Klasifikasi Data

Tugas 1 | PD - A | 2019

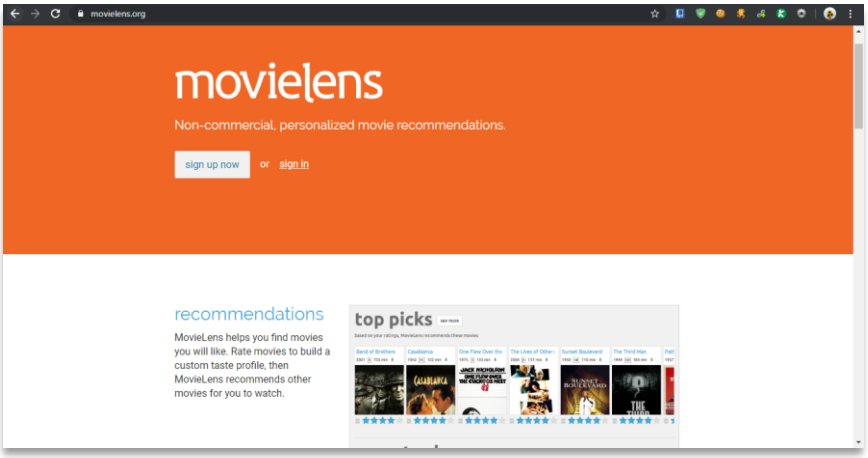
KELOMPOK 1

Zakaria Ahcmad	05211340000118
Firin Handayani	05211640000006
Humaira Nur Pradani	05211640000011

DAFTAR ISI

DAFTAR ISI

PENGENALAN DATASET



Dataset yang digunakan adalah *"MovieLens"* yang merupakan rangkaian data tentang film, pengguna dan rating yang ditujukan untuk film pada tahun 2000. Dalam dataset tersebut terdapat tiga file berbentuk *.dat* yaitu "ratings.dat", "users.dat" dan "movies.dat". Ketiga file tersebut dijelaskan lebih lanjut pada tabel berikut.

	ratings.dat	**users.dat**	**movies.dat**
Penjelasan	Rating untuk suatu film	Profil pengguna MovieLens	Film-film yang direview pada MovieLens
Jumlah record	1,000,209	6,040	3,900
Atribut	• UserID • MovieID • Rating • Timestamp	• UserID • Gender • Age • Occupation • Zip-Code	• MovieID • Title • Genre

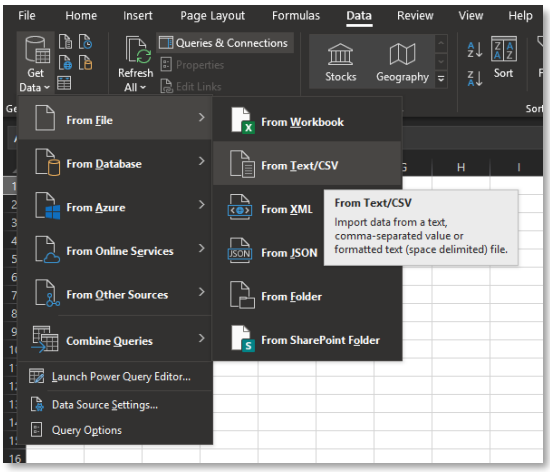
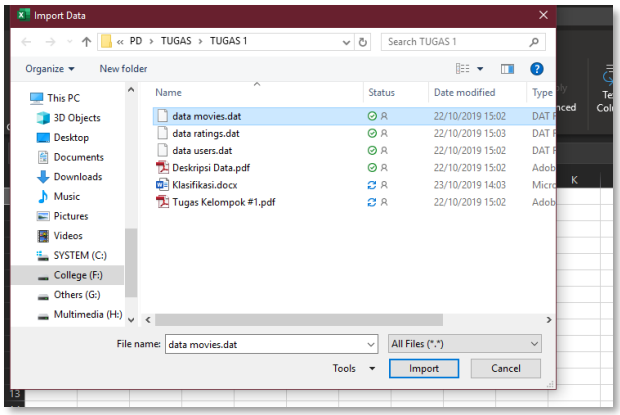
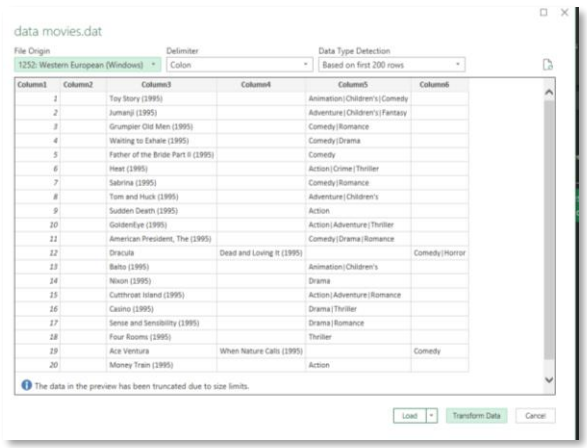
Dataset tersebut memiliki beberapa karakteristik sebagai berikut :

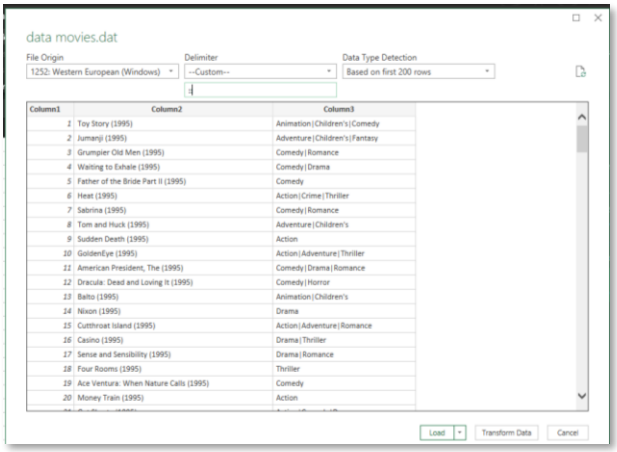
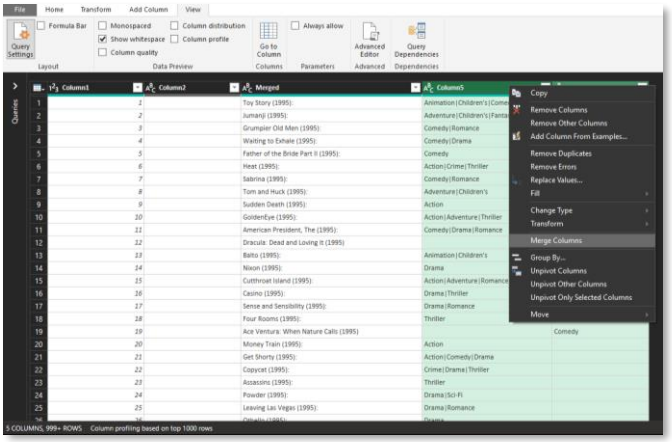
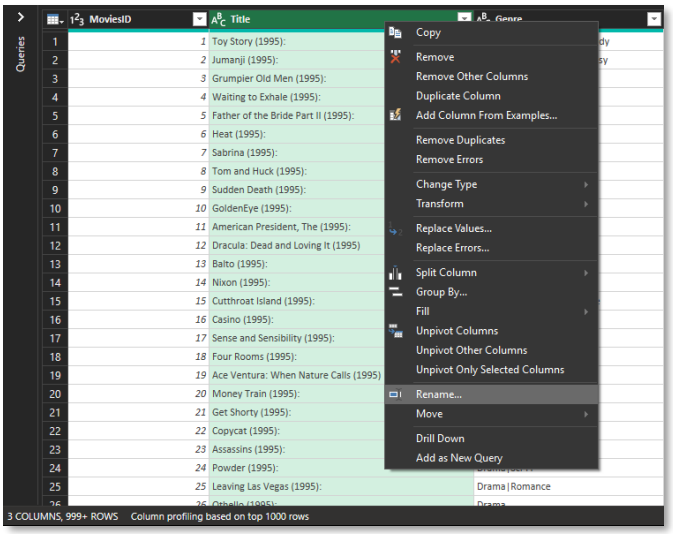
Karakteristik Dataset	Multivariate
Karakteristik Atribut	Numeric, Categorical
Associated Tasks	Classification
Area	Hiburan

PRE-STEP

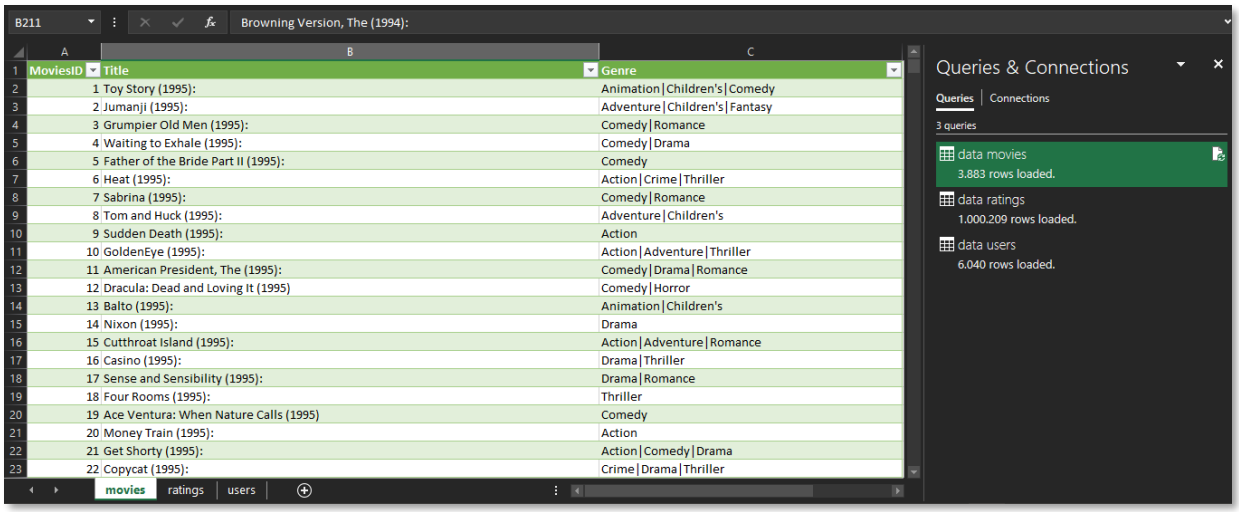
CONVERT FILE .dat MENJADI .csv

Pada proses awal perlu dilakukan convert data dari file .dat menjadi file .csv sehingga dapat dilakukan proses eksplorasi data dalam software R.

No.	Gambar	Langkah
1.		Langkah awal melakukan get data pada excel. Proses ini dilakukan untuk mengimport data .dat ke dalam excel dengan cara : • Klik Get Data → pilih From File • Kemudian pilih From Text/CSV
2.		Langkah kedua memilih file .dat yang ada di komputer → klik import
3.		Langkah ketiga melakukan transform data. Terdapat 2 cara : 1. Langsung klik “Transform Data”pad bagian pojok bawah. Pilihan ini langsung muncul ketika sudah melakukan import data. 2. Cara lain dengan memilih “Custom” pada Delimeter bagian atas kolom → menghapus satu tanda titik dua (:)

		
4.		Setelah klik “Transform Data” maka melakukan penggabungan kolom dengan cara pilih “Merge Columns”. • Proses ini dilakukan karena informasi yang harusnya berada di 1 kolom menjadi 2 kolom sehingga perlu dijadikan 1 kolom.
5.		Langkah selanjutnya melakukan perubahan nama pada setiap kolom di data yang di convert.

