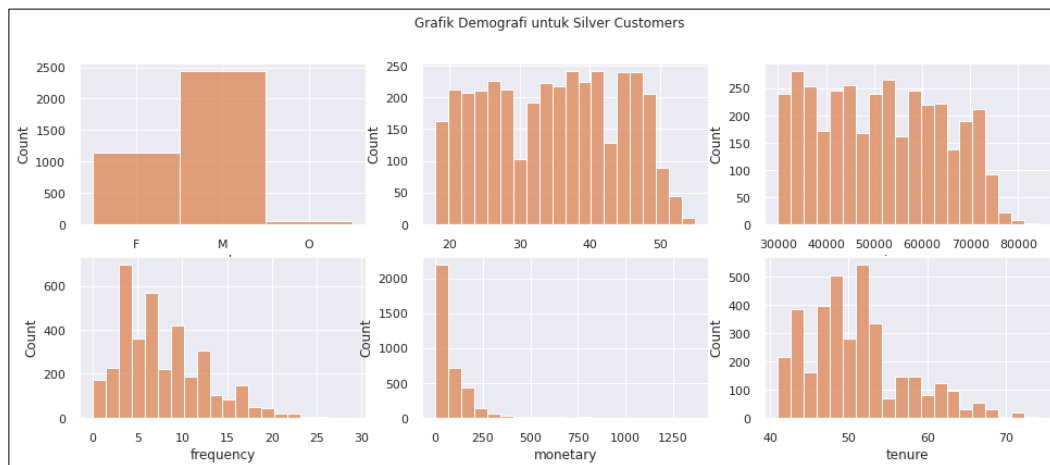


Gambar 5.39 Hasil visualisasi data untuk kluster *gold customers*

Pada Tabel 5.39 dan Gambar 5.39, penulis memperoleh *insight* bahwa pengguna di klaster pertama (*gold customers*) didominasi oleh pelanggan berjenis kelamin wanita dengan rentang usia (*age*) antara 36 hingga 100 tahun. Pelanggan di klaster ini memiliki pendapatan (*income*) antara USD 67.000 hingga USD 120.000 per tahun. Kemudian pelanggan di klaster pertama memiliki rata-rata frekuensi pembelian (*frequency*) sebanyak 6 kali dengan total pembelian (*monetary*) antara USD 1 hingga 209. Pelanggan yang berada di klaster ini telah menjadi pengguna Starbucks dengan rentang antara 3,5 hingga 8 tahun. Sehingga dapat dikatakan bahwa beberapa pelanggan di klaster ini merupakan pelanggan loyal karena telah menjadi pelanggan selama lebih dari 6 tahun. Adapun rekomendasi yang dapat diberikan kepada klaster *gold customers* adalah mengirimkan lebih banyak penawaran informasional atau meningkatkan tingkat kesulitan (*difficulty*) dari penawaran *discount* mereka, misalnya mengirimkan *offer_id* 6. Hal ini dikarenakan, klaster *gold customers* cenderung berminat untuk merespons tipe penawaran apa pun yang mereka terima. Dengan begitu, diharapkan frekuensi pembelian dari klaster pertama dan laba perusahaan dapat lebih meningkat.

Tabel 5.40 Informasi statistik untuk klaster *silver customers*

No.	Fungsi Statistik	<i>age</i>	<i>income</i>	<i>frequency</i>	<i>monetary</i>	<i>Tenure</i>
1.	<i>count</i>	3.628	3.628	3.628	3.628	3.628
2.	<i>mean</i>	34,75	51.393,32	7,69	80,53	50,90
3.	<i>std</i>	9,52	13.046,26	4,78	104,95	6,40
4.	<i>min</i>	18	30.000	0	0	41
5.	25%	26	40.000	4	16,80	46
6.	50%	35	51.000	7	40,18	50
7.	75%	43	62.000	11	118,41	53
8.	<i>max</i>	55	84.000	29	1.365,66	74

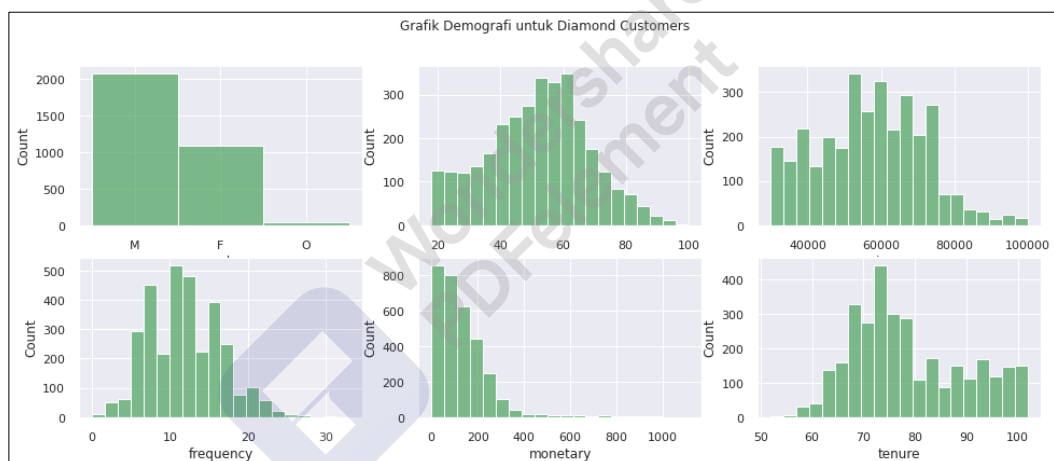


Gambar 5.40 Hasil visualisasi data untuk kluster *silver customers*

Pada Tabel 5.40 dan Gambar 5.40, penulis memperoleh *insight* bahwa pengguna di klaster kedua (*silver customers*) didominasi oleh pelanggan berjenis kelamin pria dengan rentang usia (*age*) antara 18 hingga 55 tahun. Pelanggan di klaster ini memiliki pendapatan (*income*) antara USD 30.000 hingga USD 84.000 per tahun. Kemudian pelanggan di klaster kedua memiliki rata-rata frekuensi pembelian (*frequency*) sebanyak 8 kali dengan total pembelian (*monetary*) antara USD 1 hingga 118. Pelanggan yang berada di klaster ini telah menjadi pengguna Starbucks dengan rentang antara 3,5 hingga 6 tahun. Adapun rekomendasi yang dapat diberikan kepada klaster *silver customers* adalah mengirimkan lebih banyak penawaran dengan tingkat kesulitan (*difficulty*) yang lebih rendah atau tingkat hadiah (*reward*) yang lebih tinggi, misalnya *offer_id* 3 atau *offer_id* 8. Hal ini dikarenakan, klaster *silver customers* cenderung kurang berminat untuk merespons jenis penawaran apapun dengan tingkat kesulitan yang tinggi atau tingkat hadiah yang rendah. Dengan begitu, diharapkan total pembelian dari klaster kedua dan laba perusahaan dapat lebih meningkat.

Tabel 5.41 Informasi statistik untuk klaster *diamond customers*

No.	Fungsi Statistik	<i>age</i>	<i>income</i>	<i>frequency</i>	<i>monetary</i>	<i>Tenure</i>
1.	<i>count</i>	3.221	3.221	3.221	3.221	3.221
2.	<i>mean</i>	52,01	56.974,54	11,86	135	78,51
3.	<i>std</i>	16,19	14.492,29	4,75	119,01	11,1
4.	<i>min</i>	18	30.000	0	0	52
5.	25%	41	46.000	8	53,45	70
6.	50%	53	57.000	12	107,36	76
7.	75%	63	68.000	15	181,43	87
8.	<i>max</i>	100	100.000	33	1.112,66	102

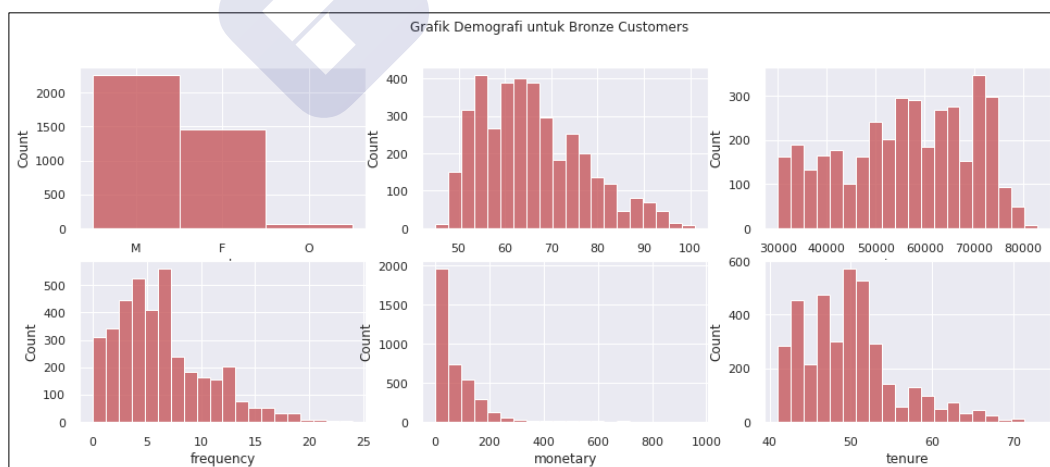
**Gambar 5.41** Hasil visualisasi data untuk klaster *diamond customers*

Pada Tabel 5.41 dan Gambar 5.41, penulis memperoleh *insight* bahwa pengguna di klaster ketiga (*diamond customers*) didominasi oleh pelanggan berjenis kelamin pria dengan rentang usia (*age*) antara 18 hingga 100 tahun. Pelanggan di klaster ini memiliki pendapatan (*income*) antara USD 30.000 hingga USD 100.000 per tahun. Kemudian pelanggan di klaster ketiga memiliki rata-rata frekuensi pembelian (*frequency*) sebanyak 12 kali dengan total pembelian (*monetary*) antara USD 1 hingga 181. Pelanggan yang berada di klaster ini telah menjadi pengguna Starbucks dengan rentang antara 4,5 hingga 8,5 tahun. Sehingga dapat dikatakan bahwa beberapa pelanggan di klaster ini merupakan pelanggan loyal karena telah menjadi pelanggan selama lebih dari 6 tahun.

Adapun rekomendasi yang dapat diberikan kepada klaster *diamond customers* adalah mengirimkan lebih banyak penawaran informasional atau meningkatkan tingkat kesulitan (*difficulty*) dari penawaran *discount* mereka, misalnya mengirimkan *offer_id* 6. Hal ini dikarenakan, klaster *gold customers* cenderung berminat untuk merespons tipe penawaran apa pun yang mereka terima. Dengan begitu, diharapkan frekuensi pembelian dari klaster pertama dan laba perusahaan dapat lebih meningkat.

Tabel 5.42 Informasi statistik untuk klaster *bronze customers*

No.	Fungsi Statistik	<i>age</i>	<i>Income</i>	<i>frequency</i>	<i>monetary</i>	<i>Tenure</i>
1.	<i>count</i>	3785	3785	3785	3785	3785
2.	<i>mean</i>	65.73	56143.19	6.18	74.47	49.69
3.	<i>std</i>	11.06	13329.30	4.16	89.63	5.71
4.	<i>min</i>	45	30000	0	0	41
5.	<i>25%</i>	57	46000	3	16.82	45
6.	<i>50%</i>	64	57000	5	44.33	49
7.	<i>75%</i>	73	67000	9	105.44	53
8.	<i>max</i>	101	83000	24	958.19	73



Gambar 5.42 Hasil visualisasi data untuk klaster *bronze customers*

Pada Tabel 5.42 dan Gambar 5.42, penulis memperoleh *insight* bahwa pengguna di klaster keempat (*bronze customers*) didominasi oleh pelanggan berjenis kelamin pria dengan rentang usia (*age*) antara 45 hingga 101 tahun.

Pelanggan di klaster ini memiliki pendapatan (*income*) antara USD 30.000 hingga USD 83.000 per tahun. Kemudian pelanggan di klaster kedua memiliki rata-rata frekuensi pembelian (*frequency*) sebanyak 6 kali dengan total pembelian (*monetary*) antara USD 1 hingga 105. Pelanggan yang berada di klaster ini telah menjadi pengguna Starbucks dengan rentang antara 3,5 hingga 6 tahun. Adapun rekomendasi yang dapat diberikan kepada klaster *bronze customers* adalah mengirimkan lebih banyak penawaran bertipe BOGO, misalnya *offer_id* 7 atau *offer_id* 8. Hal ini dikarenakan, klaster *bronze customers* cenderung kurang berminat untuk merespons tipe penawaran selain BOGO. Dengan begitu, diharapkan frekuensi serta total pembelian dari klaster keempat dan laba perusahaan dapat lebih meningkat.

