BAB 5 IMPLEMENTASI

5.1 Pre Processing Data

Berdasarkan metodologi pengerjaan skripsi pada bab 3, tahapan untuk *pre processing data*terdiri dari empat aktivitas yaitu *data selection*, *data cleansing* dan *data transformation*. Penjelasan yang lebih rinci akan dijelaskan pada setiap aktivitasnya.

5.1.1 *Data Cleansing* dan *Data Selection*

Data selection pada data mentah dilakukan untuk menyeleksi atribut yang diperlukan dalam proses clustering. Namun sebelum itu perlu dilakukan proses pengubahan nama variable untuk mempermudah proses *clustering* seperti yang telah dijelaskan pada bab 4 tabel 4.2. Selanjutnya dilakukan *data cleansing* terhadap data yang bernilai kosong dan data yang tidak konsisten. Kemudian memilih atribut untuk proses ekstraksi RFM.

5.1.1.1 *Data Cleansing*

Proses *data cleansing* untuk menghilangkan data yang bernilai kosong dan data yang tidak konsisten.pada data transaksi pelanggan. Penghapusan baris ini mengacu pada kode unik dari nomor faktur transaksi. Penghapusan baris data dengan nilai kosong yakni id_*customer* = “” yang terlihat pada Tabel 5.1 dilakukan menggunakan *SQL query* pada *Script* 5.1.

Tabel 5.1 Baris data yang memiliki nilai id_customer kosong

id_transaksi	id_customer	Tanggal	layanan	Pembayaran
FJ000473954		2017-05-02	KOLAM RENANG	105000
FJ000474024		2017-05-02	APOTEK	5500
FJ000474061		2017-05-02	APOTEK	5500
FJ000474090		2017-05-02	APOTEK	88000
FJ000474140		2017-05-02	APOTEK	95000

```
DELETE * FROM dump_transaksi WHERE id_customer = “”
```

Script 5.1 *Data Cleansing* id_customer

Sehingga dari 21513 baris data transaksi menjadi 21175 baris data transaksi. Sedangkan penghapusan baris data dengan nilai tidak konsisten yakni pembayaran = 1 dilakukan menggunakan *SQL query* pada *Script* 5.2. Sehingga dari 21175 baris data transaksi menjadi 19705 baris data transaksi.

```
DELETE FROM dump_transaksi WHERE pembayaran=1
```

Script 5.2 *Data Cleansing* pembayaran

5.1.1.2 *Data Selection*

Sebanyak 5 atribut yang berasal dari data dipilih secara manual. Atribut yang digunakan untuk proses selanjutnya adalah id_customer, tanggal, dan pembayaran. Penjelasan mengenai fungsi dari atribut yang telah dipilih terdapat pada Tabel 4.2.

5.1.2 Ekstraksi RFM

Ekstraksi RFM merupakan proses untuk mencari nilai atribut yaitu *recency*, *frequency* dan *monetary*. Tahapan ini memuat ekstraksi variabel RFM berdasarkan proses *data selection*. Proses ekstraksi menggunakan *phpmyadmin* berdasarkan *SQL query*

Nilai *recency* merupakan selisih antara waktu saat pengerjaan yaitu 07 November 2017 dengan waktu terakhir melakukan transaksi. Atribut yang dibutuhkan yaitu tanggal dan *id_customer*. Nilai *recency* didapatkan dari tanggal yang paling terakhir menggunakan fungsi MAX berdasarkan *id_customer* dengan fungsi GROUP BY, sedangkan untuk mengetahui tanggal saat ini menggunkan fungsi NOW dan mencari selisihnya menggunakan fungsi DATEDIFF.

Nilai *frequency* merupakan nilai yang menggambarkan berapa kali jumlah transaksi pelanggan. Atribut yang dibutuhkan yaitu *tanggal* dan *id_customer*. Nilai *frequency* didapatkan dari menghitung banyaknya tanggal transaksi dengan fungsi COUNT berdasarkan *id_customer* dengan fungsi GROUP BY.

Nilai *monetary* merupakan total biaya yang dikeluarkan *customer* untuk melakukan transaksi. Atribut yang dibutuhkan yaitu biaya dan *id_customer*. Nilai *monetary* didapatkan dari menjumlahkan biaya dengan fungsi SUM berdasarkan *id_customer* dengan fungsi GROUP BY.