Dari hasil pengerjaan tersebut dihasilkan 3 tabel untuk masing-masing data. Dari tiga tabel yang terbentuk tersebut disimpan ke dalam 3 file csv yang berbeda yaitu movies.csv, ratings.csv, dan users.csv untuk selanjutnya diolah lebih lanjut pada RStudio.



	A	B	C
1	MoviesID	Title	Genre
2	1	Toy Story (1995):	Animation|Children's|Comedy
3	2	Jumanji (1995):	Adventure|Children's|Fantasy
4	3	Grumpier Old Men (1995):	Comedy|Romance
5	4	Waiting to Exhale (1995):	Comedy|Drama
6	5	Father of the Bride Part II (1995):	Comedy
7	6	Heat (1995):	Action|Crime|Thriller
8	7	Sabrina (1995):	Comedy|Romance
9	8	Tom and Huck (1995):	Adventure|Children's
10	9	Sudden Death (1995):	Action
11	10	GoldenEye (1995):	Action|Adventure|Thriller
12	11	American President, The (1995):	Comedy|Drama|Romance
13	12	Dracula: Dead and Loving It (1995)	Comedy|Horror
14	13	Balto (1995):	Animation|Children's
15	14	Nixon (1995):	Drama
16	15	Cutthroat Island (1995):	Action|Adventure|Romance
17	16	Casino (1995):	Drama|Thriller
18	17	Sense and Sensibility (1995):	Drama|Romance
19	18	Four Rooms (1995):	Thriller
20	19	Ace Ventura: When Nature Calls (1995)	Comedy
21	20	Money Train (1995):	Action
22	21	Get Shorty (1995):	Action|Comedy|Drama
23	22	Copycat (1995):	Crime|Drama|Thriller

BAB I : EKSPLORASI DATA

SUMMARIZATION DATA

PENJELASAN TIAP ATRIBUT

Dalam melakukan eskplorasi data, hal yang pertama dilakukan yaitu dengan mengetahui karakteristik dari tiap atribut dalam data frame tersebut. Selain itu, terdapat pencarian distinct value dimana hal ini digunakan untuk mencari nilai yang unik (tidak terdapat duplikat dalam atribut).

Syntax (Karakteristik Tiap Atribut)

```
#Cari distinct value data movies
length(unique(movies$MovieID))
length(unique(movies$Title))
length(unique(movies$Genre))

#Cari distinct value data ratings
length(unique(ratings$Rating))
length(unique(ratings$Timestamp))

#Cari distinct value data users
length(unique(users$UserID))
length(unique(users$Age))
length(unique(users$Zip.code))
length(unique(users$Occupation))
length(unique(users$Gender))
```

Hasil (Karakteristik Tiap Atribut)

```
> length(unique(movies$MovieID))
[1] 3883
> length(unique(movies$Title))
[1] 3883
> length(unique(movies$Genre))
[1] 301
> length(unique(ratings$Rating))
[1] 5
> length(unique(ratings$Timestamp))
[1] 458455
> length(unique(users$UserID))
[1] 6040
> length(unique(users$Age))
[1] 7
> length(unique(users$Zip.code))
[1] 3439
> length(unique(users$Occupation))
[1] 21
> length(unique(users$Gender))
[1] 2
```

Hasil dari pencarian distinct value dan penentuan karaktersitik setiap atribut terdapat dalam tabel di bawah ini.

Atribut	Distinct Value	Categorical/Numerical
UserID	6040	Numerik
MovieID	3883	Numerik
Rating	5	Kategorikal
Timestamp	458455	Numerik
Gender	2	Kategorikal
Age	7	Kategorikal
Occupation	21	Kategorikal
Zip-code	3439	Numerik
Title	3883	Kategorikal
Genre	301	Kategorikal

➔ Pada tabel di atas didapatkan 8 atribut dimana karakteristik digolongkan menjadi dua jenis yaitu numerik dan kategorikal.

➔ Pencarian distinct value dari setiap atribut dengan menggunakan fungsi length(unique(nama_data_frame$nama_kolom)). Penggunaan fungsi ini akan menemukan atribut yang unik dimana tidak ada duplikat atribut dalam data tersebut.

Pada tabel di bawah ini akan menampilkan penggunaan syntax dan hasil pencarian distinct value dan karakteristik dari tiap atribut dengan menggunakan software R.

STRUKTUR DATA

Hal selanjutnya setelah mengetahui karakteristik atribut dalam setiap data yaitu melihat struktur data. Dalam hal ini terdapat 3 data yaitu movies, users, dan ratings.

Penggunaan fungsi str(nama_data_frame) bertujuan untuk melihat tipe dan struktur dari setiap data frame. Selain itu, fungsi ini akan menampilkan jumlah baris, nama variabel, tipe variabel, dan sebagian baris pertama dari data. Di bawah ini merupakan tampilan penggunaan syntax dan hasil dari struktur data.

Syntax

```
#melihat struktur data
str(movies)
str(users)
str(ratings)
```

Hasil

```
> str(users)
'data.frame':   6040 obs. of  5 variables:
 $ UserID    : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Gender    : Factor w/ 2 levels "F","M": 1 2 2 2 2 1 2 2 2 1 ...
 $ Age       : int  1 56 25 45 25 50 35 25 25 35 ...
 $ Occupation: int  10 16 15 7 20 9 1 12 17 1 ...
 $ Zip.code  : Factor w/ 3439 levels "00231","00606",..: 1589 2249 1864 141 1939 1864 295 489 2107 3189
 ...
> str(ratings)
'data.frame':   1000209 obs. of  4 variables:
 $ UserID   : int  1 1 1 1 1 1 1 1 1 1 ...
 $ MovieID  : int  1193 661 914 3408 2355 1197 1287 2804 594 919 ...
 $ Rating   : int  5 3 3 4 5 3 5 5 4 4 ...
 $ Timestamp: int  978300760 978302109 978301968 978300275 978824291 978302268 978302039 978300719 978302
268 978301368 ...
```

DIMENSI

Dalam menampilkan jumlah baris dan atribut yang terdapat dalam 3 data frame maka dapat digunakan fungsi dim(nama_data_frame). Atribut dapat diartikan sebagai kolom karena setiap kolom mewaikili dari atribut yang ada dalam data frame. Berikut ini merupakan penerapan syntax dan hasilnya dalam menampilkan jumlah baris dan kolom.

Syntax

```
#melihat dimensi data antara baris dan kolomnya
dim(movies)
dim(users)
dim(ratings)
```

Hasil

```
> dim(movies)
[1] 3883    3
> dim(users)
[1] 6040    5
> dim(ratings)
[1] 1000209       4
```


HEAD & TAIL

Fungsi Head & Tail digunakan untuk menampilkan data. Head bertujuan untuk menampilkan data teratas dari suatu frame dimana biasanya data yang ditampilkan berjumlah 6 (n=6). Tail bertujuan untuk menampilkan data terbawah dimana konsepnya sama dengan Head namun hanya berbanding terbalik saja. Data yang ditampilkan berupa 6 baris teratas dan terbawah dari semua atribut yang ada dalam data tersebut.

Penerapan fungsi head & tail → head(nama_data_frame) sedangkan untuk tail yaitu tail(nama_data_frame). Di bawah ini penerapan syntax untuk 3 data frame dan output yang dihasilkan.

Syntax

```
#head (meihat 6 data teratas) dan tail (melihat 6 data terbawah)
head(movies)
head(users)
head(ratings)

tail(movies)
tail(users)
tail(ratings)
```

Hasil

```
> head(movies)
  MovieID                               Title                        Genre
1       1                    Toy Story (1995)  Animation|Children's|Comedy
2       2                      Jumanji (1995) Adventure|Children's|Fantasy
3       3             Grumpier Old Men (1995)                Comedy|Romance
4       4            Waiting to Exhale (1995)                  Comedy|Drama
5       5  Father of the Bride Part II (1995)                        Comedy
6       6                           Heat (1995)         Action|Crime|Thriller
> head(users)
  UserID Gender Age Occupation Zip.code
1      1      F   1         10    48067
2      2      M  56         16    70072
3      3      M  25         15    55117
4      4      M  45          7    02460
5      5      M  25         20    55455
6      6      F  50          9    55117

> head(ratings)
  UserID MovieID Rating Timestamp
1      1    1193      5 978300760
2      1     661      3 978302109
3      1     914      3 978301968
4      1    3408      4 978300275
5      1    2355      5 978824291
6      1    1197      3 978302268
> tail(movies)
     MovieID                          Title          Genre
3878    3947            Get Carter (1971)       Thriller
3879    3948     Meet the Parents (2000)         Comedy
3880    3949 Requiem for a Dream (2000)          Drama
3881    3950             Tigerland (2000)          Drama
3882    3951     Two Family House (2000)          Drama
3883    3952       Contender, The (2000) Drama|Thriller
> tail(users)
     UserID Gender Age Occupation Zip.code
6035   6035      F  25          1    78734
6036   6036      F  25         15    32603
6037   6037      F  45          1    76006
6038   6038      F  56          1    14706
6039   6039      F  45          0    01060
6040   6040      M  25          6    11106
> tail(ratings)
        UserID MovieID Rating Timestamp
1000204   6040    1090      3 956715518
1000205   6040    1091      1 956716541
1000206   6040    1094      5 956704887
1000207   6040     562      5 956704746
1000208   6040    1096      4 956715648
1000209   6040    1097      4 956715569
```


SUMMARY

Dalam menampilkan ringkasan dari suatu data frame dapat menggunakan fungsi summary. Fungsi ini bertujuan untuk menampilkan hasil dari beberapa nilai statistik setiap atribut yang ada di data frame tersebut. Nilai tersebut berupa nilai minimum (Min), nilai quantil pertama (1st Qu.), Nilai tengah (Median), nilai Quantil ketiga (3rd Qu.), dan nilai maksimum.

Penerapan fungsi summary → summary(nama_data_frame).

Di bawah ini merupakan syntax summary dan hasil untuk 3 data frame.