Setelah memperoleh informasi demografis dari tiap klaster pengguna Starbucks, langkah terakhir yang dilakukan penulis ialah menarik beberapa kesimpulan atau *insight* terkait hasil *clustering* pada data pengguna Starbucks. Adapun beberapa *insight* tersebut sebagai berikut. Pengguna berjenis kelamin wanita pada klaster *gold customers* cenderung menghabiskan lebih banyak uang daripada pengguna berjenis kelamin pria. Selain itu, pada klaster ini, ketika pendapatan pengguna meningkat, tingkat pembelian juga ikut meningkat. Pengguna berusia muda pada klaster *diamond customers* sering melakukan pembelian namun dalam nominal yang kecil. Pada klaster ini, semakin lama usia pendaftaran pengguna di aplikasi Starbucks, maka semakin sering pengguna melakukan transaksi. Berlaku pada semua klaster, seiring bertambahnya usia dan pendapatan pengguna, total pembelian juga meningkat hingga usia paruh baya. Setelah mencapai usia maksimal paruh baya (sekitar 60 tahun), aktivitas pembelian pelanggan cenderung bergerak melandai.

BAB VI PENUTUP

6.1 Simpulan

Dari hasil penelitian, setelah melakukan segmentasi data pelanggan Starbucks berdasarkan pembobotan model FMT menggunakan algoritma *K-Means* dengan *Principal Component Analysis* (PCA), lantas penulis mengambil beberapa simpulan antara lain:

- a. Algoritma *K-Means* dengan reduksi dimensi *Principal Component Analysis* (PCA) dapat diimplementasikan untuk mengelompokkan data pelanggan Starbucks berdasarkan pembobotan model FMT ke dalam jumlah klaster tertentu. Sedikit berbeda dengan model RFM (*Recency, Frequency, Monetary*), model yang digunakan dalam penelitian ini adalah model FMT (*Frequency, Monetary, Tenure*). Model FMT dipilih karena tidak adanya informasi tanggal transaksi (*Recency*) pada *dataset* dalam penelitian ini.
- b. Model FMT dilakukan normalisasi data dengan tujuan untuk mengatasi ketidaksimetrisan distribusi data. Metode normalisasi data yang digunakan adalah *Min-Max Normalization* yang menerapkan transformasi linier.
- c. Metode PCA yang digunakan dalam penelitian ini berhasil mereduksi dimensi dengan mengkonversi lima variabel demografi pengguna menjadi hanya tiga komponen. Pemilihan jumlah komponen sebanyak tiga juga merupakan pilihan yang tepat karena telah mewakili sebesar 89% dari keseluruhan informasi yang ada dalam model FMT.
- d. Setelah dilakukan proses pemilihan jumlah klaster optimal diperoleh bahwa empat klaster merupakan yang terbaik. Kemudian penulis melakukan proses *string mapping* untuk penamaan masing-masing klaster meliputi: klaster pertama (*gold customers*), klaster kedua (*silver customers*), klaster ketiga (*diamond customers*), dan klaster keempat (*bronze customers*).

- e. Metode uji performa yang digunakan ialah metode *Elbow* dengan perhitungan *Sum of Squared Errors* (SSE). Perhitungan SSE yang dipilih menggunakan jenis evaluasi bernama *Within-Cluster Sum of Squares* (WCSS). Metode *Elbow* diterapkan untuk menentukan jumlah kluster k yang optimal dan diperoleh $k = 4$, sehingga *K-Means clustering* diimplementasikan dengan empat segmen pelanggan. Perhitungan nilai WCSS dilakukan untuk memvalidasi apakah empat adalah jumlah kluster yang optimal. Kemudian diperoleh hasil perhitungan nilai WCSS yaitu jumlah kluster $k = 4$ dengan nilai sebesar 979,80. Nilai tersebut merupakan nilai yang mengalami penurunan drastis dan membentuk sudut siku pada visualisasi grafik berupa *elbow plot*.

6.2 Saran

Peneliti menyadari bahwa penelitian ini belum sempurna. Maka dari itu, peneliti menyarankan perbaikan untuk penelitian selanjutnya berupa:

- Menggunakan *dataset* pelanggan dengan atribut dan nilai yang lebih lengkap (contoh: model RFMT) sehingga diperoleh *business insight* yang lebih akurat untuk keperluan strategi pemasaran perusahaan.
- Menggunakan algoritma pengelompokkan data lainnya seperti *DBSCAN*, *EM Clustering*, dan *Agglomerative Hierarchical Clustering*. Selain itu, metode uji performa lainnya seperti *Silhouette Coefficient*, *Dunn's Index*, dan *Davis Bouldin Index* sehingga diperoleh lebih banyak pengembangan baru terkait analisa data melalui proses *clustering*.
- Mengimplementasikan hasil analisis model FMT dan algoritma *clustering* ke dalam aplikasi atau situs web sehingga pengaksesan dan pengujian data menjadi lebih mudah dilakukan.
- Mengajukan proposal penelitian secara langsung ke pihak perusahaan sehingga peneliti bisa memperoleh lebih banyak masukan dan *insight* agar sistem yang dihasilkan bisa sesuai dengan yang diharapkan.

DAFTAR PUSTAKA

- Adiana, B.E., Soesanti, I. dan Permanasari, A.E. (2018), *Analisis Segmentasi Pelanggan Menggunakan Kombinasi RFM Model Dan Teknik Clustering*, Jurnal Terapan Teknologi Informasi, 2(1), 23–32.
- Ait-Sahalia, Y. dan Xiu, D. (2017), *Using Principal Component Analysis To Estimate A High Dimensional Factor Model With High-Frequency Data*, Journal of Econometrics, 201(2), 384–399.
- Akhyar, S. (2017), *Pengelompokan Kabupaten/Kota Di Jawa Timur Berdasarkan Indikator Pembangunan Ekonomi Menggunakan Model-Based Clustering*, Skripsi, S.Kom., Institut Teknologi Sepuluh Nopember, Surabaya.
- Angelie, A.V. (2017), *Segmentasi Pelanggan Menggunakan Clustering K-Means Dan Model RFM (Studi Kasus: PT. Bina Adidaya Surabaya)*, Skripsi, S.Kom., Institut Teknologi Sepuluh Nopember, Surabaya.
- Badan Pusat Statistik, I. (2018), *Statistik Kopi Indonesia: Indonesian Coffee Statistics 2018*, Jakarta: Badan Pusat Statistik.
- Berry, M. dan Linoff, G. (2000), *Mastering Data Mining: The Art Of Science Of Customer Relationship Management*, New York: John Wiley and Sons.
- Bengfort, dkk, (2019), *Yellowbrick: Visualizing the Scikit-Learn Model Selection Process*, Journal of Open Source Software, 4(35), 1075.
- Bi, M., Xu, J., Wang, M. dan Zhou, F. (2016), *Anomaly Detection Model Of User Behavior Based On Principal Component Analysis*, Journal of Ambient Intelligence and Humanized Computing, 7(4), 547–554.
- Budiman, A. dan Tjahjadi, B. (2019), *Evaluasi Strategi Bersaing Differensiasi PT Terminal Teluk Lamong Pada Pelayanan Bongkar Muat Petikemas: Studi Banding Antar Terminal Petikemas Di Lingkungan Pelabuhan Tanjung Perak*, Business and Finance Journal, 4(2), 81–92.
- Chiu, C.Y., Chen, Y.F., Kuo, I.T. dan Ku, H.C. (2009), *An Intelligent Market Segmentation System Using K-Means And Particle Swarm Optimization*, Expert Systems with Applications, 36(3), 4558–4565.
- Christy, A.J., Umamakeswari, A., Priyatharsini, L. dan Neyaa, A. (2021), *RFM Ranking - An Effective Approach To Customer Segmentation*, Journal of King Saud University-Computer and Information Sciences, 33(10), 1251–1257.
- Dash, B., Mishra, D., Rath, A. dan Acharya, M. (2010), *A Hybridized K-Means Clustering Approach For High Dimensional Dataset*, International Journal of Engineering, Science and Technology, 2(2), 59–66.
- De Mast, J. dan Kemper, B.P.H. (2009), *Discussion Of “Principles Of Exploratory Data Analysis In Problem Solving: What Can We Learn From A Well-Known Case?” - Rejoinder*, Quality Engineering, 21(4), 382–383.

- Fildzah, A.N. dan Mayangsari, I.D. (2018), *Analisis Strategi Promosi Pada UMKM Social Enterprise (Studi Kasus Pascorner Cafe And Gallery)*, Jurnal Komunikasi, 12(2), 101–112.
- Fraley, C. dan Raftery, A.E. (1998), *How Many Clusters? Which Clustering Method? Answers Via Model-Based Cluster Analysis*, The Computer Journal, 41(8), 578–588.
- Ganesh, V., Zhang, L., Wan, B.A., Drost, L., Tsao, M., Barnes, E., dan Chow, E. (2018), *Symptom Clusters Using The EORTC QLQ-C15-PAL In Palliative Radiotherapy*, Ann Palliat Med, 7(2), 192–204.
- Hadi, F., Mustakim, M., Rahmadia, D.O., Nugraha, F.H., Bulan, N.P. dan Monalisa, S. (2017), *Penerapan K-Means Clustering Berdasarkan RFM Mofek Sebagai Pemetaan Dan Pendukung Strategi Pengelolaan Pelanggan (Studi Kasus: PT. Herbal Penawar Alwahidah Indonesia Pekanbaru)*, Jurnal Sains Dan Teknologi Industri, 15(1), 69–76.
- Hanifa, T.T., Al-faraby, S. (2017), *Analisis Churn Prediction Pada Data Pelanggan PT. Telekomunikasi Dengan Logistic Regression Dan Underbagging*, Vol 4(2), 3210–3225.
- Harani, N.H., Prianto, C. dan Nugraha, F.A. (2020), *Segmentasi Pelanggan Produk Digital Service Indihome Menggunakan Algoritma K-Means Berbasis Python*, Jurnal Manajemen Informatika (JAMIKA), 10(2) 133-146.
- Hardiansyah, B. dan Primandari, P.N. (2018), *Sistem Pakar Pengenalan Ekspresi Wajah Manusia Menggunakan Metode Kohonen Self Organizing Dan Principal Componen Analysis*, INTEGER: Journal of Information Technology, 3(2), 43-54.
- Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., ..., Oliphant, T. E. (2020). *Array programming with NumPy*, Nature, 585, 357–362.
- Heckert, N.A., Filliben, J.J., Croarkin, C.M., Hembree, B., Guthrie, W.F., dan Tobias, P. (2002), *Handbook 151: NIST/SEMATECH E-Handbook Of Statistical Methods*, Gaithersburg: National Institute of Standards and Technology.
- Hughes, A.M. (2000), *Strategic Database Marketing: The Masterplan For Starting And Managing A Profitable Customer-Based Marketing Program*, New York: McGraw-Hill.
- Hunter, J.D. (2007), *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, Vol. 9(3), 90-95.
- Izzuddin, A. (2015), *Optimasi Cluster Pada Algoritma K-Means Dengan Reduksi Dimensi Dataset Menggunakan Principal Component Analysis Untuk Pemetaan Kinerja Dosen*, Energy - Jurnal Ilmiah Ilmu-Ilmu Teknik, 5(2), 41–46.

- Jansen, S. (2007), *Customer Segmentation And Customer Profiling For A Mobile Telecommunications Company Based On Usage Behavior*, Disertasi, Ph.D, Maastricht University, Maastricht.
- Jiawei Han, M.K. and J.P. (2012), *Data Mining: Concepts And Techniques - Third Edition*, Burlington: Morgan Kaufmann.
- Kurniawati, I.Y. (2018), *Segmentasi Pelanggan Menggunakan Clustering K-Means*, Skripsi, S.Kom., Universitas 17 Agustus 1945, Surabaya.
- Mahendra, I.W.E. dan Parmithi, N.N. (2015), *Statistik Dasar Dalam Penelitian Pendidikan*, Surabaya: Paramita.
- Maimon, O. dan Rokach, L. (2005), *Data Mining And Knowledge Discovery Handbook*, New York: Springer.
- Martinez, W.L., Martinez, A.R. dan Solka, J.L. (2017), *Exploratory Data Analysis With MATLAB*, Boca Raton: Chapman and Hall/CRC.
- McKinney, W. (2010). *Data structures for statistical computing in python*, In Proceedings of the 9th Python in Science Conference, Vol. 445, 51–56.
- Merliana, N.P.E., Ernawati, dan Santoso, A.J. (2015), *Analisa Penentuan Jumlah Cluster Terbaik Pada Metode K-Means Clustering*, Prosiding Seminar Nasional Multi Disiplin Ilmu dan Call For Papers Unisbank.
- Miglautsch, J.R. (2000), *Thoughts On RFM Scoring*, Journal of Database Marketing & Customer Strategy Management, 8(1), 67–72.
- Moreau, T., Olivier Grisel, Pierre Glaser, Roman Yurchak, Albert Thomas, Alexandre Abadie, Bas Nijholt, Christopher J. CJ Wright, Joan Massich, Jérémie du Boisberranger, Lukasz Migas, Michał Górny, & Christos Aridas. (2021). *joblib/loky: 3.0.0*. Zenodo, <https://doi.org/10.5281/zenodo.5770407>.
- Pedregosa, F., Varoquaux, Gael, Gramfort, A., Michel, V., Thirion, B., Grisel, O., dkk. (2011). *Scikit-learn: Machine learning in Python*, Journal of Machine Learning Research, Vol. 12, 2825–2830.
- Plotly Technologies Inc (2015), *Collaborative data science*, Montreal: Plotly Technologies Inc.
- Python Core Team (2015), *Python: A dynamic, open source programming language*, Beaverton: Python Software Foundation.
- Radhi, M., Amalia, A., Sitompul, D.R.H., Sinurat, S.H., dan Indra, E. (2021), *Analisis Big Data Dengan Metode Exploratory Data Analysis (EDA) Dan Metode Visualisasi Menggunakan Jupyter Notebook*, Jurnal Sistem Informasi Dan Ilmu Komputer Prima (JUSIKOM PRIMA), 4(2), 23–27.
- Sartono, B. (2016), *Analisis Gerombol: Penggerombolan Tak Berhirarki - Algoritma K-Means*, (<http://bagusco.staff.ipb.ac.id/files/2016/06/Analisis-Cluster-Bagian-2.pdf>), diakses 2 Juni 2022.
- Savitri, A.D. (2018), *Segmentasi Pelanggan Menggunakan Metode K-Means Clustering Berdasarkan Model RFM Pada Klinik Kecantikan (Studi Kasus: Belle Crown Malang)*, Skripsi, S.Kom., Universitas Brawijaya, Malang.