Nilai *recency, frequency,* dan *monetary* ditampilkan secara urut menurut *id_customer* dengan fungsi ORDER BY. Secara lebih rinci berikut merupakan *query* yang digunakan untuk mencari nilai RFM untuk setiap *customer*.

```
INSERT INTO rfm(id_customer,Recency,Frequency,Monetery)

SELECT id_customer, DATEDIFF(DATENOW(),MAX(tanggal)), COUNT(
tanggal), SUM(pembayaran) FROM dump_transaksi GROUP BY
id_customer
```

Script 5.3 Pengubahan nilai ke dalam bentuk RFM

Setelah mengeksekusi *query* dan berhasil mendapatkan nilai atribut RFM pada setiap *id_customer*, jumlah baris data yang di dapatkan sebanyak 4716 baris. Data tersebut menunjukkan bahwa terdapat 4716 *customer* yang melakukan transaksi pada periode Mei 2017 hingga Oktober 2017.

5.1.3 Data Transformation

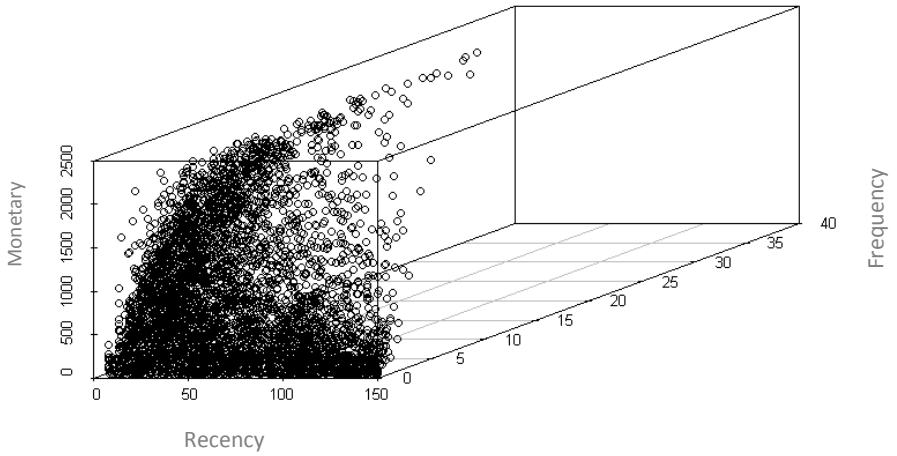
Data dinormalisasi menggunakan metode *Logarithmic*. Hal tersebut dikarenakan pada variabel RFM memiliki nilai yang tidak konsisten atau tidak normal dengan rentang yang berbeda-beda dan cukup jauh Oleh karena itu, nilainya diubah menggunakan fungsi log10(). Proses *Logarithmic* dilakukan dengan *Script* 5.4

```
> rtrans <- log10(recency)

> ftrans <- log10(frequency)

> mtrans <- log10(monetary)
```

Script 5.4 Transformasi data menggunakan metode *Logrithmic*

5.1.4 *Data Cleansing*

Data yang mengandung *outlier* seperti pada Gambar 5.1 dapat merusak hasil *clustering,* sehingga perlu dilakukan *data cleansing* terlebih dahulu.



Gambar 5.1 *Scatterplot* Sebelum *Cleansing*

Script 5.5 adalah *script* yang digunakan oleh Klodian Dhana (2016) dalam menghapus *outlier*.

```
outlierKD <- function(dt, var) {
 var_name <- eval(substitute(var),eval(dt))
 tot <- sum(!is.na(var_name))
  na1 <- sum(is.na(var_name))
  m1 <- mean(var_name, na.rm = T)
  par(mfrow=c(2, 2), oma=c(0,0,3,0))
  boxplot(var_name, main="With outliers")
  hist(var_name, main="With outliers", xlab=NA, ylab=NA)
  outlier <- boxplot.stats(var_name)$out
  mo <- mean(outlier)
  var_name <- ifelse(var_name %in% outlier, NA, var_name)
  boxplot(var_name, main="Without outliers")
  hist(var_name, main="Without outliers", xlab=NA, ylab=NA)
  title("Outlier Check", outer=TRUE)   na2 <- sum(is.na(var_name))
  m2 <- mean(var_name, na.rm = T)
  response <- readline(prompt="Do you want to remove outliers and to re
place with NA? [yes/no]: ")   if(response == "y" | response == "yes"){
```

```
    dt[as.character(substitute(var))] <- invisible(var_name)
      assign(as.character(as.list(match.call())$dt), dt, envir = .GlobalEnv)
    message("Outliers successfully removed", "\n")
    return(invisible(dt))
  } else{
    message("Nothing changed", "\n")
    return(invisible(var_name)) }}
outlierKD(dat,var)
```