Syntax

```
#summary -melihat ringkasan data
summary(movies)
summary(users)
summary(ratings)
```

Hasil

```
> summary(movies)
    MovieID                                   Title                 Genre
 Min.   :   1.0   $1,000,000 Duck (1971)          :   1   Drama         : 843
 1st Qu.: 982.5   ...And Justice for All (1979)    :   1   Comedy        : 521
 Median :2010.0   1-900 (1994)                    :   1   Horror        : 178
 Mean   :1986.0   10 Things I Hate About You (1999):   1   Comedy|Drama  : 162
 3rd Qu.:2980.5   101 Dalmatians (1961)           :   1   Comedy|Romance: 142
 Max.   :3952.0   101 Dalmatians (1996)           :   1   Drama|Romance : 134
                  (Other)                         :3877   (Other)       :1903
> summary(users)
     UserID      Gender         Age          Occupation        Zip.code
 Min.   :   1   F:1709   Min.   : 1.00   Min.   : 0.000   48104  :  19
 1st Qu.:1511   M:4331   1st Qu.:25.00   1st Qu.: 3.000   22903  :  18
 Median :3020            Median :25.00   Median : 7.000   55104  :  17
 Mean   :3020            Mean   :30.64   Mean   : 8.147   94110  :  17
 3rd Qu.:4530            3rd Qu.:35.00   3rd Qu.:14.000   10025  :  16
 Max.   :6040            Max.   :56.00   Max.   :20.000   55105  :  16
                                                          (Other):5937
> summary(ratings)
     UserID        MovieID         Rating        Timestamp
 Min.   :   1   Min.   :   1   Min.   :1.000   Min.   :9.567e+08
 1st Qu.:1506   1st Qu.:1030   1st Qu.:3.000   1st Qu.:9.653e+08
 Median :3070   Median :1835   Median :4.000   Median :9.730e+08
 Mean   :3025   Mean   :1866   Mean   :3.582   Mean   :9.722e+08
 3rd Qu.:4476   3rd Qu.:2770   3rd Qu.:4.000   3rd Qu.:9.752e+08
 Max.   :6040   Max.   :3952   Max.   :5.000   Max.   :1.046e+09
```

DESCRIBE

Fungsi describe hampir sama dengan summary dimana berguna untuk menampilkan ringkasan dari suatu data frame. Perbedaan dari keduanya yaitu output yang dihasilkan lebih lengkap pada penerapan fungsi describe apalagi untuk data numerik.

Terdapat tampilan informasi missing yang berarti ada tidaknya suatu nilai yang tidak terbaca atau tidak mempunyai nilai (missing value). Selain itu, penggunaan fungsi ini juga dapat menampilkan distinct dimana nilai unik dari setiap atribut dalam data frame. Penerapan sama dimana hanya menuliskan nama dari data frame yang akan ditampilkan. Di bawah ini merupakan syntax describe dan hasil untuk 3 data frame.

Syntax

```
#melakukan pendeskripsian data
#:lebih lengkap outputnya drpd summary
describe(movies)
describe(users)
describe(ratings)
```


Hasil

```
> describe(movies)
movies

 3  Variables      3883  Observations
--------------------------------------------------------------------------------
MovieID
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90
    3883        0     3883        1     1986     1324    196.1    392.2    982.5   2010.0   2980.5   3562.8
     .95
  3756.9

lowest :    1    2    3    4    5, highest: 3948 3949 3950 3951 3952
--------------------------------------------------------------------------------
Title
       n  missing distinct
    3883        0     3883

lowest : $1,000,000 Duck (1971)                  ...And Justice for All (1979)             1-900 (1994)
 10 Things I Hate About You (1999)         101 Dalmatians (1961)
highest: Zed & Two Noughts, A (1985)               Zero Effect (1998)                        Zero Kelvin (Kjærlighetens kjøtere) (1995)
 Zeus and Roxanne (1997)                   Zone 39 (1997)
--------------------------------------------------------------------------------
Genre
       n  missing distinct
    3883        0      301

lowest : Action                                    Action|Adventure                          Action|Adventure|Animation
         Action|Adventure|Animation|Children's|Fantasy Action|Adventure|Animation|Horror|Sci-Fi
highest: Sci-Fi|Thriller|War                       Sci-Fi|War                                Thriller
         War                                       Western
--------------------------------------------------------------------------------
> describe(users)
users

 5  Variables      6040  Observations
--------------------------------------------------------------------------------
UserID
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90
    6040        0     6040        1     3020     2014    302.9    604.9   1510.8   3020.5   4530.2   5436.1
     .95
  5738.1

lowest :    1    2    3    4    5, highest: 6036 6037 6038 6039 6040
Gender
       n  missing distinct
    6040        0        2

Value          F     M
Frequency   1709  4331
Proportion 0.283 0.717
--------------------------------------------------------------------------------
Age
       n  missing distinct     Info     Mean      Gmd
    6040        0        7    0.943    30.64    14.07

Value          1    18    25    35    45    50    56
Frequency    222  1103  2096  1193   550   496   380
Proportion 0.037 0.183 0.347 0.198 0.091 0.082 0.063
--------------------------------------------------------------------------------
Occupation
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90
    6040        0       21    0.993    8.147    7.208        0        0        3        7       14       17
     .95
      19

lowest :  0  1  2  3  4, highest: 16 17 18 19 20
--------------------------------------------------------------------------------
Zip.code
       n  missing distinct
    6040        0     3439

lowest : 00231 00606 00681 00693 00918, highest: 99703 99709 99801 99826 99945
> describe(ratings)
ratings

 4  Variables      1000209  Observations
--------------------------------------------------------------------------------
UserID
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90
 1000209        0     6040        1     3025     1995      331      669     1506     3070     4476     5443
     .95
    5740

lowest :    1    2    3    4    5, highest: 6036 6037 6038 6039 6040
--------------------------------------------------------------------------------
MovieID
       n  missing distinct     Info     Mean      Gmd      .05      .10      .25      .50      .75      .90
 1000209        0     3706        1     1866     1263      172      357     1030     1835     2770     3430
     .95
    3675

lowest :    1    2    3    4    5, highest: 3948 3949 3950 3951 3952
--------------------------------------------------------------------------------
Rating
       n  missing distinct     Info     Mean      Gmd
 1000209        0        5    0.927    3.582    1.219

Value          1      2      3      4      5
Frequency   56174 107557 261197 348971 226310
Proportion  0.056  0.108  0.261  0.349  0.226
--------------------------------------------------------------------------------
Timestamp
        n   missing  distinct      Info      Mean       Gmd       .05       .10       .25       .50       .75
  1000209         0    458455         1 972243695  11025423 958704091 960681570 965302637 973018006 975220939
      .90       .95
978133376 993074153

lowest :  956703932  956703954  956703977  956704056  956704081, highest: 1046454320 1046454338 1046454443 1046454548 1046454590
--------------------------------------------------------------------------------
```

DISTRIBUSI KELAS

Distibusi kelas bertujuan untuk melihat distribusi maupun presentasi dari setiap kelas suatu data frame. Dalam tabel di bawah ini hanya menampilkan salah satu contoh penerapan distribusi kelas untuk data frame ratings dengan kolom Rating.

Di bawah ini merupakan syntax dan hasil dari distribusi kelas untuk kolom Rating dari data frame ratings. Pada Rating terdapat angka 1-5 dimana dengan melihat distribusi kelas maka 5 angka ditampilkan. Selain itu terdapat freq dimana jumlah dari setiap rating sedangkan percentage digunakan untuk menampilkan dalam bentuk persentase.

Syntax

```
#distribution class
y <- ratings$Rating
cbind(freq=table(y), percentage=prop.table(table(y))*100)
```

Hasil

```
> y <- ratings$Rating
> cbind(freq=table(y), percentage=prop.table(table(y))*100)
     freq percentage
1  56174   5.616226
2 107557  10.753453
3 261197  26.114242
4 348971  34.889808
5 226310  22.626271
```

KORELASI

Korelasi dilakukan untuk melihat nilai keterkaitan dan hubungan linear dari satu vaiabel dengan variabel lainnya. Penerapan korelasi dapat dilakukan dengan menggunakan syntax : Correlations <- cor (nama_data_frame$nama_kolom_atribut1, nama_data_frame$nama_kolom_atribut2).

Pada tabel di bawah ini terdapat 2 jenis penerapan yang dilakukan yaitu :

a. Korelasi dilakukan antara Age dan Rating dengan tujuan untuk mengetahui nilai korelasi yang dihasilkan dari kedua atribut tersebut.
b. Korelasi dilakukan untuk semua atribut dari data yang sudah digabungkan. Data yang digabung merupakan data movies, ratings, dan users. Tujuannya untuk melihat korelasi dari setiap atribut yang ada dalam 3 data frame tersebut. Dapat dilihat dari hasil tabel korelasi, bahwa nilai Rating paling diperngaruhi oleh Drama, dengan nilai korelasi sebesar 0,12.

Syntax

a. Syntax korelasi antara Age dan Rating

```
#correlation antara Age dengan Rating
correlations <- cor(merge2$Age,merge2$Rating)
print(correlations)
```

b. Syntax korelasi semua atribut

```
#korealasi semua atribut dari data siap train dan test
cor(data_dropf)
```

Hasil

a. Hasil korelasi antara Age dan Rating

```
> correlations <- cor(merge2$Age,merge2$Rating)
> print(correlations)
[1] 0.05686866
```