- Tan, Pang-Ning, Steinbach, M., Adeyeye Oshin, M., Kumar, V. dan Vipin (2005), *Introduction To Data Mining*, London: Pearson.
- Tsiptsis, K. dan Chorianopoulos, A. (2010), *Data Mining Techniques In CRM: Inside Customer Segmentation*, New York: John Wiley and Sons.
- Wakhidah, N. (2010), *Clustering Menggunakan K-Means Algorithm*, Jurnal Transformatika, 8(1), 33–39.
- Waskom, M.L. (2021), *Seaborn: statistical data visualization*, Journal of Open Source Software, Vol. 6(60), 3021.
- Widiowati, M.K. (2014), *Deteksi Outlier Pada Data Campuran Numerik Dan Kategorikal Menggunakan Algoritma Enhanced Class Outlier Distance Based (ECODB) (Studi Kasus: Data Kredit BPR XYZ)*, Skripsi, S.Kom., Universitas Sanata Dharma, Yogyakarta.
- Wu, J. dan Lin, Z. (2005), *Research On Customer Segmentation Model By Clustering*, New York: ACM International.
- Zhou, J., Wei, J. dan Xu, B. (2021), *Customer Segmentation By Web Content Mining*, Journal of Retailing and Consumer Services, 61(1), 102-588.

LAMPIRAN

Lampiran 1 Data-data yang diperoleh

Dataset portfolio.json

No.	<i>reward</i>	<i>channels</i>	<i>Difficulty</i>	<i>duration</i>	<i>offer_type</i>	<i>id</i>
1.	10	['email', 'mobile', 'social']	10	7	bogo	ae264e363720 4a6fb9bb56bc 8210ddfd
2.	10	['web', 'email', 'mobile', 'social']	10	5	bogo	4d5c57ea9a69 40dd891ad53e 9dbe8da0
3.	0	['web', 'email', 'mobile']	0	4	informatio nal	3f207df678b1 43eea3cee631 60fa8bed
4.	5	['web', 'email', 'mobile']	5	7	bogo	9b98b8c7a33c 4b65b9aebfe6 a799e6d9
5.	5	['web', 'email']	20	10	discount	0b1e1539f2cc 45b7b9fa7c27 2da2e1d7
6.	3	['web', 'email', 'mobile', 'social']	7	7	discount	2298d6c36e96 4ae4a3e7e970 6d1fb8c2
7.	2	['web', 'email', 'mobile', 'social']	10	10	discount	fafdc668e37 43c1bb461111 dcafc2a4

8.	0	['email', 'mobile', 'social']	0	3	informatio nal	5a8bc65990b2 45e5a138643c d4eb9837
9.	5	['web', 'email', 'mobile', 'social']	5	5	bogo	f19421c1d4aa 40978ebb69ca 19b0e20d
10.	2	['web', 'email', 'mobile']	10	7	discount	2906b810c7d4 411798c6938a dc9daaa5

Dataset profile.json

No.	gender	age	Id	become_member_on	income
1.		118	68be06ca386d4c3 1939f3a4f0e3dd7 83	20170212	
2.	F	55	0610b486422d49 21ae7d2bf64640c 50b	20170715	112000
3.		118	38fe809add3b4fcf 9315a9694bb96ff 5	20180712	
⋮	⋮	⋮	⋮	⋮	⋮
16999.	F	83	9dc1421481194d cd9400aec7c9ae6 366	20160307	50000
17000.	F	62	e4052622e5ba45a 8b96b59aba68cf0 68	20170722	82000

Dataset transcript.json

No.	person	event	value	time
1.	78afa995795e4d85 b5d9ceeca43f5fef	offer received	{'offer id': '9b98b8c7a33c4b65b9aebf e6a799e6d9'}	0
2.	a03223e636434f42 ac4c3df47e8bac43	offer received	{'offer id': '0b1e1539f2cc45b7b9fa7c 272da2e1d7'}	0
3.	e2127556f4f64592 b11af22de27a7932	offer received	{'offer id': '2906b810c7d4411798c69 38adc9daaa5'}	0
4.	8ec6ce2a7e7949b1 bf142def7d0e0586	offer received	{'offer id': 'fafdcd668e3743c1bb4611 11dcafc2a4'}	0
5.	68617ca6246f4fbc 85e91a2a49552598	offer received	{'offer id': '4d5c57ea9a6940dd891ad5 3e9dbe8da0'}	0
⋮	⋮	⋮	⋮	⋮
306530.	b3a1272bc9904337 b331bf348c3e8c17	transacti on	{'amount': 1.5899999999999999}	714
306531.	68213b08d99a4ae1 b0dcb72aebd9aa35	transacti on	{'amount': 9.53}	714
306532.	a00058cf10334a30 8c68e7631c529907	transacti on	{'amount': 3.61}	714
306533.	76ddbd6576844afe 811f1a3c0fbb5bec	transacti on	{'amount': 3.5300000000000002}	714
306534.	c02b10e8752c4d8e 9b73f918558531f7	transacti on	{'amount': 4.05}	714

Lampiran 2 Hasil tahap normalisasi data

No.	<i>age_norm</i>	<i>inc_norm</i>	<i>f_norm</i>	<i>m_norm</i>	<i>t_norm</i>
1.	0.44578313 25301205	0.91111111 11111111	0.0909090 90909090 91	0.04787124 9277362325	0.213114754 09836067
2.	0.68674698 79518072	0.77777777 77777778	0.2121212 12121212 13	0.09900602 353467727	0.245901639 34426224
3.	0.60240963 85542169	0.44444444 44444444	0.0909090 90909090 91	0.03588634 230336485	0.049180327 8688524
4.	0.56626506 02409639	0.25555555 555555554	0.0909090 90909090 91	0.02264575 524184274	0.098360655 73770492
5.	0.48192771 08433735	0.23333333 333333328	0.0909090 90909090 91	0.00970976 3845116211	0.147540983 60655732
6.	0.51807228 91566265	0.30000000 000000004	0.1818181 81818181 82	0.05317991 657808527	0.180327868 852459
7.	0.09638554 21686747	0.17777777 77777778	0.2727272 72727272 7	0.03494769 03567499	0.885245901 6393444
⋮	⋮	⋮	⋮	⋮	⋮
14823.	0.37349397 590361444	0.47777777 777777775	0.2424242 42424242 43	0.02470333 0038727163	0.311475409 8360656
14824.	0.78313253 01204819	0.22222222 22222222	0.4242424 24242424 25	0.11790338 722811729	0.475409836 06557385
14825.	0.53012048 19277109	0.57777777 77777778	0.1818181 81818181 82	0.08890463 669196676	0.213114754 09836067

Lampiran 3 Hasil tahap *Principal Component Analysis* (PCA)

No.	<i>comp1</i>	<i>comp2</i>	<i>comp3</i>
1.	0.4558028281494 0454	-0.09149197579411143	-0.2709808145861505
2.	0.4573303978389 0367	- 0.015918994835143762	0.00306787130668016 46
3.	0.1552198721881 1303	-0.2708550131433704	0.10840057663281046
4.	- 0.0226801282656 9341	-0.23954986442860302	0.17948157309531984
5.	- 0.0880509403427 9108	-0.19888432673246373	0.12127268460846914
6.	- 0.0253061544011 02097	-0.12751246117412468	0.11588430084416959
7.	- 0.3829238641667 849	0.5384026751042338	-0.16576375710240399
⋮	⋮	⋮	⋮
14820.	0.4932073290252 6876	-0.21854291033139156	-0.22785004442485243
14821.	- 0.1597884197820 8014	-0.26624403973364397	- 0.02775964401312410 2
14822.	0.1088393233524 6572	-0.26272809334801883	0.02846463482697201
14823.	0.0302198510026 31394	0.019891383536779172	-0.09662984481455165
14824.	0.0106652844246 86066	0.23957223214880946	0.3848360377899604
14825.	0.2133298393628 6848	-0.07523394853527743	-0.02329798172532049

Lampiran 4 Hasil tahap *K-Means clustering*

<i>User ID</i>	<i>frequency</i>	<i>monetary</i>	<i>tenure</i>	<i>Cluster Name</i>
1	3	77.01	54	<i>Gold Customers</i>
2	7	159.27	56	<i>Gold Customers</i>
3	3	57.73	44	<i>Bronze Customers</i>
4	3	36.43	47	<i>Bronze Customers</i>
5	3	15.62	50	<i>Bronze Customers</i>
6	6	85.55	52	<i>Bronze Customers</i>
7	9	56.22	95	<i>Diamond Customers</i>
8	8	160.25	71	<i>Diamond Customers</i>
9	7	144.10	86	<i>Diamond Customers</i>
10	5	19.1	48	<i>Bronze Customers</i>
11	5	20.78	44	<i>Bronze Customers</i>
12	16	300.68	68	<i>Diamond Customers</i>
13	7	211.29	52	<i>Gold Customers</i>
14	8	155.46	55	<i>Bronze Customers</i>
15	5	91.4	102	<i>Diamond Customers</i>
16	3	75.62	44	<i>Gold Customers</i>
17	5	17.9	51	<i>Silver Customers</i>
18	1	2.88	52	<i>Silver Customers</i>
19	9	52.08	84	<i>Diamond Customers</i>
20	8	206.88	89	<i>Gold Customers</i>
⋮	⋮	⋮	⋮	⋮
14815	4	75.16	53	<i>Bronze Customers</i>
14816	10	135.62	66	<i>Diamond Customers</i>
14817	8	32.44	48	<i>Bronze Customers</i>
14818	15	143.75	70	<i>Diamond Customers</i>
14819	5	17.86	52	<i>Silver Customers</i>
14820	5	115.59	44	<i>Gold Customers</i>
14821	7	20.03	43	<i>Silver Customers</i>
14822	7	25.97	42	<i>Bronze Customers</i>
14823	8	39.74	60	<i>Gold Customers</i>
14824	14	189.67	70	<i>Diamond Customers</i>
14825	6	143.02	54	<i>Gold Customers</i>

Lampiran 5 Struktur *dataset* dalam format JSON*Dataset portfolio.json*

```
[
  {
    "reward": 10,
    "channels": [
      "email",
      "mobile",
      "social"
    ],
    "difficulty": 10,
    "duration": 7,
    "offer_type": "bogo",
    "id": "ae264e3637204a6fb9bb56bc8210ddfd"
  },
  {
    "reward": 10,
    "channels": [
      "web",
      "email",
      "mobile",
      "social"
    ],
    "difficulty": 10,
    "duration": 5,
    "offer_type": "bogo",
    "id": "4d5c57ea9a6940dd891ad53e9dbe8da0"
  },
  {
    "reward": 0,
    "channels": [
      "web",
      "email",
      "mobile"
    ],
    "difficulty": 0,
    "duration": 4,
    "offer_type": "informational",
    "id": "3f207df678b143eea3cee63160fa8bed"
  },
  {
    "reward": 5,
    "channels": [
      "web",
```



```
"email",
"mobile"
],
"difficulty": 5,
"duration": 7,
"offer_type": "bogo",
"id": "9b98b8c7a33c4b65b9aebfe6a799e6d9"
},
{
  "reward": 5,
  "channels": [
    "web",
    "email"
  ],
  "difficulty": 20,
  "duration": 10,
  "offer_type": "discount",
  "id": "0b1e1539f2cc45b7b9fa7c272da2e1d7"
},
{
  "reward": 3,
  "channels": [
    "web",
    "email",
    "mobile",
    "social"
  ],
  "difficulty": 7,
  "duration": 7,
  "offer_type": "discount",
  "id": "2298d6c36e964ae4a3e7e9706d1fb8c2"
},
{
  "reward": 2,
  "channels": [
    "web",
    "email",
    "mobile",
    "social"
  ],
  "difficulty": 10,
  "duration": 10,
  "offer_type": "discount",
  "id": "fafdcd668e3743c1bb461111dcafc2a4"
},
{
```



```
"reward": 0,
"channels": [
  "email",
  "mobile",
  "social"
],
"difficulty": 0,
"duration": 3,
"offer_type": "informational",
"id": "5a8bc65990b245e5a138643cd4eb9837"
},
{
  "reward": 5,
  "channels": [
    "web",
    "email",
    "mobile",
    "social"
  ],
  "difficulty": 5,
  "duration": 5,
  "offer_type": "bogo",
  "id": "f19421c1d4aa40978ebb69ca19b0e20d"
},
{
  "reward": 2,
  "channels": [
    "web",
    "email",
    "mobile"
  ],
  "difficulty": 10,
  "duration": 7,
  "offer_type": "discount",
  "id": "2906b810c7d4411798c6938adc9daaa5"
}
]
```