Script 5.5 Penghapusan *Outlier*

5.1.5 Normalisasi Menggunakan *Min-Max*

Metode normalisasi yang selanjutnya digunakan dalam penelitian ini adalah Metode *Min-Max.* Metode ini mengubah nilai dari data pada setiap variabel menjadi 0 hingga 1. *Tools* R digunakan sebagai alat bantu dalamnormalisasi seperti pada *Script* 5.6.

```
>rnorm <- (routlier-min(routlier))/ (max(routlier)-min(routlier))

> fnorm <- (foutlier-min(foutlier))/ (max(foutlier)-min(foutlier))

> mnorm <- (moutlier-min(moutlier))/ (max(moutlier)-min(moutlier))
```

Script 5.6 Normalisasi *Min-Max*

Nilai dari *recency* memiliki makna terbalik dengan *Frequency* dan *Monetary*. Nilai *Recency* yang kecil merupakan *Recency* yang terbaik sedangkan nilai *Frequency* dan *Monetary* terkecil merupakan nilai terburuk. Oleh karena itu, berdasarkan penelitian Angelie, Annisa Veronica (2017) ketiga nilai dalam variabel tersebut disamakan terlebih dahulu dengan membalik nilai *Recency* setiap *customer*. Proses membalik ini dengan cara 1 dikurangi dengan nilai *recency* setiap *customer.* Hasil akhir dari pra proses secara keseluruhan terdapat pada Lampiran B.

5.2 Proses *Clustering*

Dalam proses *clustering* terdapat dua proses didalamnya yaitu menentukan nilai k dengan metode Elbow dan melakukan *clustering* dengan metode K-Means. Kedua proses tersebut dilakukan pada aplikasi Rstudio.

5.2.1 Penentuan Jumlah *Cluster*

Data set yang telah dilakukan *pre processing data* merupakan data yang telah siap untuk digunakan, selanjutnya dilakukan pemilihan jumlah *cluster* atau nilai k dengan menggunakan metode *Elbow*. Metode ini memilih jumlah *cluster* dengan melihat penurunan secara signifikan pada nilai SSE dan di titik itu nilai SSE sudah mulai stabil (tidak turun terlalu signifikan). Titik tersebut yang menjadi titik siku pada grafik.

Proses dari metode *Elbow* ini menggunakan percobaan jumlah *cluster* antara 1 sampai 10. Proses terakhir melihat hasil plot untuk mengetahui titik siku yang terbentuk. Rincian proses tersebut dijalankan menggunakan *Script* 5.7.

```
> rhasilelbow <- sapply(1:10,function(k){kmeans(rhasildataframereadybcrow
n, k, nstart=10 )$tot.withinss})

> plot(1:10, rhasilelbow,type="b", pch = 19, xlab="Number of clusters K",ylab
="Total within-clusters sum of squares ")
```

Script 5.7 Metode Elbow

5.2.2 *Clustering* dengan K-Means

Proses *clustering* dilakukan setelah mendapatkan nilai k. Proses ini dimulai dengan membuat inisiasi nama yakni r2cluster, inisiasi ini berisi fungsi untuk menjalankan K-Means pada RStudio dan memasukkan nilai k sebanyak 3. Hasilnya proses K-Means dapat ditampilkan dengan memanggil inisiasi tersebut. Untuk mendapatkan hasil yang lebih mudah dipahami, maka dilakukan penggabungan antara data dan hasil *clustering* menggunakan fungsi *data.frame* dan disimpan kedalam bentuk file dengan format csv menggunakan fungsi write.csv. Rincian proses tersebut dijalankan menggunakan *Script* 5.8.

```
>r3cluster<-kmeans(rhasildataframereadybcrown,3)

>r3cluster

> rhasil3cluster<-data.frame(rfulldataframereadybcrown,r3cluster$cluster)

> write.csv(rhasil2cluster, "hasil2cluster.csv")
```