b. Hasil korelasi semua atribut

```
> cor(data_dropf)
                Action    Adventure    Animation   Children's       Comedy        Crime  Documentary        Drama
Action      1.00000000  0.374961275 -0.110293594 -0.141313530 -0.268092179  0.088518911 -0.0525650888 -0.20241469
Adventure   0.37496127  1.000000000  0.004731502  0.098282997 -0.124959510 -0.045924488 -0.0351091293 -0.19456962
Animation  -0.11029359  0.004731502  1.000000000  0.576204235  0.018543896 -0.062519563 -0.0189906092 -0.15447931
Children's -0.14131353  0.098282997  0.576204235  1.000000000  0.058710833 -0.081976830 -0.0249008449 -0.13570728
Comedy     -0.26809218 -0.124959510  0.018543896  0.058710833  1.000000000 -0.078030056 -0.0406974155 -0.24984014
Crime       0.08851891 -0.045924488 -0.062519563 -0.081976830 -0.078030056  1.000000000 -0.0262428479  0.07047923
Documentary -0.05256509 -0.035109129 -0.018990609 -0.024900845 -0.040697416 -0.026242848  1.0000000000 -0.06219407
Drama      -0.20241469 -0.194569617 -0.154479314 -0.135707275 -0.249840144  0.070479234 -0.0621940724  1.00000000
Fantasy     0.01455086  0.227046280  0.012025289  0.263279950 -0.006010461 -0.033744836 -0.0173264052 -0.09692859
Film-Noir  -0.08028760 -0.014178284  0.037012966 -0.038033400 -0.101425006  0.136237043 -0.0121754493 -0.06729730
Horror     -0.04273263 -0.057255808 -0.049730125 -0.077098692 -0.093064194 -0.047898576 -0.0256731681 -0.18955128
Musical    -0.10043235 -0.022327105  0.335230516  0.312566967  0.030565515 -0.061179331 -0.0071547234 -0.09477807
Mystery    -0.05408376 -0.043503439 -0.042487575 -0.052785846 -0.105345795  0.080093123 -0.0182649503 -0.02768906
Romance    -0.06782984 -0.024389480 -0.054540291 -0.084550254  0.112843082 -0.073320437 -0.0371366108  0.02355211
Sci-Fi      0.31911668  0.284190076 -0.055525650 -0.038844023 -0.187079398 -0.083730109 -0.0385683582 -0.21274652
Thriller    0.20275577 -0.038422848 -0.085713442 -0.132642475 -0.299501142  0.115095348 -0.0431910196 -0.15371668
War         0.13587216  0.016646681 -0.046114100 -0.066538801 -0.127101205 -0.079715175 -0.0160820340  0.13658185
Western     0.02224224 -0.011964179 -0.030907987 -0.031268978  0.007926703 -0.042711299 -0.0129737566 -0.04594463
Rating     -0.04763300 -0.036717753  0.019669601 -0.039828701 -0.039621793  0.033446074  0.0280977547  0.12256061
Age        -0.03097502 -0.016729835 -0.047020477 -0.052857863 -0.044046283 -0.007930635  0.0044065344  0.06385614
Occupation  0.01834690  0.014309332 -0.003833788 -0.006905585 -0.006149114  0.002821292 -0.0026887959 -0.01232587
F          -0.09438046 -0.038644670  0.017719437  0.031661567  0.040757848 -0.027064769 -0.0002339954  0.05239027
M           0.09438046  0.038644670 -0.017719437 -0.031661567 -0.040757848  0.027064769  0.0002339954 -0.05239027
                Fantasy    Film-Noir       Horror      Musical      Mystery      Romance       Sci-Fi     Thriller
Action      0.014550861 -0.080287599 -0.042732627 -0.100432354 -0.054083757 -0.067829840  0.31911668  0.202755773
Adventure   0.227046280 -0.014178284 -0.057255808 -0.022327105 -0.043503439 -0.024389480  0.28419008 -0.038422848
Animation   0.012025289  0.037012966 -0.049730125  0.335230516 -0.042487575 -0.054540291 -0.05552565 -0.085713442
Children's  0.263279950 -0.038033400 -0.077098692  0.312566967 -0.052785846 -0.084550254 -0.03884402 -0.132642475
Comedy     -0.006010461 -0.101425006 -0.093064194  0.030565515 -0.105345795  0.112843082 -0.18707940 -0.299501142
Crime      -0.033744836  0.136237043 -0.047898576 -0.061179331  0.080093123 -0.073320437 -0.08373011  0.115095348
Documentary -0.017326405 -0.012175449 -0.025673168 -0.007154723 -0.018264950 -0.037136611 -0.03856836 -0.043191020
Drama      -0.096928585 -0.067297296 -0.189551277 -0.094778073 -0.027689056  0.023552106 -0.21274652 -0.153716682
Fantasy     1.000000000 -0.026464247 -0.055802544 -0.020134344 -0.039700231 -0.014821524  0.12184300 -0.087374055
Film-Noir  -0.026464247  1.000000000 -0.039156806 -0.028384337  0.215353937 -0.047350504 -0.00405642  0.115231480
Horror     -0.055802544 -0.039156806  1.000000000 -0.018923632 -0.002423171 -0.099433985  0.05650515  0.056628521
Musical    -0.020134344 -0.028384337 -0.018923632  1.000000000 -0.042580647  0.023506079 -0.06801167 -0.100690203
Mystery    -0.039700231  0.215353937 -0.002423171 -0.042580647  1.0000000000 -0.040161764 -0.02827264  0.225280680
Romance    -0.014821524 -0.047350504 -0.099433985  0.023506079 -0.040161764  1.000000000 -0.13375237 -0.081383797
Sci-Fi      0.121843002 -0.004056420  0.056505154 -0.068011672 -0.028272641 -0.133752371  1.00000000  0.102546497
Thriller   -0.087374055  0.115231480  0.056628521 -0.100690203  0.225280679 -0.081383797  0.10254650  1.000000000
War        -0.044927792 -0.036984099 -0.077984718 -0.034429236 -0.055481543  0.053347206  0.03931430 -0.088018109
Western    -0.028199427 -0.019816037 -0.041784122 -0.030245412 -0.029726947 -0.044650330 -0.01093526 -0.058896888
Rating     -0.023312110  0.060258867 -0.094352764  0.015642722  0.015847592  0.009643728 -0.04448693 -0.004806119
Age        -0.024222079  0.033495075 -0.023901469  0.005157612  0.024307840 -0.017503470 -0.01087860 -0.014099894
Occupation  0.001299080  0.005245927  0.001438909 -0.007311561  0.002421403  0.017503470 -0.02625029  0.008981231
F          -0.002805652 -0.005152347 -0.036565863  0.038050735  0.000905152  0.091271577 -0.07237220 -0.038038848
M           0.002805652  0.005152347  0.036565863 -0.038050735 -0.000905152 -0.091271577  0.07237220  0.038038848
                   War      Western       Rating          Age   Occupation            F            M
Action      0.13587216  0.022242238 -0.047632996 -0.030975018  0.018346901 -0.0943804575  0.0943804575
Adventure   0.01664668 -0.011964179 -0.036717753 -0.016729835  0.014309332 -0.0386446695  0.0386446695
Animation  -0.04611410 -0.030907987  0.019669601 -0.047020477 -0.003833788  0.0177194371 -0.0177194371
Children's -0.06653880 -0.031268978 -0.039828701 -0.052857863 -0.006905585  0.0316615669 -0.0316615669
Comedy     -0.12710121  0.007926703 -0.039621793 -0.044046283 -0.006149114  0.0407578476 -0.0407578476
Crime      -0.07971518 -0.042711299  0.033446074 -0.007930635  0.002821292 -0.0270647691  0.0270647691
Documentary -0.01608203 -0.012973757  0.028097755  0.004406534 -0.002688796 -0.0002339954  0.0002339954
Drama       0.13658185 -0.045944627  0.122560607  0.063856141 -0.012325869  0.0523902724 -0.0523902724
Fantasy    -0.04492779 -0.028199427 -0.023312110 -0.024222079  0.001299080 -0.0028056521  0.0028056521
Film-Noir  -0.03698410 -0.019816037  0.060258867  0.033495075  0.005245927 -0.0051523472  0.0051523472
Horror     -0.07798472 -0.041784122 -0.094352764 -0.023901469  0.001438909 -0.0365658631  0.0365658631
Musical    -0.03442924 -0.030245412  0.015642722  0.005157612 -0.007311561  0.0380507353 -0.0380507353
Mystery    -0.05548154 -0.029726947  0.015847592  0.024307840  0.002421403  0.0009051552 -0.0009051552
Romance     0.05334721 -0.044650330  0.009643728  0.017503470 -0.014018004  0.0912715766 -0.0912715766
Sci-Fi      0.03931430 -0.010935261 -0.044486931 -0.010878601  0.026250290 -0.0723721992  0.0723721992
Thriller   -0.08801811 -0.058896888 -0.004806119 -0.014099894  0.008981231 -0.0380388477  0.0380388477
Thriller   -0.08801811 -0.058896888 -0.004806119 -0.014099894  0.008981231 -0.0380388477  0.0380388477
War         1.00000000 -0.019802519  0.075688170  0.038445732  0.010264437 -0.0256361256  0.0256361256
Western    -0.01980252  1.000000000  0.007311165  0.038176803  0.005924365 -0.0263974792  0.0263974792
Rating      0.07568817  0.007311165  1.000000000  0.056868662  0.006752932  0.0198606330 -0.0198606330
Age         0.03844573  0.038176803  0.056868662  1.000000000  0.078370569  0.0031888929 -0.0031888929
Occupation  0.01026444  0.005924365  0.006752932  0.078370569  1.000000000 -0.1149738374  0.1149738374
F          -0.02563613 -0.026397479  0.019860633  0.003188893 -0.114973837  1.0000000000 -1.0000000000
M           0.02563613  0.026397479 -0.019860633 -0.003188893  0.114973837 -1.0000000000  1.0000000000
> |
```

STANDARD DEVIASI

Dalam mencari nilai standard deviasi dari suatu atribut yang terdapat pada data frame maka digunakan fungsi sapply(). Di bawah ini merupakan syntax dan hasil dari standard deviasi salah satu atribut yaitu Age yang diambil dari data frame merge2.

Penerapan fungsi standard deviasi → standev = nama_data_frame$nama_kolom_atribut

Syntax

```
#Standard Deviasi
standev = merge2$Age
sd(standev)
```

Hasil

```
> standev = merge2$Age
> sd(standev)
[1] 11.75198
```

Dari hasil tersebut didapatkan bahwa standar deviasi dari umur pengguna MovieLens adalah sebesar 11,75.

VISUALISASI DATA

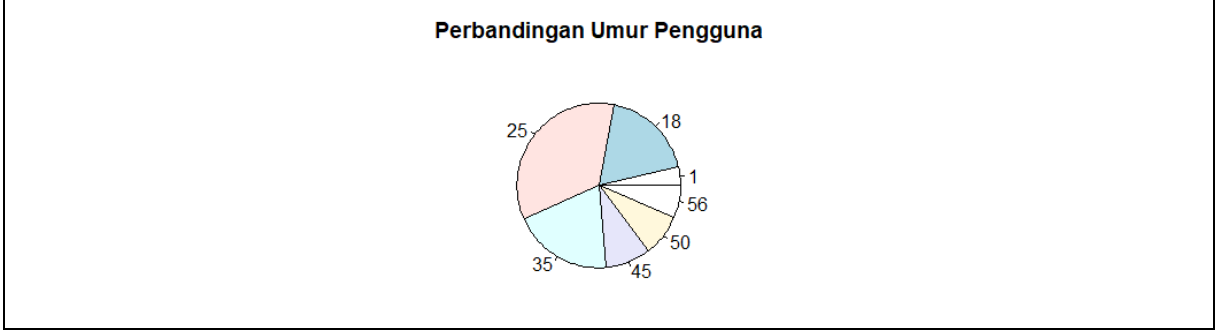
DATA USER

- **Perbandingan Umur Pengguna**

Syntax

```
#user -> umur berapa
pie(table(users$Age), main = "Perbandingan Umur Pengguna")
```

Hasil



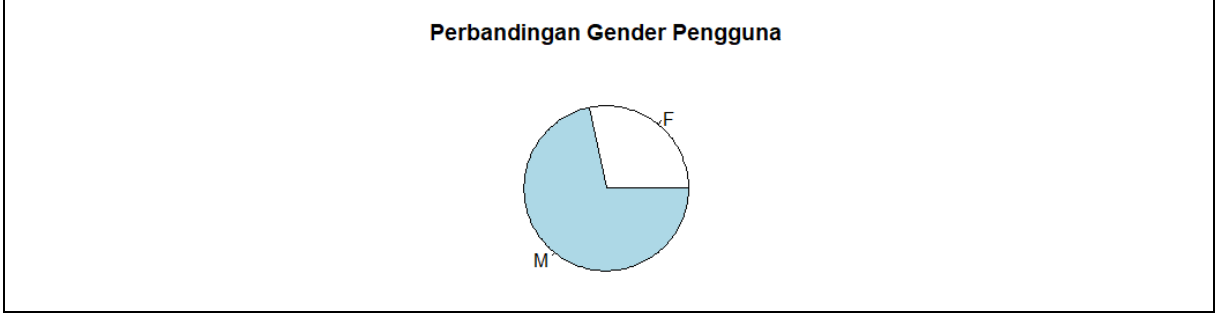
Dari pie chart yang telah terbentuk, terlihat bahwa umur pengguna MovieLens yang paling banyak yaitu pada kategori 25 yaitu dengan range umur 25-34 tahun.