*Dataset profile.json*

```
[
  {
    "gender": null,
    "age": 118,
    "id": "68be06ca386d4c31939f3a4f0e3dd783",
    "became_member_on": "20170212",
    "income": null
  },
  {
    "gender": "F",
    "age": 55,
    "id": "0610b486422d4921ae7d2bf64640c50b",
    "became_member_on": "20170715",
    "income": 112000
  },
  {
    "gender": null,
    "age": 118,
    "id": "38fe809add3b4fcf9315a9694bb96ff5",
    "became_member_on": "20180712",
    "income": null
  },
  {
    "gender": "F",
    "age": 75,
    "id": "78afa995795e4d85b5d9ceeca43f5fef",
    "became_member_on": "20170509",
    "income": 100000
  },
  {
    "gender": null,
    "age": 118,
    "id": "a03223e636434f42ac4c3df47e8bac43",
    "became_member_on": "20170804",
    "income": null
  },
  {
    "gender": "M",
    "age": 68,
    "id": "e2127556f4f64592b11af22de27a7932",
    "became_member_on": "20180426",
    "income": 70000
  },
  {
    "gender": null,
```

```
"age": 118,
"id": "8ec6ce2a7e7949b1bf142def7d0e0586",
"became_member_on": "20170925",
"income": null
},
{
  "gender": null,
  "age": 118,
  "id": "68617ca6246f4fbc85e91a2a49552598",
  "became_member_on": "20171002",
  "income": null
},
{
  "gender": "M",
  "age": 65,
  "id": "389bc3fa690240e798340f5a15918d5c",
  "became_member_on": "20180209",
  "income": 53000
},
{
  "gender": null,
  "age": 118,
  "id": "8974fc5686fe429db53ddde067b88302",
  "became_member_on": "20161122",
  "income": null
},
{
  "gender": null,
  "age": 118,
  "id": "c4863c7985cf408faee930f111475da3",
  "became_member_on": "20170824",
  "income": null
},
{
  "gender": null,
  "age": 118,
  "id": "148adfcaa27d485b82f323aaaad036bd",
  "became_member_on": "20150919",
  "income": null
},
{
  "gender": "M",
  "age": 58,
  "id": "2eeac8d8feae4a8cad5a6af0499a211d",
  "became_member_on": "20171111",
  "income": 51000
```

```
},
{
  "gender": "F",
  "age": 61,
  "id": "aa4862eba776480b8bb9c68455b8c2e1",
  "became_member_on": "20170911",
  "income": 57000
},
{
  "gender": "M",
  "age": 26,
  "id": "e12aeaf2d47d42479ea1c4ac3d8286c6",
  "became_member_on": "20140213",
  "income": 46000
},
{
  "gender": "F",
  "age": 62,
  "id": "31dda685af34476cad5bc968bdb01c53",
  "became_member_on": "20160211",
  "income": 71000
},
{
  "gender": "M",
  "age": 49,
  "id": "62cf5e10845442329191fc246e7bcea3",
  "became_member_on": "20141113",
  "income": 52000
},
{
  "gender": null,
  "age": 118,
  "id": "744d603ef08c4f33af5a61c8c7628d1c",
  "became_member_on": "20170801",
  "income": null
},
{
  "gender": "M",
  "age": 57,
  "id": "6445de3b47274c759400cd68131d91b4",
  "became_member_on": "20171231",
  "income": 42000
},
{
  "gender": "F",
  "age": 61,
```

```
"id": "a448667f336b42c9a66fc5ffd5d73772",
"became_member_on": "20180501",
"income": 40000
},
{
  "gender": "F",
  "age": 40,
  "id": "440cf1fd7580490c971d8c651ed962af",
  "became_member_on": "20160504",
  "income": 71000
},
:
:
:
{
  "gender": "M",
  "age": 61,
  "id": "2cb4f97358b841b9a9773a7aa05a9d77",
  "became_member_on": "20180713",
  "income": 72000
},
{
  "gender": "M",
  "age": 49,
  "id": "01d26f638c274aa0b965d24cefe3183f",
  "became_member_on": "20170126",
  "income": 73000
},
{
  "gender": "F",
  "age": 83,
  "id": "9dc1421481194dcd9400aec7c9ae6366",
  "became_member_on": "20160307",
  "income": 50000
},
{
  "gender": "F",
  "age": 62,
  "id": "e4052622e5ba45a8b96b59aba68cf068",
  "became_member_on": "20170722",
  "income": 82000
}
]
```

Dataset Transcript.json

```
[
  {
    "person": "78afa995795e4d85b5d9ceeca43f5fef",
    "event": "offer received",
    "value": {
      "offer id": "9b98b8c7a33c4b65b9aebfe6a799e6d9"
    },
    "time": 0
  },
  {
    "person": "a03223e636434f42ac4c3df47e8bac43",
    "event": "offer received",
    "value": {
      "offer id": "0b1e1539f2cc45b7b9fa7c272da2e1d7"
    },
    "time": 0
  },
  {
    "person": "e2127556f4f64592b11af22de27a7932",
    "event": "offer received",
    "value": {
      "offer id": "2906b810c7d4411798c6938adc9daaa5"
    },
    "time": 0
  },
  {
    "person": "8ec6ce2a7e7949b1bf142def7d0e0586",
    "event": "offer received",
    "value": {
      "offer id": "fafdcd668e3743c1bb461111dcafc2a4"
    },
    "time": 0
  },
  {
    "person": "68617ca6246f4fbc85e91a2a49552598",
    "event": "offer received",
    "value": {
      "offer id": "4d5c57ea9a6940dd891ad53e9dbe8da0"
    },
    "time": 0
  },
  {
    "person": "389bc3fa690240e798340f5a15918d5c",
    "event": "offer received",
    "value": {
```

```
"offer id": "f19421c1d4aa40978ebb69ca19b0e20d"
},
"time": 0
},
{
  "person": "c4863c7985cf408faee930f111475da3",
  "event": "offer received",
  "value": {
    "offer id": "2298d6c36e964ae4a3e7e9706d1fb8c2"
  },
  "time": 0
},
{
  "person": "2eeac8d8feae4a8cad5a6af0499a211d",
  "event": "offer received",
  "value": {
    "offer id": "3f207df678b143eea3cee63160fa8bed"
  },
  "time": 0
},
{
  "person": "aa4862eba776480b8bb9c68455b8c2e1",
  "event": "offer received",
  "value": {
    "offer id": "0b1e1539f2cc45b7b9fa7c272da2e1d7"
  },
  "time": 0
},
{
  "person": "5ca2620962114246ab218fc648eb3934",
  "event": "transaction",
  "value": {
    "amount": 2.2
  },
  "time": 714
},
:
:
:
{
  "person": "b3a1272bc9904337b331bf348c3e8c17",
  "event": "transaction",
  "value": {
    "amount": 1.59
  },
  "time": 714
},
```



```
{
  "person": "68213b08d99a4ae1b0dcb72aebd9aa35",
  "event": "transaction",
  "value": {
    "amount": 9.53
  },
  "time": 714
},
{
  "person": "a00058cf10334a308c68e7631c529907",
  "event": "transaction",
  "value": {
    "amount": 3.61
  },
  "time": 714
},
{
  "person": "76ddbd6576844afe811f1a3c0fbb5bec",
  "event": "transaction",
  "value": {
    "amount": 3.53
  },
  "time": 714
},
{
  "person": "c02b10e8752c4d8e9b73f918558531f7",
  "event": "transaction",
  "value": {
    "amount": 4.05
  },
  "time": 714
}
]
```


Lampiran 6 Source code sistem segmentasi pelanggan Starbucks

```

1 import json
2 import numpy as np
3 import pandas as pd
4 import plotly.graph_objs as go
5 import plotly.subplots as subplots
6 import matplotlib.pyplot as plt
7 import seaborn as sns
8 pd.set_option('mode.chained_assignment', None)
9 %matplotlib inline
10 sns.set()
11
12 portfolio = pd.read_json('https://raw.githubusercontent.com/alfianhid/
13 Starbucks-EDA-and-Customer-Segmentation-with-K-means-Algorithm/master/
14 data/portfolio.json', orient='records', lines=True)
15 profile = pd.read_json('https://raw.githubusercontent.com/alfianhid/
16 Starbucks-EDA-and-Customer-Segmentation-with-K-means-Algorithm/master/
17 data/profile.json', orient='records', lines=True)
18 transcript = pd.read_json('https://raw.githubusercontent.com/alfianhid/
19 Starbucks-EDA-and-Customer-Segmentation-with-K-means-Algorithm/master/
20 data/transcript.json', orient='records', lines=True)
21
22 print(portfolio.shape)
23 print(profile.shape)
24 print(transcript.shape)
25
26 print(portfolio.info())
27 portfolio
28 print(profile.info())
29 profile.head()
30 print(transcript.info())
31 transcript.head()
32
33 # fungsi untuk mengetahui data yang kosong
34 def show_missing_values(df, encoded_col=None, encoded_val=None):
35     """
36     Show the number and proportion of missing values in every column of a dataframe.
37     Args:
38     (1) df (Pandas dataframe) - data to inspect
39     (2) encoded_col (str) - name of column with missing values encoded *
40     (3) encoded_val (str or int or float) - value that was
41         used to encoded missing values *
42     * Pass in args (2) and (3) if a column had its missing values encoded
43     Returns:
44     Dataframe showing number and proportion of missing values
45     in each column (Pandas dataframe).
46     """
47     missing = df.isnull().sum().reset_index() # jumlah data yang kosong
48     missing.columns = ['column', 'n_missing']
49     if encoded_col: # jika data yang kosong adalah kolom yang telah dikodekan
50         missing.loc[missing.column == encoded_col, 'n_missing'] =
51             (df[encoded_col] == encoded_val).sum()
52     # persentase data yang kosong
53     missing['pct_missing'] = np.round(100 * missing.n_missing / df.shape[0], 2)
54     return missing
55
56 # Hanya data pengguna yang memiliki nilai yang kosong
57 # nilai 'age' yang kosong telah dikodekan dengan angka 118
58 show_missing_values(profile, 'age', 118)
59
60 # Periksa apakah semua nilai yang kosong berasal dari baris yang sama
61 profile[profile.age == 118].info()
62
63 # Hapus pengguna dengan data yang kosong
64 missing_users = profile.loc[profile.gender.isnull(), 'id'].values
65
66 # event ID yang berkaitan dengan pengguna juga akan dihapus
67 missing_user_events = transcript[transcript.person.isin(missing_users)].index.values
68
69 transcript.drop(missing_user_events, inplace=True) # hapus event
70 profile.dropna(inplace=True) # hapus pengguna