Script 5.8 *Clustering* 2 Segmen Menggunakan K-Means

5.3 Uji Performa *Cluster*

Data hasil clustering dengan K-Mean yang pada mulanya membagi ke dalam 3 segmen perlu diuji peformanya untuk mengetahui apakah 3 segmen tersebut telah optimal menggunakan metode lainnya. Metode yang diterapkan dalam pengujian peforma yaitu metode SSE, *Connectivity, Dunn Index*, dan *Sillhoutte Width*.

5.3.1 Uji Performa Menggunakan SSE

Untuk melihat detail nilai SSE dari hasil metode *Elbow* dapat menggunakan *Script* 5.9

```
> rhasilelbow <- sapply(1:10,function(k){kmeans(rhasildataframereadybcrown, k, nstar
t=10 )$tot.withinss})

> View(rhasilelbow)
```

Script 5.9 *Script Menampilkan* rhasilelbow

5.3.2 Uji Performa Menggunakan ClValid()

Terdapat *package* dalam R CRAN yang bernama ClValid(). *Package* ini dapat digunakan untuk uji performa jumlah *cluster* dengan membandingkan masing-masing nilai k berdasarkan metode *dunn index, sillhoutte width* dan *connectivity*. *Script* yang digunakan untuk validasi jumlah *cluster* yakni terdapat pada *Script* 5.10.

```
> library(clValid)

> rclvalid<- clValid(rhasildataframereadybcrown, nClust= 2:15, clMethods="km
eans", validation="internal")

> summary(rclvalid)
```

Script 5.10 *Script* ClValid

5.4 Verifikasi *Cluster*

Untuk memastikan bahwa objek merupakan anggota dalam suatu *cluster,* maka dapat digunakan persamaan *Euclidian distance* dengan menghitung jarak data terhadap titik pusatnya. Dari perhitungan tersebut akan ditemukan titik *cluster* terdekat. *Script* yang digunakan dalam verifikasi *cluster* yakni pada *Script* 5.11.

```
#mengambil setiap baris centroid

> rcentroid2cluster1<-rdataframecentroid[1,]

> rcentroid2cluster2<-rdataframecentroid[2,]

> rcentroid2cluster3<-rdataframecentroid[3,]



#cari distance

>rdatadist3cluster$distcluster1<-sqrt(((rhasil3cluster$rbalik-rcentroid2cluster1
$recency)^2)+((rhasil3cluster$fnorm-rcentroid3cluster1$frequency)^2)+((rhasi
l3cluster$mnorm-rcentroid3cluster1$monetary)^2))

>rdatadist3cluster$distcluster2<-sqrt(((rhasil3cluster$rbalik-rcentroid3cluster2
$recency)^2)+((rhasil3cluster$fnorm-rcentroid3cluster2$frequency)^2)+((rhasi
l3cluster$mnorm-rcentroid3cluster2$monetary)^2))
```

```
>rdatadist3cluster$distcluster3<-sqrt(((rhasil3cluster$rbalik-rcentroid3cluster3
$recency)^2)+((rhasil3cluster$fnorm-rcentroid3cluster3$frequency)^2)+((rhasi
l3cluster$mnorm-rcentroid3cluster32$monetary)^2))

> View(rdatadist3cluster)

#menentukan nilai min dari jarak ke-3 cluster

>rdataframeeudist<-data.frame(rdatadist3cluster$distcluster1,rdatadist3clust
er$distcluster2, rdatadist3cluster$distcluster3)

> rdataframeeudist3$hasilcek<-apply(rdataframeeudist3,1,which.min)

> rdataframeeudist3$min<-apply(rdataframeeudist3,1,min)

#menggabungkan ke dalam satu tabel

> rhasildataeuclideandistance3<-rdataframeeudist3

> rhasildataeuclideandistance3$clusterke<-rhasil3cluster$r3cluster.cluster

> rhasildataeuclideandistance3$id_customer<-rhasil3cluster$id_custnorm

> View(rhasildataeuclideandistance3)
```

Script 5.11 *Script* Menunjukkan Jarak Terdekat