- **Perbandingan Jenis Kelamin Pengguna**

Syntax

```
#user -> gender
pie(table(users$Gender), main = "Perbandingan Gender Pengguna")
```

Hasil



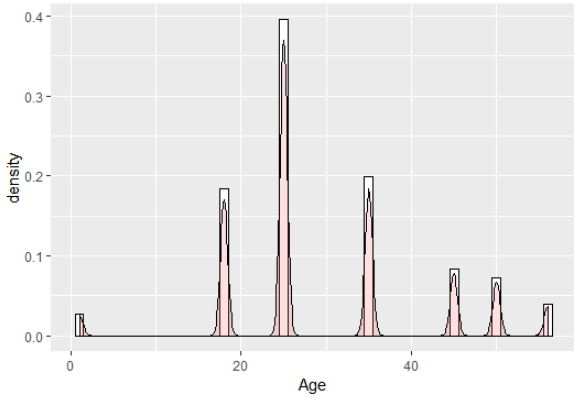
Dapat dilihat dari data tersebut bahwa jenis kelamin pengguna MovieLens yang terbanyak adalah laki-laki.

- **Distribusi Umur**

Syntax

```
mlDat_user <- ddply(merge3, ~UserID + Age + Gender + Occupation, summarize,
                    mean_rating = mean(Rating))
agePlot <- ggplot(merge3, aes(Age)) + geom_histogram(aes(y = ..density..),
                                                     binwidth = 1, colour = "black", fill = "white")
agePlot <- agePlot + geom_density(alpha = 0.2, fill = "#FF6666")
print(agePlot)
```

Hasil



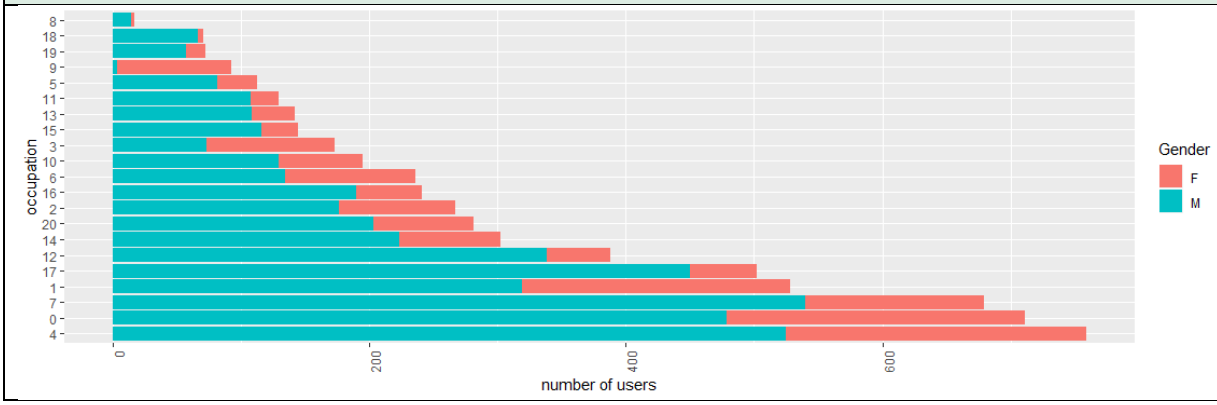
Graifk diatas menunjukkan distribusi dari umur pengguna MovieLens. Dapat dilihat bahwa rentang 20-40 memiliki nilai yang paling tinggi.

- **Distribusi Gender Menurut Pekerjaan Pengguna MovieLens**

Syntax

```
# urutkan data
userPlot <- ggplot(users, aes(x = reorder(Occupation, Occupation, function(x) -length(x)),
                              fill = Gender)) + geom_bar()
# buat axis
userPlot <- userPlot + theme(axis.text.x = element_text(angle = 90, hjust = 1))
userPlot <- userPlot + ylab("jumlah pengguna") + xlab("occupation")
# flip axis
userPlot <- userPlot + coord_flip()
userPlot
```

Hasil



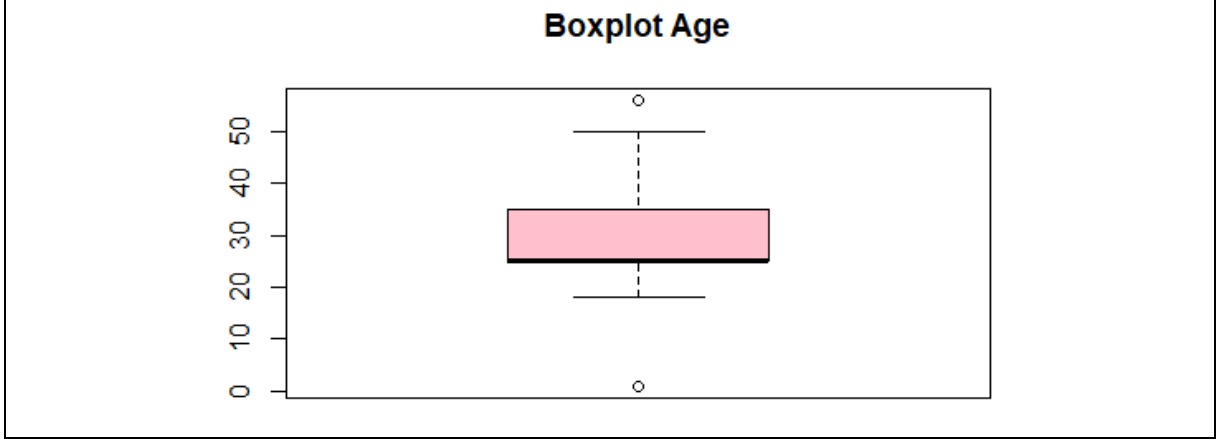
Dari hasil visualisasi diatas dapat dilihat persebaran pekerjaan menurut jenis kelamin untuk pengguna movieLens. Pekerjaan yang paling sedikit adalah berkode 8 yaitu farmer, dan yang paling banyak adalah pekerjaan berkode 4 yaitu college/grad student

- **Boxplot Umur Pengguna**

Syntax

```
boxplot(users$Age, col="pink", main = "Boxplot Age")
```

Hasil



Boxplot merupakan ringkasan distribusi sampel yang disajikan secara grafis yang bisa menggambarkan bentuk distribusi data (skewness), ukuran tendensi sentral dan ukuran penyebaran (keragaman) data pengamatan. Visualisasi boxplot diatas menggambarkan bahwa data age pada user memiliki nilai median antara 20-30. Kemudian batasan minimum pada data tersebut ada pada nilai 20, sedangkan nilai maksimum pada 50. Dua titik yang ad diatas dan bawah menggambarkan nilai yang sifatnya *outlier* atau berbeda dengan mayoritas data.

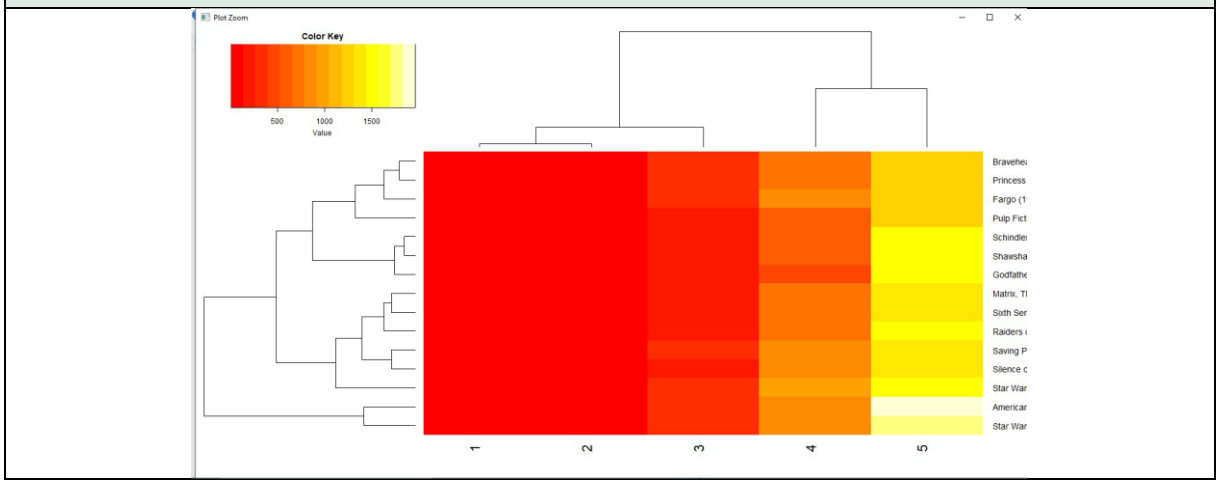
DATA MOVIES

- **Heatmap Rating terhadap Judul Movie**

Syntax

```
eksplorMovie <- merge(movi3s, ratings, by.x = 'MovieID', by.y = 'MovieID')
eksplorMovie <- merge(eksplorMovie, users, by.x = 'UserID', by.y = 'UserID')


coba <- select(eksplorMovie, Title, Rating)


dadi <- count(coba, c("Title", "Rating"))
View(dadi)

library(plyr)
library(dplyr)
library(tidyr)
library("gplots")


dadi <- dadi %>%
  spread(Rating, freq)

row.names(dadi) <- dadi[,1]

dadi <- select(dadi, 2, 3, 4, 5, 6)
dadi[is.na(dadi)] <- 0
dadislice <- dadi %>% top_n(15)
row.names(dadislice) <- dadislice[,1]
dadislice <- select(dadislice, 2, 3, 4, 5, 6)
dadislice <- select(dadislice, 5, 4, 3, 2, 1)

mdadi <- as.matrix(dadislice)

heatmap.2(mdadi, scale = "none",
          trace = "none", density.info = "none")
```

Hasil



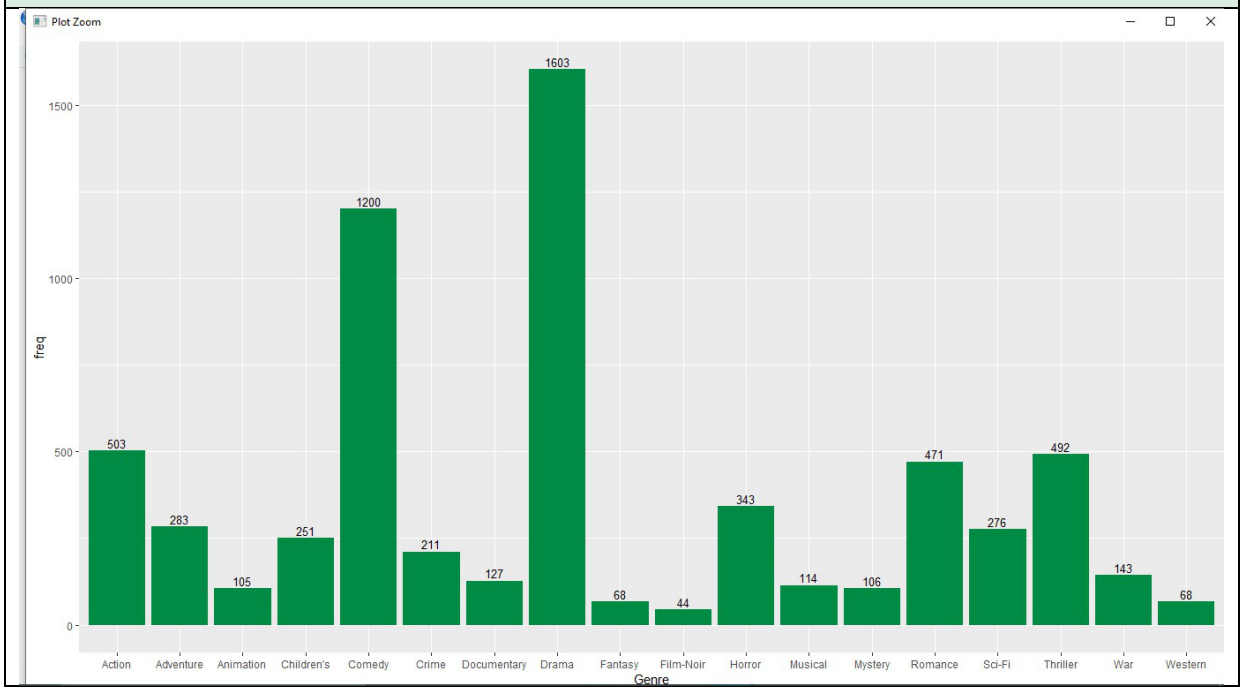
Dari hasil visualisasi diatas, didapatkan informasi mengenai persebaran rating terhadap judul movie. Terlihat bahwa dari 15 film yang dilihat, rating paling dominan adalah dengan skor 5. Sedangkan rating 1-3 tergambarkan dengan warna merah yang berarti hanya sedikit user yang mereview jelek untuk film-film tersebut.