```

```

72 print(profile.shape)
73 print(transcript.shape)
74
75 # fungsi untuk mendapatkan nilai yang unik
76 def get_unique_values(df_col, type_dict=False):
77     """
78     Get the unique values of a dataframe column with lists or
79     the unique keys of a column with dictionaries.
80     Args:
81         (1) df_col (Pandas series) - dataframe column with iterable values
82         (2) type_dict (bool) - True if value type is dictionary or
83                               False if value type is list
84     Returns:
85         Unique values from column iterables (list[str]).
86     """
87     if type_dict: # untuk tipe data dictionary
88         # mendapatkan lists of keys
89         df_col = df_col.apply(lambda d: list(d.keys()))
90     # mendapatkan unique values
91     return np.unique(np.concatenate(df_col.values)).tolist()
92
93 # fungsi untuk mengekstraksi kolom secara iteratif
94 def extract_from_iterable_col(df, old_col, drop_old_col=True, unique_values=None,
95                               unique_keys=None):
96     """
97     Extract data into new columns from a column with
98     iterable values like lists or dictionaries.
99     If extracting from dictionaries, the new column names
100     will be the dictionary keys and the
101     values will be dictionary values. If extracting from lists,
102     one-hot encode the list values.
103     Args:
104         (1) df (Pandas dataframe) - data containing a column with iterable values
105         (2) old_col (str) - name of column to extract data from
106         (3) drop_old_col (bool) - whether or not to drop the old column
107                                after extracting data from it
108         (4) unique_vals (list[str]) - pass in unique values
109                                     if data type of column is list *
110         (5) unique_keys (list[str]) - pass in unique keys if data type
111                                     of column is dictionary *
112         * Pass in arg (4) for a list column or arg (5) for a dictionary column
113     Returns:
114         Same data with new columns extracted from the old column (Pandas dataframe).
115     """
116     df = df.copy()
117     if unique_keys is not None: # untuk tipe data dictionary
118         for k in unique_keys:
119             if ' ' not in k: # untuk melompati key yang terduplikasi dengan spasi
120                 # mendapatkan nilai dari dictionary
121                 df[k] = df[old_col].apply(lambda d: d[k] if k in d
122                                           # untuk key yang terduplikasi dengan spasi
123                                           else (d[k.replace('_', ' ')] if k.replace('_', ' ') in d
124                                                  # nilai NaN (Not a Number) jika key tidak ada dalam dictionary
125                                                  else np.NaN))
126     elif unique_values is not None: # untuk tipe data list
127         for v in unique_values:
128             # 1 jika nilai dalam list selain 0
129             new_col = df[old_col].apply(lambda lst: int(v in lst))
130             if np.var(new_col): # jika kolom baru tidak bernilai konstan
131                 df[v] = new_col # tambah kolom baru
132     if drop_old_col:
133         df.drop(old_col, axis=1, inplace=True)
134     return df
135
136 # unique channel values dari dataset portfolio
137 channels = get_unique_values(portfolio.channels)
138 # unique offer keys dari dataset transcript
139 offer_keys = get_unique_values(transcript.value, type_dict=True)
140
141 print("Unique values dari dataset portfolio:", channels)
142 print("Unique values dari dataset transcript:", offer_keys)

```

```

144 # Ekstraksi channels dari dataset portfolio
145 portfolio = extract_from_iterable_col(portfolio, 'channels', unique_values=channels)
146 portfolio.head()
147
148 # Ekstraksi nilai dictionary dari dataset transcript
149 transcript = extract_from_iterable_col(transcript, 'value', unique_keys=offer_keys)
150 transcript.head()
151
152 # Urutkan jenis penawaran berdasarkan tingkat kesulitan: informational < discount < BOC
153 portfolio.sort_values(['offer_type', 'difficulty', 'duration'],
154                       ascending=[False, True, False], inplace=True)
155 portfolio.reset_index(drop=True, inplace=True)
156 portfolio.head()
157
158 # fungsi untuk membuat mapping dictionary
159 def create_map_dict(series):
160     """
161     Create a mapping of unique hash strings of a dataframe column to integers.
162     Args:
163         series (Pandas series) - dataframe column with hash strings
164     Returns:
165         Mapping of unique hash strings to integers (dict)
166     """
167     ids, idn = dict(), 1
168     for s in series.unique():
169         ids[s] = idn
170         idn += 1
171     return ids
172
173 # fungsi untuk mapping ID string ke numerik
174 def map_ids_to_num(series, map_dict):
175     """
176     Map ID hash strings of a dataframe column to integers.
177
178     Args:
179         (1) series (Pandas series) - dataframe column to map
180         (2) map_dict (dict) - mapping with hash strings as keys and integers as values
181
182     Returns:
183         Dataframe column with values mapped to integers (Pandas series).
184     """
185
186     return series.map(map_dict)
187
188 # Membuat mapping untuk offer ID dan user ID
189 offer_mapping = create_map_dict(portfolio.id)
190 user_mapping = create_map_dict(profile.id)
191
192 len(offer_mapping)
193 len(user_mapping)
194
195 # Konversi offer ID dalam offer metadata menjadi numerik
196 portfolio['offer_id'] = map_ids_to_num(portfolio.id, offer_mapping)
197 portfolio.drop('id', axis=1, inplace=True)
198 portfolio
199
200 # Konversi user ID dalam user data menjadi numerik
201 profile['user_id'] = map_ids_to_num(profile.id, user_mapping)
202 profile.drop('id', axis=1, inplace=True)
203 profile.head()
204
205 # Konversi offer ID dan user ID dalam event data menjadi numerik
206 transcript['offer_id'] = map_ids_to_num(transcript.offer_id, offer_mapping)
207 transcript['user_id'] = map_ids_to_num(transcript.person, user_mapping)
208 transcript.drop('person', axis=1, inplace=True)
209 transcript.head()
210
211 # Data yang kosong dalam event data (dataset transcript)
212 show_missing_values(transcript)
213
214 # Perhitungan nilai yang kosong untuk setiap jenis peristiwa
215 print('Perhitungan peristiwa (event) secara keseluruhan:')
216 print(transcript.event.value_counts())

```

```

219 for e in transcript.event.unique():
220     print(f'\nNilai yang kosong untuk peristiwa (event): ', e)
221     print(transcript[transcript.event == e].isnull().sum().iloc[2:5])
222
223 # Isi nilai yang kosong dengan angka 0
224 transcript.fillna(0, inplace=True)
225 transcript.isnull().sum().sum() # semua kolom terisi data
226
227 # Periksa tipe data
228 print(portfolio.info())
229 print(profile.info())
230 print(transcript.info())
231
232 # Kolom yang akan dikonversi menjadi tipe data integer
233 profile['income'] = profile.income.astype(int) # `income` dalam user data
234 # `offer_id` dalam event data
235 transcript['offer_id'] = transcript.offer_id.astype(int)
236
237 transcript['reward'] = transcript.reward.astype(int) # `reward` dalam event data
238 transcript.head()
239
240 # Konversi `became_member_on` dalam user data menjadi
241 # tipe data datetime dan ubah nama kolom menjadi `signup_date`
242 profile['signup_date'] = pd.to_datetime(profile.became_member_on.astype(str))
243 profile.drop('became_member_on', axis=1, inplace=True)
244 profile.head()
245
246 print('Jumlah pengguna yang terduplikasi:', profile.duplicated().sum())
247 print('Jumlah peristiwa (event) yang terduplikasi:', transcript.duplicated().sum())
248
249 # Peristiwa yang terduplikasi
250 dup_events = transcript[transcript.duplicated(keep=False)]
251 dup_events
252
253 # Jenis peristiwa (event) yang terduplikasi
254 dup_events.event.value_counts()
255
256 # Hapus data duplikat
257 transcript.drop_duplicates(inplace=True)
258 transcript.duplicated().sum() # semua data duplikat telah dihapus
259
260 # fungsi untuk memfilter peristiwa (event)
261 def filter_for_events(event_list, event_df=transcript, event_col='event',
262                      cols_to_keep=['event', 'time', 'offer_id', 'user_id', 'amount'],
263                      merge_with=None, merge_on=None):
264     """
265     Filter the event data for specified events and optionally perform a left merge
266     with the offer metadata or user data.
267     Args:
268         (1) event_list (list[str]) - events to filter for
269         (2) event_df (Pandas df) - event data
270         (3) event_col (str) - name of event type column
271         (4) cols_to_keep (list[str]) - columns to keep in the filtered dataframe
272         (5) merge_with (Pandas dataframe) - offer metadata or user data
273         (6) merge_on (str) - column to merge on
274     Returns:
275         Filtered event data containing only the specified events (Pandas dataframe)
276     """
277     filtered_event = event_df.loc[event_df[event_col].isin(event_list),
278                                  cols_to_keep].reset_index()
279     if merge_with is not None:
280         filtered_event = pd.merge(filtered_event, merge_with, on=merge_on, how='left')
281     return filtered_event
282
283 # Filter untuk peristiwa "offer received" & lakukan left-join dengan dataset portfolio
284 received_offers = filter_for_events(['offer received'], cols_to_keep=['time', 'user_id',
285                               'offer_id'], merge_with=portfolio, merge_on='offer_id')
286
287 print(f'Total sebanyak {received_offers.shape[0]} offers telah dikirimkan kepada
288       {received_offers.user_id.nunique()} pengguna.\n')
289 received_offers.head()

```

```

291 # Konversi jam ke hari
292 received_offers['day'] = received_offers.time // 24
293 received_offers.day.value_counts()
294
295 print('Jumlah peristiwa "offers received" berdasarkan tipe:',
296       f'\n{received_offers.offer_type.value_counts()}')
297 print('\nJumlah peristiwa "offers received" berdasarkan offer ID:',
298       f'\n{received_offers.offer_id.value_counts()}')
299
300 # Filter untuk peristiwa reward offers
301 offer_events = filter_for_events(['offer received', 'offer viewed', 'offer completed'],
302                                  merge_with=portfolio, merge_on='offer_id')
303 # drop informational offers
304 offer_events = offer_events.query('offer_type != "informational"')
305 offer_events.shape
306 offer_events.head()
307
308 def concat_str_cols(df, cols, sep='_', pref='', suff=''):
309     """
310     Concatenate the string values of multiple columns.
311     Args:
312         (1) df (Pandas dataframe) - data
313         (2) cols (list[str]) - names of columns to concatenate
314         (3) sep (str or list[str]) - separator between columns
315         (4) pref (str) - string to prepend to the final values
316         (5) suff (str) - string to append to the final values
317     Returns:
318         Column with concatenated strings (Pandas series).
319     """
320     if isinstance(sep, str):
321         sep = [sep] * (len(cols) - 1) + [suff]
322     new_col = pref
323     for i in range(len(cols)):
324         new_col += df[cols[i]].astype(str) + sep[i]
325     return new_col
326
327 # Tambahkan waktu kedaluwarsa
328 offer_events['expire_time'] = offer_events.apply(lambda e: e.duration * 24 + e.time
329                                                  if e.event == 'offer received' else -1, axis=1)
330
331 # Satukan offer ID and user ID untuk mengelompokkan penawaran
332 offer_events['oid_uid'] = concat_str_cols(offer_events, ['offer_id', 'user_id'])
333 offer_events = offer_events.sort_values(['oid_uid', 'index']).reset_index(drop=True)
334 offer_events.head()
335
336 # fungsi untuk mengisi nilai
337 def fill_with(view_val, comp_val, rows_to_fill, off_evt_df=offer_events,
338              view_col='viewed', comp_col='completed'):
339     """
340     Fill specified rows of the new binary features in the offer event data
341     with 1 if yes or 0 if no. For offers being
342     viewed, the fill value is 1 only if the offer was viewed
343     BEFORE it was completed, otherwise it is 0
344     Args:
345         (1) view_val (int) - 1 if the offer was viewed, else 0
346         (2) comp_val (int) - 1 if the offer was completed, else 0
347         (3) rows_to_fill (list[int]) - indices of rows to fill
348         (4) off_evt_df (Pandas dataframe) - offer event data
349         (5) view_col (str) - name of binary column indicating whether the offer was vie
350         (6) comp_col (str) - name of binary column indicating whether the offer was con
351     Returns: None
352     """
353     for row in rows_to_fill:
354         if view_val in [0, 1]:
355             off_evt_df.loc[row, view_col] = view_val
356         if comp_val in [0, 1]:
357             off_evt_df.loc[row, comp_col] = comp_val

```

```

359 # fungsi untuk menuju ke event yang dimaksud
360 def go_to_event(curr_idx, delta=1, event_df=offer_events):
361     """
362     Move from the current event index to another event index.
363     Example: if delta is -1, go to the previous index
364             if delta is 1, go to the next index
365     Args:
366         (1) curr_idx (int) - current index
367         (2) delta (int) - change in index
368         (3) event_df (Pandas dataframe) - event data
369     Returns:
370         (1) Index of new event (int).
371         (2) New event (Pandas series) if index within range
372             or index (int) if out of range
373     """
374     next_idx = next_evt = curr_idx + delta
375     if 0 <= next_idx < event_df.shape[0]:
376         next_evt = event_df.iloc[next_idx]
377     return next_idx, next_evt
378
379 # Buat fitur biner untuk menunjukkan apakah penawaran telah dilihat dan diselesaikan
380 offer_events['viewed'] = -1
381 offer_events['completed'] = -1
382 offer_events.head()
383
384 # Isi nilai 'viewed' dan 'completed'
385 for idx, ids, evt in offer_events[['oid_uid', 'event']].itertuples():
386     # peristiwa saat ini: offer received
387     if idx + 2 < offer_events.shape[0] and evt == 'offer received':
388         # dua peristiwa selanjutnya
389         evt2, evt3 = offer_events.iloc[idx + 1], offer_events.iloc[idx + 2]
390         if evt2.oid_uid == ids and evt2.event == 'offer viewed':
391             # dua peristiwa selanjutnya: offer viewed, offer completed
392             if evt3.oid_uid == ids and evt3.event == 'offer completed':
393                 fill_with(1, 1, [idx, idx + 1, idx + 2])
394             else: # dua peristiwa selanjutnya: offer viewed, offer received atau offer
395                 fill_with(1, 0, [idx, idx + 1])
396         elif evt2.oid_uid == ids and evt2.event == 'offer completed':
397             # dua peristiwa selanjutnya: offer completed, offer viewed
398             if evt3.oid_uid == ids and evt3.event == 'offer viewed':
399                 fill_with(0, 1, [idx, idx + 1, idx + 2])
400             else: # dua peristiwa selanjutnya: offer completed, offer received atau off
401                 fill_with(0, 1, [idx, idx + 1])
402         else: # peristiwa selanjutnya: "offer received"
403             fill_with(0, 0, [idx])
404     # Jika suatu peristiwa tidak diisi, itu karena ada dau peristiwa penyelesaian
405     # berturut-turut tanpa "received" atau "viewed" di antaranya
406     elif offer_events.loc[idx, 'viewed'] < 0:
407         if evt == 'offer viewed': # peristiwa saat ini: offer viewed
408             fill_with(0, 1, [idx])
409         if evt == 'offer completed': # peristiwa saat ini: offer completed
410             # Kembali 3 baris unuk peristiwa 0
411             idx0, evt0 = go_to_event(idx, -3)
412             fill_viewed = True
413             # Ulangi secara mundur peristiwa 0 untuk menemukan penawaran
414             # yang dimiliki oleh penyelesaian ini
415             while not isinstance(evt0, int) and evt0.oid_uid == ids: # lanjutkan sampai
416                 if evt0.completed < 1:
417                     if evt0.event == 'offer viewed' and fill_viewed: # peristiwa 0: off
418                         fill_with(-1, 1, [idx0])
419                         fill_viewed = False
420                     if evt0.event == 'offer received': # peristiwa 0: offer received
421                         fill_with(-1, 1, [idx0])
422                         fill_with(evt0.viewed, 1, [idx])
423                     break
424             idx0, evt0 = go_to_event(idx0, -1) # kembali satu baris
425
426 print('Jumlah peristiwa yang tidak diisi:',
427       offer_events.query('viewed < 0 or completed < 0').shape[0])
428 offer_events.head()

```

```

430 # fungsi untuk mengelompokkan penawaran
431 def group_offer(event_row, viewed_col='viewed', completed_col='completed'):
432     """
433     Helper function to assign a group label to an offer-related event.
434     Group 1: offers that were neither viewed or completed
435     Group 2: offers that were viewed, but not completed
436     Group 3: offers that were completed, but not viewed
437     Group 4: offers that were both viewed and completed
438     Args:
439         (1) event_row (Pandas Series) - row in event data
440         (2) viewed_col (str) - name of 'viewed' binary column
441         (3) completed_col (str) - name of 'completed' binary column
442     Returns:
443         Group number the offer belongs to (int).
444     """
445     if event_row[viewed_col] == 0 and event_row[completed_col] == 0:
446         return 1
447     if event_row[viewed_col] == 1 and event_row[completed_col] == 0:
448         return 2
449     if event_row[viewed_col] == 0 and event_row[completed_col] == 1:
450         return 3
451     if event_row[viewed_col] == 1 and event_row[completed_col] == 1:
452         return 4
453
454 # Tambahkan fitur 'group' untuk mengelompokkan penawaran seperti yang dijelaskan di atas
455 offer_events['group'] = offer_events.apply(group_offer, axis=1)
456 offer_events.head()
457
458 def show_group_count(event_df, plot_counts=True):
459     """
460     Print a summary of offer groups and optionally plot
461     a stacked horizontal bar chart of offer group counts.
462     Args:
463         (1) event_df (Pandas dataframe) - event data
464         (2) plot_counts (bool) - whether to plot the offer group counts
465     Returns: None.
466     """
467     # Total penawaran yang diterima dan diselesaikan
468     received_offers = event_df.query('event == "offer received"')
469     completed_offers = received_offers.query('completed == 1')
470     print('Total penawaran yang dikirimkan:', received_offers.shape[0])
471     print('Penawaran yang diselesaikan:', completed_offers.shape[0])
472     print('Penawaran yang tidak selesai:', received_offers.shape[0] - completed_offers.shape[0])
473     # Jumlah penawaran di setiap grup
474     group_count = received_offers.groupby('group').value_counts()
475     print('\n(Group 1) Penawaran yang tidak dilihat dan tidak diselesaikan:', group_count[1])
476     print('(Group 2) Penawaran yang dilihat, tapi tidak diselesaikan:', group_count[2])
477     print('(Group 3) Penawaran yang diselesaikan, tapi tidak dilihat:', group_count[3])
478     print('(Group 4) Penawaran yang dilihat dan diselesaikan:', group_count[4])
479     # Plotting
480     if plot_counts:
481         data = [go.Bar(x=[group_count[1], group_count[3]],
482                        y=['Offers not completed', 'Offers completed'],
483                        name='Not viewed', orientation='h'),
484                go.Bar(x=[group_count[2], group_count[4]],
485                        y=['Offers not completed', 'Offers completed'],
486                        name='Viewed', orientation='h')]
487         go.Figure(data=data, layout=go.Layout(barmode='stack',
488                                                title='Offer Groups',
489                                                xaxis=dict(title='Number of offers'))).show()
490 # Menampilkan jumlah penawaran di setiap grup
491 print('Reward Offers')
492 print('-----')
493 show_group_count(offer_events)
494
495 # Filter untuk penawaran informasional, merge dengan dataset portfolio,
496 # dan mengisi nilai kosong dengan angka -1
497 info_events = filter_for_events(['offer received', 'offer viewed', 'transaction'],
498                                 merge_with=portfolio, merge_on='offer_id').fillna(-1)
499 info_events = info_events[info_events.offer_id.isin([0, 1, 2])] # menghapus reward offer

```



```

501 # Konversi kolom metadata ke tipe data integer
502 for col in ['reward', 'difficulty', 'duration', 'mobile', 'social', 'web']:
503     info_events[col] = info_events[col].astype(int)
504
505 info_events.info()
506
507 # Hapus transaksi pengguna yang tidak menerima penawaran informasional
508 users_with_info_offers = info_events.query('event != "transaction"').user_id.unique()
509 info_events = info_events[info_events.user_id.isin(users_with_info_offers)]
510 info_events.shape
511
512 # Tambahkan waktu kedaluwarsa (durasi dimulai ketika penawaran dilihat)
513 info_events['expire_time'] = info_events.apply(lambda e: -1
514                                                if e.event == 'transaction' else e.duration * 24 + e.time,
515                                                axis=1)
516
517 # Gabungkan offer ID dan user ID untuk mengelompokkan penawaran
518 info_events['oid_uid'] = concat_str_cols(info_events, ['offer_id', 'user_id'])
519 info_events = info_events.sort_values(['user_id', 'index']).reset_index(drop=True)
520 info_events.head()
521
522 # Buat fitur biner untuk menunjukkan apakah penawaran telah dilihat dan diselesaikan
523 info_events['viewed'] = -1
524 info_events['completed'] = -1
525
526 # Isi nilai 'viewed' dan 'completed'
527 for idx, uid, oid, evt in info_events[['user_id', 'offer_id', 'event']].itertuples():
528     if evt == 'offer received': # peristiwa saat ini: offer received
529         # inisialisasi peristiwa saat ini sebagai tidak dilihat dan tidak selesai
530         view_val = comp_val = 0
531         rows_to_fill = [idx]
532         idx2, evt2 = go_to_event(idx, event_df=info_events) # peristiwa selanjutnya
533
534         # Periksa apakah penawaran telah dilihat
535         while not isinstance(evt2, int) and uid == evt2.user_id:
536             # peristiwa selanjutnya: offer viewed
537             if evt2.event == 'offer viewed' and evt2.viewed < 0:
538                 view_val = 1 # perbarui menjadi viewed
539                 rows_to_fill.append(idx2) # untuk menerapkan perubahan pada kedua peris
540                 break
541             idx2, evt2 = go_to_event(idx2, event_df=info_events) # peristiwa selanjutnya
542
543         # Dapatkan waktu mulai dan berakhir dari penawaran
544         if len(rows_to_fill) == 1: # penawaran tidak dilihat
545             # kembali ke peristiwa saat ini
546             idx2, evt2 = go_to_event(idx, 0, event_df=info_events)
547             start, end = evt2[['time', 'expire_time']].values
548             idx2, evt2 = go_to_event(idx2, event_df=info_events)
549
550         # Periksa transaksi dalam durasi
551         while not isinstance(evt2, int) and uid == evt2.user_id:
552             # peristiwa selanjutnya: transaction
553             if evt2.event == 'transaction' and evt2.time < end and evt2.amount >= 2.5:
554                 comp_val = 1 # perbarui menjadi viewed
555                 break
556             idx2, evt2 = go_to_event(idx2, event_df=info_events) # peristiwa selanjutnya
557
558         # Isi dengan nilai yang baru
559         fill_with(view_val, comp_val, rows_to_fill, info_events)
560
561
562 print('Jumlah penawaran yang tidak diisi:', info_events.query('event != "transaction"
563                                                                and (viewed < 0 or completed < 0)').shape[0])
564 info_events.loc[info_events.event != 'transaction', ['viewed', 'completed']].value_counts()
565
566 # Tambahkan fitur 'group' untuk mengelompokkan reward offers
567 info_events['group'] = info_events.apply(group_offer, axis=1).fillna(-1).astype(int)
568
569 # Tunjukkan jumlah penawaran di setiap grup
570 print('Informational Offers')
571 print('-----')
572 show_group_count(info_events)

```



```

574 # Gabungkan peristiwa penawaran informasional dan reward offers
575 offers = pd.concat([offer_events, info_events.query('event != "transaction"')])
576 offers = offers.sort_values('index').reset_index(drop=True)
577 offers.head()
578
579 # Hitung penawaran valid berdasarkan ID. penawaran selesai yang belum dilihat tidak di
580 # hapus penawaran yang telah diselesaikan, tetapi tidak dilihat
581 true_offers = offers.query('event == "offer received" and group != 3')
582 true_by_offer = true_offers.offer_id.value_counts().reset_index()
583 true_by_offer.columns = ['offer_id', 'n_offers']
584
585 # Hitung penawaran yang diselesaikan berdasarkan ID
586 # penawaran yang dilihat, lalu diselesaikan
587 comp_offers = offers.query('event == "offer received" and group == 4')
588 comp_by_offer = comp_offers.offer_id.value_counts().reset_index()
589 comp_by_offer.columns = ['offer_id', 'n_completed']
590 true_offers.shape, comp_offers.shape
591
592 # Hitung tingkat penyelesaian berdasarkan offer ID
593 comp_by_offer = pd.merge(comp_by_offer, true_by_offer, on='offer_id', how='left')
594 comp_by_offer['pct_completed'] = (100 * comp_by_offer.n_completed /
595                                   comp_by_offer.n_offers).round(2)
596 comp_by_offer.sort_values('pct_completed', ascending=False, inplace=True)
597 comp_by_offer
598
599 # Gabungkan dengan offer metadata
600 comp_by_offer = pd.merge(comp_by_offer, portfolio, on='offer_id', how='left')
601
602 # Menggabungkan informasi lain untuk plotting
603 comp_by_offer['offer'] = concat_str_cols(comp_by_offer,
604                                           ['offer_id', 'offer_type', 'difficulty', 'reward', 'duration']
605                                           ['.', ' ', ' - \$', ' => \$', ' (exp in ', ' days)'])
606 comp_by_offer['summary'] = concat_str_cols(comp_by_offer,
607                                           ['n_offers', 'n_completed', 'pct_completed'],
608                                           [' | Completed: ', ' (', '%)'], pref='Received: ')
609
610 comp_by_offer.head()
611
612 # Plotting tingkat penawaran yang selesai
613 plt.figure(figsize=(8, 4))
614 sns.barplot(data=comp_by_offer, x='pct_completed', y='offer', orient='h')
615 plt.legend(handles=[plt.Rectangle((0,0), 1, 1, color=color) for color in sns.color_palette(
616                                   labels=comp_by_offer.summary.tolist(), fontsize='large', bbox_to_anchor=(1,
617 plt.title('Tingkat Penyelesaian dari Semua Penawaran', fontsize='xx-large')
618 plt.ylabel('Offer ID dan tipe penawaran (belanjaan a
619           => dapatkan reward b, durasi)', fontsize='large')
620 plt.xlabel('Persentase dari penawaran yang selesai', fontsize='large');
621
622 # Menambahkan fitur "signup_year"
623 profile['signup_year'] = profile.signup_date.dt.year
624 profile.describe()