- **Frekuensi Jumlah Film terhadap Genre**

Syntax

```
moviealdy <- select(movies.split, Genre)

dadialdy <- count(moviealdy, c("Genre"))

library(ggplot2)

p<-ggplot(data=dadialdy, aes(x=Genre, y=freq)) +
  geom_bar(stat="identity", fill="springgreen4")+
  geom_text(aes(label=freq), vjust=-0.3, size=3.5)

p
```

Hasil



Dari hasil visualisasi diatas, didapatkan informasi mengenai persebaran movie terhadap genre. Terlihat juga bahwa banyak sekali movie bergenre drama dan sangat timpang terhadap genre lainnya. Judul film paling banyak dimiliki oleh Genre Drama. Sedangkan genre dengan jumlah film paling sedikit adalah Film-Noir. Hal ini menggambarkan bahwa produksi film bergenre drama lebih banyak dan lebih diminati dibandingkan dengan genre-genre lainnya. Sedangkan genre Film-Noir kurang diminati sehingga jarang diproduksi oleh pembuat film.

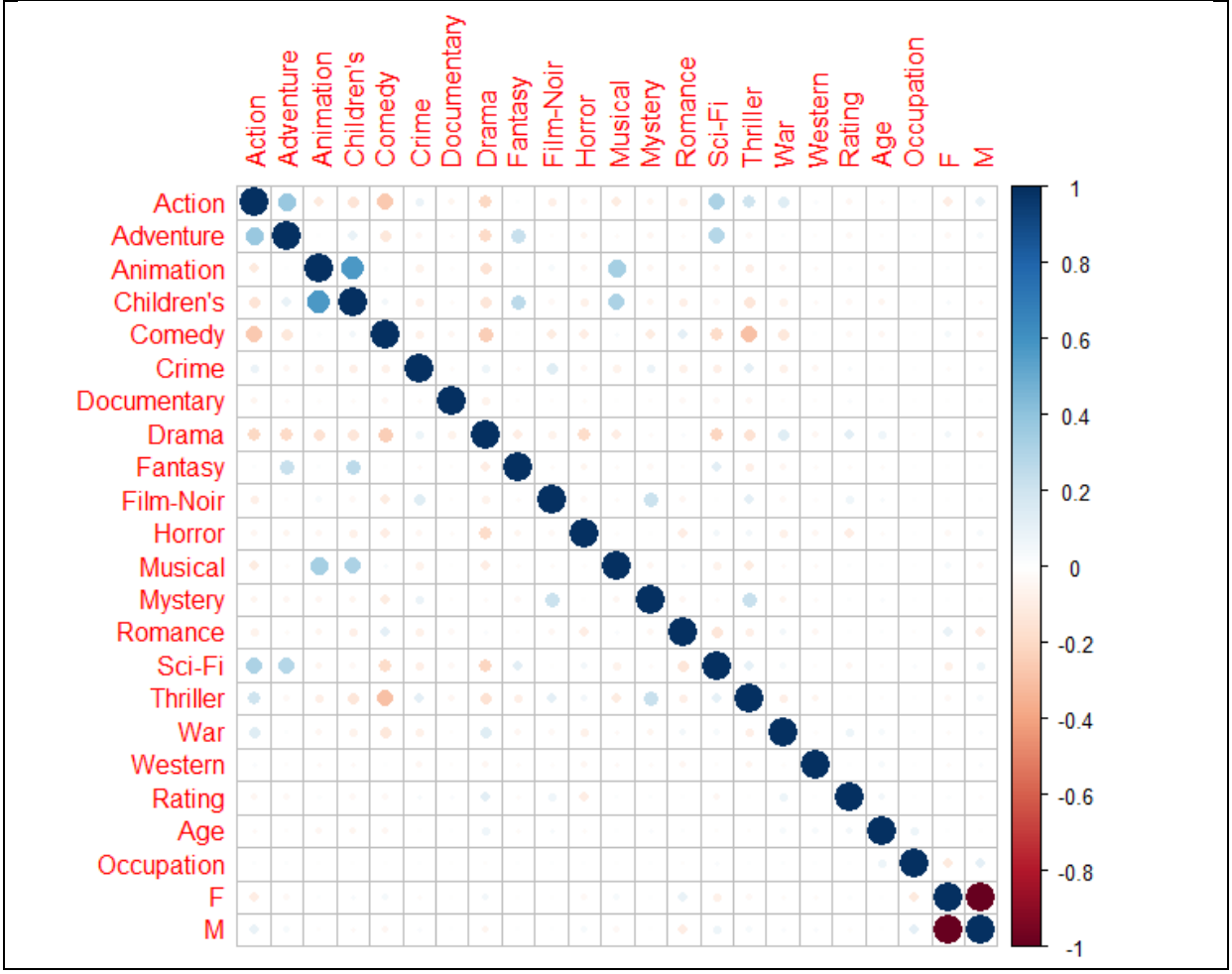
DATA RATING

- **Korelasi Keseluruhan Variabel**

Syntax

```
library(corrplot)
i <- cor(data_dropf[,1:23])
corrplot(i, method = "circle")
```

Hasil



Dari hasil corplot diatas dapat dilihat bahwa beberapa variabel memiliki korelasi yang negatif (representasi dengan warna merah) dan beberapa variabel memiliki korelasi yang positif(representasi dengan warna biru). Beberapa informasi yang didapatkan adalah :

- ➔ Genre Animation dan Children's memiliki korelasi positif yang cukup kuat.
- ➔ Genre Adventure dan Action memiliki korelasi positif yang cukup kuat.
- ➔ Genre Comedy dan Thriller memiliki korelasi negatif.

➔ Genre Musical dan Children's memiliki korelasi positif.
➔ Genre Musical dan Animation memiliki korelasi yang positif.
➔ Genre Sci-Fi dan Action memiliki korelasi yang positif.
➔ Genre Sci-Fi dan Adventure memiliki korelasi yang positif.

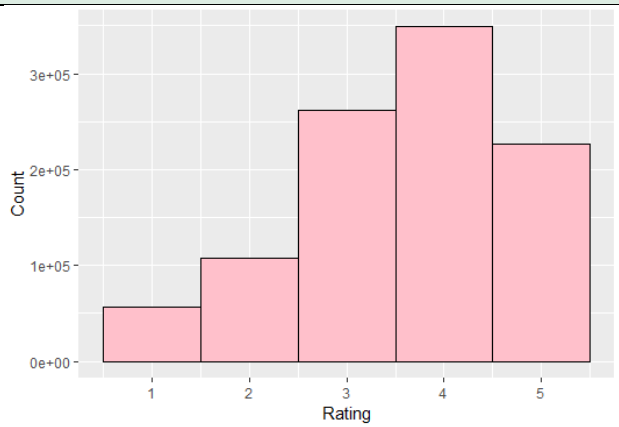
Korelasi positif dapat menggambarkan bahwa terdapat hubungan yang sebanding antar variabel. Misal, pada poin pertama yaitu genre animation dan children's, maka ada kecenderungan film animasi juga merupakan film anak-anak. Sedangkan, korelasi negatif menggambarkan bahwa terdapat hubungan yang berbanding terbalik antar variabel. Misal, pada poin ke 3, yaitu genre comedy dan thriller, maka ada kecenderungan bahwa film komedi bukanlah film thriller. Selain itu, beberapa korelasi lainnya bernilai sangat rendah mendekati nol sehingga dapat dikatakan kurang berkorelasi satu sama lain.

- **Distribusi Rating**

Syntax

```
# distribusi plot
ggplot(data=ratings, aes(x=Rating)) +
  geom_histogram(binwidth = 1, color="black", fill="pink") +
  xlab('Rating') + ylab('Count')
```

Hasil



Dapat dilihat pada hasil visualisasi diatas bahwa rating yang paling banyak adalah bernilai 4, kemudian diikuti nlai 3 dan 5. Sedangkan, rating paling sedikt adalah bernilai 1.

DATA DUPLIKAT

Pertama-tama, dalam melakukan pra-proses data, harus ada pengecekan dan penindaklanjutan untuk data-data yang sifatnya duplikat atau redundan. Untuk melakukan hal tersebut, diperlukan library dplyr dengan fungsi distinct().

Syntax (Pengecekan Data Dupikat)

```
#data duplikat
library(dplyr)
clear.movies<- movies %>% distinct()
clear.users<- users %>% distinct()
clear.ratings<- ratings %>% distinct()
```

Hasil (Pengecekan Data Dupikat)

Sebelum

ratings	1000209 obs. of 4 variables
users	6040 obs. of 5 variables
movies	3883 obs. of 3 variables

Sesudah

clear.movies	3883 obs. of 3 variables
clear.ratings	1000209 obs. of 4 variables
clear.users	6040 obs. of 5 variables

Dari hasil analisis data duplikat pada dataset yang tersedia yaitu pada data movies, users dan reatings, tidak ditemukan adanya data yang sifatnya duplikat. Dapat dilihat dari hasil pemanggilan fungsi distinct() yang tidak mengurangi data satu barispun.

MISSING VALUE

Hal kedua yang perlu diberi perhatian adalah data-data yang berisi nilai N/A atau NULL. Pengecekan tersebut dapat dilakukan dengan fungsi is.na().

Syntax (Missing Value)

```
#mengecek ada tidaknya missing value dari setiap data
sum(is.na(ratings))
sum(is.na(users))
sum(is.na(movies))
```

Hasil (Missing Value)

```
> sum(is.na(ratings))
[1] 0
> sum(is.na(users))
[1] 0
> sum(is.na(movies))
[1] 0
```

Dari hasil diatas yang dapat dilihat bahwa dari ketiga data tidak ada satupun data yang hilang atau bernilai null. Maka tidak perlu ada tindak lanjut untuk data yang hilang.