```

```

626 # fungsi untuk plotting data
627 def plot_user_demographics(user_df, title='Gambaran Demografi Pengguna',
628                             height=600, width=1000, vspacing=0.15,
629                             gender_col='gender', age_col='age',
630                             signup_col='signup_year', income_col='income'):
631     '''
632     Plot the distribution of user demographics - 'gender', 'age', 'signup year', and 'income'
633     Args:
634         (1) user_df (Pandas dataframe) - user data
635         (2) title (str) - plot title
636         (3) height (int) - figure height
637         (4) width (int) - figure width
638         (5) vspacing (float) - vertical spacing between subplots
639         (6) gender_col (str) - name of gender column
640         (7) age_col (str) - name of age column
641         (8) signup_col (str) - name of signup column
642         (9) income_col (str) - name of income column
643     Returns: None
644     '''
645     fig = subplots.make_subplots(2, 2, vertical_spacing=vspacing, y_title='Number of users',
646                                   subplot_titles=['Gender', 'Age', 'Signup Year', 'Income'])
647     fig.add_trace(go.Histogram(x=user_df[gender_col].sort_values(), name='Gender'), 1, 1)
648     fig.add_trace(go.Histogram(x=user_df[age_col], name='Age'), 1, 2)
649     fig.add_trace(go.Histogram(x=user_df[signup_col], name='Signup Year'), 2, 1)
650     fig.add_trace(go.Histogram(x=user_df[income_col], name='Income'), 2, 2)
651     fig.update_layout(title=title, height=height, width=width, showlegend=False)
652     fig.show()
653
654 # Plotting demografi pengguna
655 plot_user_demographics(profile)
656
657 # Kelompokkan usia dan pendapatan
658 profile['age_group'] = pd.qcut(profile.age, 5, labels=range(1, 6)).astype(int)
659 profile['income_group'] = pd.qcut(profile.income, 5, labels=range(1, 6)).astype(int)
660 profile.shape
661
662 # Hitung jumlah total yang dibelanjakan oleh setiap pengguna
663 total_spent = transcript.groupby('user_id').amount.sum().reset_index()
664 profile = pd.merge(profile, total_spent, on='user_id', how='left')
665
666 # Kelompokkan "user spending" dari setiap pengguna
667 profile['spend_group'] = pd.qcut(profile.amount, 10, range(1, 11)).astype(int)
668 profile.sort_values('amount', ascending=False).head()
669
670 # Plotting pendapatan menurut usia dan jenis kelamin
671 sns.boxplot(data=profile.query('gender != "O"'), x='age_group', y='income', hue='gender')
672 plt.title('Pendapatan berdasarkan Usia dan Jenis Kelamin', fontsize='x-large')
673 plt.legend(title='Gender', bbox_to_anchor=(1, 1));
674
675 # Box plot dengan warna
676 fig, ax = plt.subplots(1, 2, sharey=True, figsize=(16, 4))
677 fig.suptitle('Jumlah Total yang Dibelanjakan berdasarkan Demografi Pengguna', fontsize=14)
678 sns.boxplot(data=profile.query('gender != "O"'), x='age_group',
679             y='amount', hue='gender', showfliers=False, ax=ax[0])
680 sns.boxplot(data=profile.query('gender != "O"'), x='income_group',
681             y='amount', hue='gender', showfliers=False, ax=ax[1])
682 ax[0].legend(title='Gender', bbox_to_anchor=(1.025, 1))
683 ax[1].legend('');
684
685 # Fungsi agregat untuk setiap fitur
686 agg_funcs = {
687     'amount': 'mean', # avg amount spent
688     'user_id': 'count', # num users
689     'signup_year': 'median', # med signup year
690     'gender': lambda g: (g == 'M').mean() * 100, # % male
691     'age': 'mean', # avg age
692     'income': 'mean', # avg income
693 }
694
695 # Hitung demografi pengguna berdasarkan pembelanjaan (10 kuantil)
696 spending_q10 = profile.groupby('spend_group').agg(agg_funcs).astype(int)
697 spending_q10.columns = ['avg_spent', 'n_users', 'median_signup', 'pct_male', 'avg_age',
698                        'avg_income']
699 spending_q10['pct_female'] = 100 - spending_q10.pct_male

```

```

701 # Plotting perubahan dalam demografi pengguna saat pembelanjaan meningkat
702 fig = subplots.make_subplots(3, 2, vertical_spacing=0.1, x_title='x = Spend Group',
703                             subplot_titles=['Average Amount Spent', 'Median Signup Year',
704                                             'Average Income', 'Average Age',
705                                             'Percent of Female', 'Percent of Male'])
706 fig.add_trace(go.Scatter(x=spending_q10.index,
707                          y=spending_q10.avg_spent, name='Average amount spent'), 1, 1)
708 fig.add_trace(go.Scatter(x=spending_q10.index,
709                          y=spending_q10.median_signup, name='Median signup year'), 1, 2)
710 fig.add_trace(go.Scatter(x=spending_q10.index,
711                          y=spending_q10.avg_income, name='Average income'), 2, 1)
712 fig.add_trace(go.Scatter(x=spending_q10.index,
713                          y=spending_q10.avg_age, name='Average age'), 2, 2)
714 fig.add_trace(go.Scatter(x=spending_q10.index,
715                          y=spending_q10.pct_female, name='Percent female'), 3, 1)
716 fig.add_trace(go.Scatter(x=spending_q10.index,
717                          y=spending_q10.pct_male, name='Percent male'), 3, 2)
718
719 fig.update_layout(title='Demografi Pengguna saat Pembelanjaan Meningkat',
720                  height=800, width=1000, showlegend=False)
721 fig.update_yaxes(nticks=3, row=1, col=2)
722 fig.show()
723
724 # heatmap untuk mengetahui korelasi antarfitur
725 plt.figure(figsize=(10, 6))
726 sns.heatmap(spending_q10.corr(), annot=True)
727 plt.xticks(rotation=45, ha='right');
728
729 # Gabungkan user data dengan event data
730 offers = pd.merge(offers, profile.drop('amount', axis=1), on='user_id', how='left')
731 # Ekstraksi penawaran valid berdasarkan ID - penawaran yang belum dilihat tidak dihitung
732 true_offers = offers.query('event == "offer received" and group != 3')
733 true_offers.head()
734
735 def plot_demographic_completion(data, hue, x='offer_id', y='completed'):
736     """
737     Plot offer completion rate by a specified user demographic feature.
738     Args:
739         (1) data (Pandas dataframe) - data to plot
740         (2) hue (str) - name of demographic feature column
741         (3) x (str) - name of offer ID column
742         (4) y (str) - name of completed indicator column
743     Returns: None
744     """
745     title = hue.replace("_", " ").title()
746     plt.figure(figsize=(16, 4))
747     sns.barplot(data=data, x=x, y=y, hue=hue)
748     plt.title(f'Tingkat Penyelesaian Penawaran berdasarkan {title}', fontsize='xx-large')
749     plt.ylabel('Persentase Penawaran Diselesaikan')
750     plt.xlabel('Offer ID')
751     plt.legend(title=title, bbox_to_anchor=(1, 1))
752     plt.show()
753
754 # Plotting tingkat penyelesaian penawaran berdasarkan demografi pengguna
755 for feat in ['gender', 'age_group', 'income_group', 'signup_year', 'spend_group']:
756     print('Perhitungan untuk:', true_offers.groupby(feat).offer_id.count())
757     plot_demographic_completion(true_offers, feat)

```

```

1 import joblib
2 import numpy as np
3 import pandas as pd
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 import plotly.express as px
7 import plotly.graph_objs as go
8 import plotly.subplots as subplots
9 from pandas.testing import assert_frame_equal
10 from sklearn import preprocessing
11 from sklearn.decomposition import PCA
12 from sklearn.preprocessing import StandardScaler, MinMaxScaler
13 from sklearn.cluster import KMeans
14 from yellowbrick.cluster.elbow import kelbow_visualizer
15 from google.colab import drive
16 %matplotlib inline
17 sns.set()
18
19 drive.mount('/content/drive', force_remount=True)
20 profile = pd.read_pickle('/content/drive/My Drive/data/out-analysis/profile.pkl')
21 transcript = pd.read_pickle('/content/drive/My Drive/data/out-analysis/transcript.pkl')
22 offers = pd.read_pickle('/content/drive/My Drive/data/out-analysis/offers.pkl')
23 profile.shape, transcript.shape, offers.shape
24
25 print(profile.info())
26 profile.head()
27 print(transcript.info())
28 transcript.head()
29 print(offers.info())
30 offers.head()
31
32 # Ekstraksi data transaksi
33 transactions = transcript.query('event == "transaction"').drop('event', axis=1)
34
35 # Gabungkan dengan user data
36 transactions = pd.merge(transactions, profile.drop(['amount', 'spend_group'], axis=1),
37 transactions.head()
38
39 # Buat fitur biner yang menunjukkan apakah jumlah transaksi setidaknya 1 dolar
40 transactions['gte1'] = transactions.amount.apply(lambda a: int(a >= 1))
41
42 # Hitung frequency, monetary untuk semua pengguna
43 fmt = transactions.groupby('user_id', as_index=False).agg({
44     'gte1': 'sum', # frequency
45     'amount': 'sum', # monetary
46 })
47 fmt.columns = ['user_id', 'frequency', 'monetary']
48
49 # Tambahkan pengguna dengan transaksi awal = 0
50 fmt = pd.merge(profile[['user_id', 'signup_date']], fmt, on='user_id', how='left')
51 fmt = fmt.set_index('user_id').fillna(0)
52 fmt['frequency'] = fmt.frequency.astype(int)
53 fmt.head()
54
55 # Tentukan tanggal referensi untuk pembobotan tenure
56 ref_date = pd.to_datetime('2021-12-31 00:00')
57
58 # Hitung tenure untuk semua pengguna
59 fmt['tenure'] = fmt.signup_date.apply(lambda d: (ref_date - d).days // 30) # bulan
60 fmt.drop('signup_date', axis=1, inplace=True)
61 fmt.head()
62
63 # Analisis deskriptif
64 fmt.describe()

```

```

66 def FMT_DistributionPlot(fmt_df):
67     # Distribusi frequency
68     plt.figure(figsize=[5,3])
69     fmt_hist = sns.histplot(data = fmt_df, x = 'frequency', multiple = "dodge",
70                             element = 'step', bins = 30, kde = True,
71                             stat = 'count').set_title("Distribusi Frequency")
72     plt.xlabel("Frequency")
73     plt.ylabel("Jumlah pengguna")
74
75     # Distribusi monetary
76     plt.figure(figsize=[5,3])
77     fmt_hist = sns.histplot(data = fmt_df, x = 'monetary'
78                             multiple = "dodge", element = 'step',
79                             bins = 20, kde = True, stat = 'count')
80     .set_title("Distribusi Monetary")
81     plt.xlabel("Monetary")
82     plt.ylabel("Jumlah pengguna")
83
84     # Distribusi tenure
85     plt.figure(figsize=[5,3])
86     fmt_hist = sns.histplot(data = fmt_df, x = 'tenure',
87                             multiple = "dodge", element = 'step',
88                             bins = 100, kde = True, stat = 'count')
89     .set_title("Distribusi Tenure")
90     plt.xlabel("Tenure (bulan)")
91     plt.ylabel("Jumlah pengguna")
92
93     return fmt_hist
94
95 # tampilkan hasil plotting
96 FMT_DistributionPlot(fmt)
97
98 # Memotong dan memperkecil range data ke bentuk kuantil tertentu
99 # 6 kuantil (1%-16,6%, 16,7%-33,2%, ..., 83,4%-100%)
100 fmt['F'] = pd.qcut(fmt.frequency, 6, range(1, 7)).astype(int)
101 # 8 kuantil (1%-12,5%, 12,6-25%, ..., 87,6%-100%)
102 fmt['M'] = pd.qcut(fmt.monetary, 8, range(1, 9)).astype(int)
103 # 3 kuantil (1%-33,3%, 33,4%-66,6%, 66,7%-100%)
104 fmt['T'] = pd.qcut(fmt.tenure, 3, range(1, 4)).astype(int)
105
106 # Gabungkan nilai atribut F, M, dan T ke dalam satu atribut FMT
107 # akan digunakan untuk fungsi agregat yang ada di proses selanjutnya
108 fmt['FMT'] = fmt['F'].astype(str) + fmt['M'].astype(str) + fmt['T'].astype(str)
109 fmt.head()
110
111 # Gabungkan FMT data dengan user data
112 fmt = pd.merge(profile, fmt.reset_index(), on='user_id', how='left')
113 fmt.head()
114
115 # Konversi gender ke dalam fitur biner
116 # Gender "others" dikategorikan sama dengan wanita (bukan pria = 0)
117 fmt['male'] = (fmt.gender == 'M').astype(int)
118 fmt.head()
119
120 # Menggunakan MinMax Scaler (MinMax Normalization)
121 scaler = MinMaxScaler()
122 cols = ['age', 'income', 'frequency', 'monetary', 'tenure']
123
124 userd_normalized = pd.DataFrame(scaler.fit_transform(fmt[cols]),
125                                 index=fmt.index, columns=cols)
126 userd_normalized = userd_normalized.set_axis(['age_norm', 'inc_norm',
127                                               'f_norm', 'm_norm', 't_norm'], axis=1)
128 userd_normalized.head()
129
130 # pahami data
131 userd_normalized.describe()
132
133 # Menggabungkan tabel hasil normalisasi data dengan FMT data
134 fmt = pd.concat([fmt, userd_normalized], axis=1)
135 fmt.head()
136
137 # menentukan jumlah komponen optimal untuk PCA
138 pca_test = PCA(random_state=0).fit(userd_normalized)