FITUR KATEGORIKAL

Untuk fitur-fitur yang bersifat kategorikal sebelumnya perlu diberi penanganan tersendiri. Pada dataset tersebut, ada 2 atribut yang sifatnya kategorikal dan perlu ditangani lebih lanjut yaitu pada tabel movies tepatnya pada atribut "Genre" dimana penulisan kategori dipisah dengan tanda "|". Misalnya, Action|Comedy. Selain itu, pada atribut Gender dimana nilai bernilai "M" untuk laki-laki dan "F" untuk perempuan. Kemudian kedua fitur/atribut tersebut perlu dibuat kebentuk lain dengan nilai biner 1 0 sebagai berikut.

- **Genre pada movies**

Syntax

```
#merubah jadi karakter
movies$Genre <- as.character(movies$Genre)

#melakukan split data Genre
library(tidyr)
movies.split <- movies %>%
  mutate(Genre=strsplit(Genre,"[|]")) %>%
  unnest(Genre)

#memasukkan nilai 1 dan 0 pada data
movies.split2 <- movies.split %>%
  mutate(value=1) %>% spread(Genre,value,fill=0)
View(movies.split2)
```

Hasil

Title	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror	Musical	Mystery
Toy Story (1995)	0	0	1	1	1	0	0	0	0	0	0	0	
Jumanji (1995)	0	1	0	1	0	0	0	0	1	0	0	0	
Grumpier Old Men (1995)	0	0	0	0	1	0	0	0	0	0	0	0	
Waiting to Exhale (1995)	0	0	0	0	1	0	0	1	0	0	0	0	
Father of the Bride Part I...	0	0	0	0	1	0	0	0	0	0	0	0	
Heat (1995)	1	0	0	0	0	1	0	0	0	0	0	0	
Sabrina (1995)	0	0	0	0	1	0	0	0	0	0	0	0	
Tom and Huck (1995)	0	1	0	1	0	0	0	0	0	0	0	0	
Sudden Death (1995)	1	0	0	0	0	0	0	0	0	0	0	0	
GoldenEye (1995)	1	1	0	0	0	0	0	0	0	0	0	0	
American President, The (...	0	0	0	0	1	0	0	1	0	0	0	0	

Showing 1 to 12 of 3,883 entries

- **Gender pada Users**

Syntax

```
users2 <- users %>% mutate(value=1) %>% spread(Gender, value, fill=0)
```

Hasil

	UserID	Age	Occupation	Zip.code	F	M
1	1	1	10	48067	1	0
2	2	56	16	70072	0	1
3	3	25	15	55117	0	1
4	4	45	7	02460	0	1
5	5	25	20	55455	0	1
6	6	50	9	55117	1	0
7	7	35	1	06810	0	1
8	8	25	12	11413	0	1
9	9	25	17	61614	0	1
10	10	35	1	95370	1	0
11	11	25	1	04093	1	0
12	12	25	12	32793	0	1

Showing 1 to 12 of 6,040 entries

PENGGABUNGAN TABEL/DATA FRAME

Setelah data kategorikal sudah diproses lebih lanjut, maka selanjutnya perlu adanya penggabungan data frame yang semula ada 3, yaitu movies, users dan ratings menjadi satu kestuan data frame. Penggabungan dilakukan dengan melibatkanatribut movieID dan userID.

Syntax

```
#merge
merge1 <- merge(movies.split2,ratings, by.x ="MovieID", by.y ="MovieID" )
merge2 <- merge(merge1,users2, by.x ="UserID", by.y ="UserID" )
```

Hasil

	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fantasy	Film-Noir	Horror
1	0	0	0	0	0	0	0	1	0	0	
2	0	0	0	0	1	0	0	0	0	0	
3	0	0	0	0	0	0	0	0	0	0	
4	0	0	1	1	1	0	0	0	0	0	
5	0	0	0	0	0	0	0	0	0	0	
6	0	0	0	0	1	0	0	0	0	0	
7	0	0	0	0	0	0	0	0	0	0	
8	0	0	1	1	0	0	0	0	0	0	
9	0	0	0	0	0	0	0	1	0	0	
10	0	0	1	1	0	0	0	0	0	0	
11	0	0	1	1	0	0	0	0	0	0	

Showing 1 to 12 of 1,000,209 entries

Selain itu, untuk mengubah urutan kolom/atribut pada data frame, perlu dijalankan syntax sebagai berikut. Perpindahan kolom dilakukan agar menempatkan variabel dependen di akhir data frame.

```
data_dropf <- data_dropf[,c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,20,21,22,23,19)]
```

DROP FITUR

Untuk melakukan klasifikasi sendiri, tidak semua atribut/fitur diperlukan. Maka dari itu, perlu adanya drop atau pembuangan fitur. Beberapa fitur yang dibuang antara lain yaitu User id, Movie id, Title, Timestamp dan Zipcode.

Syntax

```
#drop fitur
dropf <- c("UserID", "MovieID", "Title", "Timestamp", "Zip.code")
data_dropf <- merge2 [,!(names(merge2) %in% dropf)]
```

Hasil

Filter

	Action	Adventure	Animation	Children's	Comedy	Crime	Documentary	Drama	Fantasy
1	0	0	0	0	0	0	0	1	
2	0	0	0	0	1	0	0	0	
3	0	0	0	0	0	0	0	0	
4	0	0	1	1	1	0	0	0	
5	0	0	0	0	0	0	0	0	
6	0	0	0	0	1	0	0	0	
7	0	0	0	0	0	0	0	0	
8	0	0	1	1	0	0	0	0	
9	0	0	0	0	0	0	0	1	
10	0	0	1	1	0	0	0	0	

Showing 1 to 12 of 1,000,209 entries

SAMPLING

Untuk menanggulangi memori laptop yang tidak cukup dalam pengolahan data, maka perlu adanya proses sampling. Metode sampling yang digunakan pada kali ini adalah dengan random sampling. Metode ini dapat dilakukan dengan memanggil fungsi sample().

Jika merujuk pada rumus statistika sampling menggunakan metode slovin, sebagai berikut :

$$n = \frac{N}{(1+N\ x\ e^2)}$$

Dengan menggunakan tingkat kepercayaan (confidence level) sebesar 95%, maka :

$$n = \frac{1000209}{(1+1000209\ x\ 0.05^2)} = = 399.8400974$$

Dari perhitungan tersebut dapat disimpulkan bahwa jumlah sample minimal yang dibuthkan adalah 399.84 ≅ 400. Namun dengan pertimbangan spesifikasi laptop yang digunakan adalah RAM 4 GB dan intel i3, maka kami menggunakan sample sebanyak 10000 data.

Syntax

```
#sampling
data_sample <- data_dropf[sample(nrow(data_dropf), 10000), ]
```

Hasil

prestep.R* | data_sample | train | t1.R

Filter

	Action	Adventure	Animation	Children's	Comedy
769143	0	0	0	0	0
277470	1	0	0	0	0
826940	0	0	1	1	1
717112	0	0	0	0	0
884089	0	0	0	1	0
257874	0	0	0	0	0
425850	0	0	0	0	0
421616	1	1	0	0	0
310620	1	0	0	0	0
525485	0	0	0	0	0
138104	0	0	0	0	0
226269	0	0	0	0	0
790552	0	0	0	0	1

Showing 1 to 14 of 10,000 entries

TRAIN-TEST DATA

Langkah terakhir yang dilakukan pada pra proses data yaitu membagi data mejadi train set dan test set. Kali ini, digunakan perbandingan 70:30 untuk pembagiannya. Hal ini didasari karena banyak paper yang menggunakan pembagian data dengan porsi demikian. Dari 10000 data sampel yang dimiliki, maka dipilih secara random 7000 baris data menjadi data training dan 3000 baris data menjadi data testing.

Syntax

```
#data train-test
set.seed(1234)

smp_size <- floor(0.7 * nrow(data_dropf))
train_ind <- sample(seq_len(nrow(data_dropf)), size = smp_size)
train <- data_dropf[train_ind, ]
test <- data_dropf[-train_ind, ]
```

Hasil

Train set :

train	7000 obs. of 23 variables

Test set :

test	3000 obs. of 23 variables

BAB III : KLASIFIKASI

DECISION TREE

Salah satu algoritma klasifikasi yang terknal adalah Decision Tree. Decision tree adalah model prediksi menggunakan struktur pohon atau struktur berhirarki. Konsep dari pohon keputusan adalah mengubah data menjadi decision tree dan aturan-aturan keputusan.

PROSES KLASIFIKASI (DECISION TREE)

Sub-bab ini berisikan bagaimana model klasifikasi decision tree akan dibentuk.

CART

Decision tree dapat dibangun dengan beberapa metode, salah satunya adalah CART (Classification and Regression Tree). Dimana metode ini merupakan gabungan dari dua jenis pohon, yaitu classification tree dan juga regression tree. Dengan menggunakan tools R, CART dapat dibangun dengan menggunakan library *rpart*. Karena keterbatasan algoritma dan data yang kurang baik, pohon yang dihasilkan untuk data train secara keseluruahan (7000 data) tidak menghasilkan gambaran tree yang baik, karena semua data digolongkan pada kelas 4. Namun, jika data diambil beberapa, dalam hal ini digunakan rumus sampling slovin sebagai berikut :

$$n = \frac{N}{(1+N\ x\ e^2)}$$

Dengan menggunakan tingkat kepercayaan (confidence level) sebesar 95%, maka :

$$n = \frac{1000209}{(1+1000209\ x\ 0.05^2)} = = 399.8400974$$

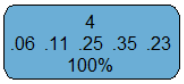
Dari perhitungan tersebut dapat disimpulkan bahwa jumlah sample minimal yang dibuthkan adalah 399.84 ≅ 400. Maka dari itu, untuk membuat tree yang lebih representatif maka dibentuklah tree dengan sample 400 data dan menghasilkan tree seperti terlampir.