```

```

140 plt.rcParams["figure.figsize"] = (12,6)
141 fig, ax = plt.subplots()
142 xi = np.arange(1, 7, step=1)
143 y = np.cumsum(pca_test.explained_variance_ratio_)
144
145 plt.ylim(0.0,1.1)
146 plt.plot(xi, y, marker='o', linestyle='--', color='b')
147
148 plt.xlabel('Jumlah Komponen')
149 plt.xticks(np.arange(1, 7, step=1))
150 plt.ylabel('Cumulative explained variance (%)')
151 plt.title('Jumlah Komponen Optimal untuk PCA')
152
153 plt.axhline(y=0.95, color='r', linestyle='-')
154 plt.text(0.85, 0.85, 'ambang batas >= 95%', color = 'red', fontsize=16)
155
156 ax.grid(axis='x')
157 plt.show()
158
159 # Implementasi PCA
160 pca = PCA(n_components=5, random_state=0)
161 user_pca = pd.DataFrame(pca.fit_transform(userd_normalized),
162                        columns=['comp' + str(i) for i in range(1, 6)])
163 user_pca.head()
164
165 # pahami data
166 user_pca.describe()
167
168 # plotting histogram PCA
169 axs = user_pca.plot(kind='hist', bins=20, figsize=(18, 6), subplots=True,
170                  layout=(1, 3), title='Distribusi dari Tiap Komponen PCA')
171 plt.setp(axs[-1, :], xlabel='Range')
172 plt.show()
173
174 def heatmap(data,title,ylabel,xlabel='Feature',vmin=-1,vmax=1,cmap='RdBu'):
175     '''
176     Plot a heatmap.
177     Args:
178         (1) data (Pandas dataframe) - data to plot
179         (2) title (str) - plot name
180         (3) xlabel (str) - x-axis name
181         (4) ylabel (str) - y-axis name
182         (5) vmin (int or float) - colorbar minimum
183         (6) vmax (int or float) - colorbar maximum
184         (7) cmap (str) - named color palette
185     Returns: None.
186     '''
187     plt.figure(figsize=(8, 6))
188     sns.heatmap(data, annot=True, vmin=vmin, vmax=vmax, cmap=cmap, fmt='.3f')
189     plt.title(title, fontsize='x-large')
190     plt.xlabel(xlabel)
191     plt.ylabel(ylabel)
192     plt.yticks(rotation=0)
193     plt.show()
194
195 # korelasi tiap component dengan variabel demografis
196 user_comps = pd.DataFrame(pca.components_, index=user_pca.columns,
197                          columns=['male','age','income','frequency','monetary','tenure'])
198 user_comps.head()
199 # Heatmap komponen PCA
200 heatmap(user_comps, 'Korelasi dari Tiap Komponen PCA dengan Fiturnya', 'Component')
201 # Menggabungkan tabel hasil PCA dengan FMT data
202 fmt = pd.concat([fmt, user_pca], axis=1)
203 fmt.head()
204
205 # menentukan jumlah awal klaster (k)
206 kelbow_visualizer(KMeans(random_state=0), user_pca, k=(2,10));

```

```

208 def segment_means(fmt_df, cols=['frequency', 'monetary', 'tenure'],
209                      group_col='cols', count_col='FMT'):
210     """
211     Calculate the frequency, monetary, and tenure (FMT) means for each segment,
212     as well as the number and percentage of users in each segment.
213     Args:
214         (1) fmt_df (Pandas dataframe) - FMT data
215         (2) cols (list[str]) - names of columns to find the mean for
216         (3) group_col (str) - column to group by
217         (4) count_col (str) - column to count in the grouping
218     Returns:
219         Frequency, monetary, and tenure (FMT) means and number
220         and percentage of users for each segment (Pandas dataframe)
221     """
222     df = fmt_df.groupby(group_col).mean()[cols].round(2)
223     df['n_users'] = fmt_df.groupby(group_col)[count_col].count()
224     df['pct_users'] = (100 * df.n_users / fmt_df.shape[0]).round(2)
225     return df
226
227 # K-means clustering
228 kmeans = KMeans(4, random_state=0)
229 kmeans.fit(user_pca)
230 fmt['cluster'] = kmeans.labels_
231
232 cols = ['age', 'income', 'frequency', 'monetary', 'tenure', 'comp1', 'comp2', 'comp3']
233 # group by berdasarkan rata-rata (mean)
234 cluster_means = segment_means(fmt, cols, 'cluster')
235 cluster_means
236
237 fmt['customer_cluster'] = fmt.cluster.map({0: 'Gold Customers',
238                                           1: 'Silver Customers',
239                                           2: 'Diamond Customers',
240                                           3: 'Bronze Customers'})
241
242 fmt.head()
243
244 wcss = []
245
246 for k in range(2, 10):
247     kmeans = KMeans(k, random_state=0)
248     model = kmeans.fit(user_pca)
249     wcss.append(model.inertia_)
250
251 plt.figure(figsize=(10, 6))
252 sns.lineplot(x=range(2, 10), y=wcss, linestyle='--', marker='o')
253 plt.title('Grafik Penurunan WCSS pada Tiap Jumlah Klaster')
254 plt.ylabel('WCSS')
255 plt.xlabel('Jumlah klaster (k)')
256 plt.show()
257
258 # Perhitungan WCSS (Within-cluster Sum of Squares)
259 print("### Perhitungan WCSS pada Metode Elbow ###")
260
261 for k in range(2, 10):
262     kmeans = KMeans(k, random_state=0)
263     model = kmeans.fit(user_pca)
264     wcss = model.inertia_
265     if(k==4):
266         print("Untuk n_clusters = ", k, ", nilai WCSS :", wcss, " (penurunan mulai mendatar)")
267     else:
268         print("Untuk n_clusters = ", k, ", nilai WCSS :", wcss)
269
270 # heatmap
271 features = ['age', 'income', 'frequency', 'monetary', 'tenure']
272 overall_means = fmt[features].mean()
273 feat_importances = cluster_means.iloc[:, :5] / overall_means - 1
274 heatmap(feat_importances, 'Tingkat Kepentingan Fitur PCA dari Tiap Klaster', 'Cluster')

```

```

269 # heatmap
270 features = ['age', 'income', 'frequency', 'monetary', 'tenure']
271 overall_means = fmt[features].mean()
272 feat_importances = cluster_means.iloc[:, :5] / overall_means - 1
273 heatmap(feat_importances, 'Tingkat Kepentingan Fitur PCA dari Tiap Klaster', 'Cluster')
274
275 def snake_plot(segment_df, feats=['frequency', 'monetary', 'tenure'],
276               normalize=True, segment_col='cluster', user_col='user_id',
277               title='Snake Plot', legend_title='Fitur PCA', palette="Set1"):
278     Create a snake plot for the specified features of customer segments.
279     Args:
280         (1) segment_df (Pandas dataframe) - user data with defined segments
281         (2) feats (list[str]) - names of columns to plot
282         (3) normalize (bool) - whether to normalize the data before plotting
283         (4) segment_col (str) - name of segment column
284         (5) user_col (str) - name of user ID column
285         (6) title (str) - plot name
286         (7) legend_title (str) - legend name
287         (8) palette (list[str] or str) - named color palette or list of colors for bars
288     # Set index
289     if user_col in segment_df.columns:
290         segment_df.set_index(user_col, inplace=True)
291     # Normalize values
292     if normalize:
293         df = pd.DataFrame(StandardScaler().fit_transform(segment_df[feats]),
294                           index=segment_df.index, columns=feats)
295         df[segment_col] = segment_df[segment_col]
296     else:
297         df = segment_df[feats]
298     # Melt features
299     melt = pd.melt(df.reset_index(),
300                   id_vars=[user_col, segment_col], value_vars=feats,
301                   var_name='Feature', value_name='Value')
302     # Create a snake plot for customer segments
303     title += ' Fitur PCA dari Tiap Klaster'
304     sns.lineplot(data=melt, x='Feature', y='Value', hue=segment_col, palette=palette)
305     plt.title(title, fontsize='x-large')
306     plt.legend(title=legend_title, bbox_to_anchor=(1, 1))
307     plt.show()
308
309 # Snake plot
310 snake_plot(fmt, features, segment_col='cluster', legend_title='Cluster')
311
312 kmeans = KMeans(4, random_state=0)
313 kmeans.fit(user_pca)
314
315 # 2D scatter plot
316 fig, ax = plt.subplots(figsize=(8, 6))
317 sns.scatterplot(data=fmt, x='comp1', y='comp2', hue='cluster', palette='deep');
318 ax.scatter(kmeans.cluster_centers[:, 0], kmeans.cluster_centers[:, 1],
319           s=100, c='black', marker='*', label='Centroid')
320 fig.suptitle('Visualisasi 2D Scatter Plot untuk K-Means Clustering');
321
322 # 2D scatter plot
323 fig, ax = plt.subplots(1, 2, figsize=(16, 6))
324 sns.scatterplot(data=fmt, x='comp1', y='comp3', hue='cluster', ax=ax[0], palette='deep')
325 ax[0].scatter(kmeans.cluster_centers[:, 0], kmeans.cluster_centers[:, 1],
326             s=100, c='black', marker='*', label='Centroid')
327 sns.scatterplot(data=fmt, x='comp2', y='comp3', hue='cluster', ax=ax[1], palette='deep')
328 ax[1].scatter(kmeans.cluster_centers[:, 0], kmeans.cluster_centers[:, 1],
329             s=100, c='black', marker='*', label='Centroid')
330 fig.suptitle('Visualisasi 2D Scatter Plot untuk K-Means Clustering');
331
332 # 3D scatter plot
333 nextfig = px.scatter_3d(fmt, x='comp1', y='comp2', z='comp3',
334                        color='customer_cluster', size_max=40,
335                        title=f'3D Scatter Plot untuk K-Means Clustering (Comp1, Comp2, Con
336 nextfig.update_traces(marker=dict(size=6, line=dict(width=1, color='DarkSlateGrey')),
337                       selector=dict(mode='markers'))
338 nextfig.show()

```



```

340 # count plot
341 ax = sns.countplot(x='customer_cluster', data=fmt)
342 for p in ax.patches:
343     ax.annotate(f'\n{p.get_height()}', (p.get_x()+0.4, p.get_height()),
344             ha='center', va='top', color='white', size=18)
345 ax.set_title('Jumlah Pengguna Starbucks dari Tiap Klaster');
346
347 feature = ['male', 'age', 'income', 'frequency', 'monetary', 'tenure']
348
349 def plot_user_demographics(segment_df, segment, color,
350                             cols=['gender'] + feature[1:],
351                             segment_col='customer_cluster'):
352     """
353     Plot the distribution of user demographics and FMT features in a segment.
354     Args:
355         (1) segment_df (Pandas dataframe) - data with defined segments
356         (2) segment (str or int) - name of segment
357         (3) color (tuple(float, float, float) or str) - RGB tuple or named color for bar
358         (4) cols (list(str)) - names of cols to plot
359         (4) segment_col (str) - name of segment column
360     Returns: None.
361     """
362     # Extract cluster
363     cluster = segment_df.loc[segment_df[segment_col] == segment, cols]
364     # Histogram subplots
365     fig, ax = plt.subplots(2, 3, figsize=(16, 6))
366     fig.suptitle('Grafik Demografi untuk ' + segment.title(), fontsize='large')
367     for i in range(3):
368         sns.histplot(cluster[cols[i]], bins=20, color=color, ax=ax[0, i])
369         sns.histplot(cluster[cols[i + 3]], bins=20, color=color, ax=ax[1, i])
370     plt.show()
371
372 bronze_cust = fmt.query("customer_cluster == 'Bronze Customers'")
373 bronze_cust = bronze_cust[['gender', 'age', 'income', 'frequency', 'monetary', 'tenure']]
374 bronze_cust.describe()
375 # histogram untuk bronze customers
376 plot_user_demographics(fmt, 'Bronze Customers', sns.color_palette()[3])
377
378 silver_cust = fmt.query("customer_cluster == 'Silver Customers'")
379 silver_cust = silver_cust[['gender', 'age', 'income', 'frequency', 'monetary', 'tenure']]
380 silver_cust.describe()
381 # histogram untuk silver customers
382 plot_user_demographics(fmt, 'Silver Customers', sns.color_palette()[1])
383
384 gold_cust = fmt.query("customer_cluster == 'Gold Customers'")
385 gold_cust = gold_cust[['gender', 'age', 'income', 'frequency', 'monetary', 'tenure']]
386 gold_cust.describe()
387 # histogram untuk gold customers
388 plot_user_demographics(fmt, 'Gold Customers', sns.color_palette()[0])
389
390 diamond_cust = fmt.query("customer_cluster == 'Diamond Customers'")
391 diamond_cust = diamond_cust[['gender', 'age', 'income', 'frequency', 'monetary', 'tenure']]
392 diamond_cust.describe()
393 # histogram untuk diamond customer
394 plot_user_demographics(fmt, 'Diamond Customers', sns.color_palette()[2])

```