Syntax

```
fit <- rpart(Rating~., data=train2, parms = list(split = 'gini'))
rpart.plot(fit, box.palette="Blues")
summary(fit)

test_pred_cartG <- predict(fit, type='class', newdata = test2[,1:22])
pred_act_cartG <- table(test_pred_cartG, test2$Rating)
cmcartG <- confusionMatrix(table(test_pred_cartG, test2$Rating))
```

Hasil

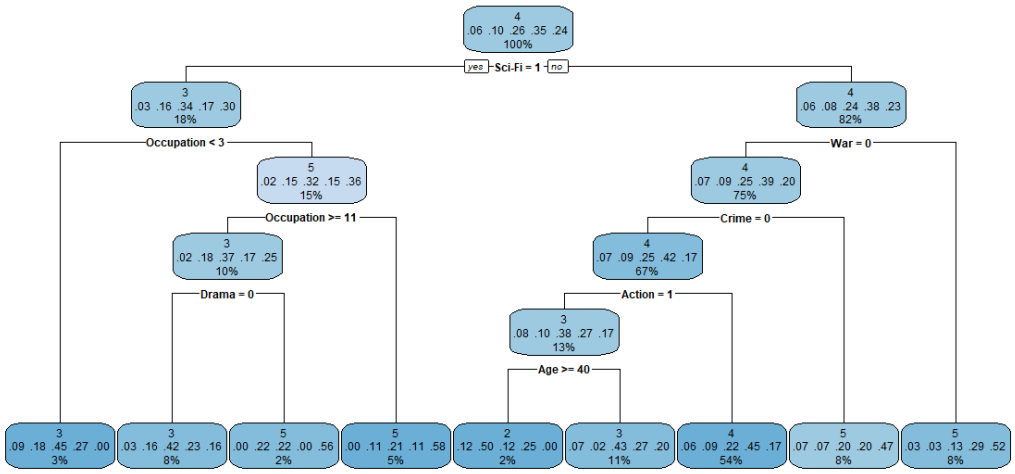
(Dengan 10000 data)



Confusion Matrix

```
Confusion Matrix and Statistics


test_pred_cartG    1    2    3    4    5
              1    0    0    0    0    0
              2    0    0    0    0    0
              3    0    0    0    0    0
              4  152  329  749 1089  681
              5    0    0    0    0    0

Overall Statistics

               Accuracy : 0.363
                 95% CI : (0.3458, 0.3805)
    No Information Rate : 0.363
    P-Value [Acc > NIR] : 0.5069

                  Kappa : 0

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity           0.00000   0.0000   0.0000    1.000    0.000
Specificity           1.00000   1.0000   1.0000    0.000    1.000
Pos Pred Value            NaN      NaN      NaN    0.363      NaN
Neg Pred Value        0.94933   0.8903   0.7503      NaN    0.773
Prevalence            0.05067   0.1097   0.2497    0.363    0.227
Detection Rate        0.00000   0.0000   0.0000    0.363    0.000
Detection Prevalence  0.00000   0.0000   0.0000    1.000    0.000
Balanced Accuracy     0.50000   0.5000   0.5000    0.500    0.500
```

(Dengan 400 sample data)



Confusion Matrix

```
Confusion Matrix and Statistics

test_pred_cartG  1  2  3  4  5
              1  0  0  0  0  0
              2  0  0  4  1  3
              3  3 12 22 32 18
              4 17 31 62 85 44
              5  3  4 16 21 22

Overall Statistics

               Accuracy : 0.3225
                 95% CI : (0.2769, 0.3707)
    No Information Rate : 0.3475
    P-Value [Acc > NIR] : 0.8652

                  Kappa : 0.0288

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity            0.0000   0.0000   0.2115   0.6115   0.2529
Specificity            1.0000   0.9773   0.7804   0.4100   0.8594
Pos Pred Value            NaN   0.0000   0.2529   0.3556   0.3333
Neg Pred Value         0.9425   0.8801   0.7380   0.6646   0.8054
Prevalence             0.0575   0.1175   0.2600   0.3475   0.2175
Detection Rate         0.0000   0.0000   0.0550   0.2125   0.0550
Detection Prevalence   0.0000   0.0200   0.2175   0.5975   0.1650
Balanced Accuracy      0.5000   0.4887   0.4960   0.5107   0.5561
```

Dari kedua model yang dibentuk, dapat dilihat bahwa nilai akurasi model pertama memiliki nilai sebesar 3.63.

C4.5

Selain CART, decision tree juga dapat dibangun dengan agoritma C4.5. Algoritma C4.5 merupakan pengembangan dari algoritma ID3, Proses pada pohon keputusan adalah mengubah bentuk data (tabel) menjadi model pohon, mengubah model pohon menjadi rule, dan menyederhanakan rule. Dalam R, algoritma ini bisa dijalankan dengan menggunakan library RWeka dan memanggil fungsi J48().

Syntax

```
#=======C45
library(RWeka)

train$Rating = factor(train$Rating)
c45model <- J48(Rating~., data = train)
print(c45model)

c45predict <- predict(c45model, test[,1:22], type="class")
confusionMatrix(table(c45predict, test$Rating))
```

Hasil

Model

```
|   |   |   |   |   |   |   |   Sci-Fi <= 0
|   |   |   |   |   |   |   |   |   Occupation <= 16: 2 (3.0/1.0)
|   |   |   |   |   |   |   |   |   Occupation > 16: 3 (2.0/1.0)
|   |   |   |   |   |   |   |   Sci-Fi > 0
|   |   |   |   |   |   |   |   |   Occupation <= 5: 1 (2.0/1.0)
|   |   |   |   |   |   |   |   |   Occupation > 5: 5 (2.0)
|   |   |   |   |   |   |   |   Age > 35: 4 (9.0/2.0)
|   |   |   |   |   |   |   F > 0: 5 (4.0/1.0)
|   |   |   |   Crime > 0
|   |   |   |   |   F <= 0
|   |   |   |   |   |   Comedy <= 0: 3 (4.0/1.0)
|   |   |   |   |   |   Comedy > 0
|   |   |   |   |   |   |   Occupation <= 2: 3 (2.0/1.0)
|   |   |   |   |   |   |   Occupation > 2: 1 (3.0/1.0)
|   |   |   |   |   F > 0: 2 (2.0)
|   |   |   Mystery > 0: 3 (17.0/8.0)
|   |   Drama > 0
|   |   |   F <= 0
|   |   |   |   Age <= 35
|   |   |   |   |   Age <= 18: 4 (4.0/2.0)
|   |   |   |   |   Age > 18
|   |   |   |   |   |   Occupation <= 16: 4 (14.0/8.0)
|   |   |   |   |   |   Occupation > 16: 3 (4.0/1.0)
|   |   |   |   Age > 35: 2 (3.0/1.0)
|   |   |   F > 0: 3 (5.0/1.0)
Documentary > 0
|   Musical <= 0
|   |   Occupation <= 0: 4 (9.0/2.0)
|   |   Occupation > 0: 5 (45.0/25.0)
|   Musical > 0: 4 (3.0/1.0)

Number of Leaves  :     651

Size of the tree :      1301
```

Confusion Matrix

```
Confusion Matrix and Statistics


c45predict   1   2   3   4   5
         1  11  18  23  27  10
         2  14  26  51  66  30
         3  33  92 160 208 111
         4  73 149 417 571 345
         5  21  44  98 217 185

Overall Statistics

               Accuracy : 0.3177
                 95% CI : (0.301, 0.3347)
    No Information Rate : 0.363
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0396

 Mcnemar's Test P-Value : <2e-16

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity          0.072368 0.079027  0.21362   0.5243  0.27166
Specificity          0.972612 0.939723  0.80275   0.4851  0.83614
Pos Pred Value       0.123596 0.139037  0.26490   0.3672  0.32743
Neg Pred Value       0.951563 0.892286  0.75417   0.6415  0.79630
Prevalence           0.050667 0.109667  0.24967   0.3630  0.22700
Detection Rate       0.003667 0.008667  0.05333   0.1903  0.06167
Detection Prevalence 0.029667 0.062333  0.20133   0.5183  0.18833
Balanced Accuracy    0.522490 0.509375  0.50819   0.5047  0.55390
```

Gambar Decision Tree yang terbentuk cukup panjang karena data train yang digunakan adalah sebanyak 7000 data. Sehingga screen tidak cukup untuk menangkap keseluruhan gambar. Maka, gambar terlampir merupakan salah satu potongan tree yang terbentuk. Tree yang terbentuk sendiri terdiri dari 651 nodes dan memiliki ukuran (tree size) sebesar 1301. Hasil prediksi yang dihasilkan oleh algoritma tersebut tersebar atau terdistribusi untuk tiap-tiap kelas, dapat dilihat di confusion matrix pada gambar terlampir.

EVALUASI MODEL (DECISION TREE)

Evaluasi kinerja dari model pengklasifikasi dilakukan dengan menggunakan repeated hold-out sebanyak 100 kali putaran dan 10-fold cross validation.

REPEATED HOLDOUT (DECISION TREE)

Metode holdout dapat dibuat lebih handal dengan mengulang proses untuk sub sampel yang berbeda. Pada setiap iterasi, suatu proporsi tertentu dipilih secara acak untuk training. Kinerja pada setiap iterasi dirata-rata untuk mendapatkan estimasi kinerja total.

Untuk melakukan mekanisme evaluasi kinerja tersebut, maka dapat menggunakan library rminer dengan membagi data sebesar 2 banding 3. Kemudian dijalankan syntax sebagai berikut.

- **CART**

Syntax

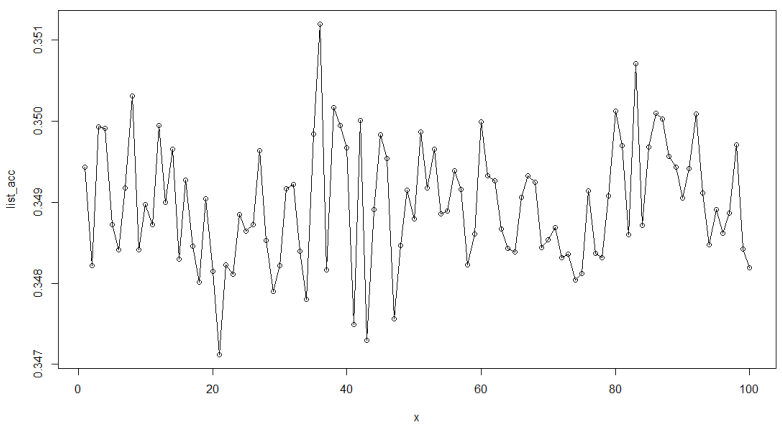
```
#holdout - CART

data_sample$Rating <- as.factor(data_sample$Rating)
library(rminer)
full_accuray_tree=0
list_acc <- list()
for(b in 1:100)
{
  H=holdout(data_sample$Rating, ratio = 2/3, mode = "random", seed=NULL)
  fit <- rpart(Rating~., data=data_sample[H$tr,], parms = list(split = 'gini'))
  cart_pred <- predict(fit, type='class', newdata = data_sample[H$ts,-23])
  resultcart <- confusionMatrix(table(cart_pred, data_sample[H$ts,]$Rating))
  accuracy <- resultcart$overall['Accuracy']
  cat("batch :",b,
      "accuracy:",accuracy,"\n")
  full_accuray_tree=full_accuray_tree+accuracy
  list_acc[[b]] <- accuracy
}
cat("Tree :",
    "accuracy:", full_accuray_tree/100, "\n")

x <- c(1:100)
plot(x, list_acc, type="o");
```

Hasil

Iterasi

```
batch : 78 accuracy: 0.3483143
batch : 79 accuracy: 0.3490823
batch : 80 accuracy: 0.3501233
batch : 81 accuracy: 0.3497003
batch : 82 accuracy: 0.3486023
batch : 83 accuracy: 0.3507053
batch : 84 accuracy: 0.3487163
batch : 85 accuracy: 0.3496823
batch : 86 accuracy: 0.3500963
batch : 87 accuracy: 0.3500303
batch : 88 accuracy: 0.3495683
batch : 89 accuracy: 0.3494333
batch : 90 accuracy: 0.3490493
batch : 91 accuracy: 0.3494153
batch : 92 accuracy: 0.3500873
batch : 93 accuracy: 0.3491153
batch : 94 accuracy: 0.3484763
batch : 95 accuracy: 0.3489083
batch : 96 accuracy: 0.3486203
batch : 97 accuracy: 0.3488633
batch : 98 accuracy: 0.3497093
batch : 99 accuracy: 0.3484253
batch : 100 accuracy: 0.3481943
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
Tree : accuracy: 0.3489709
```

- **C45**

Syntax

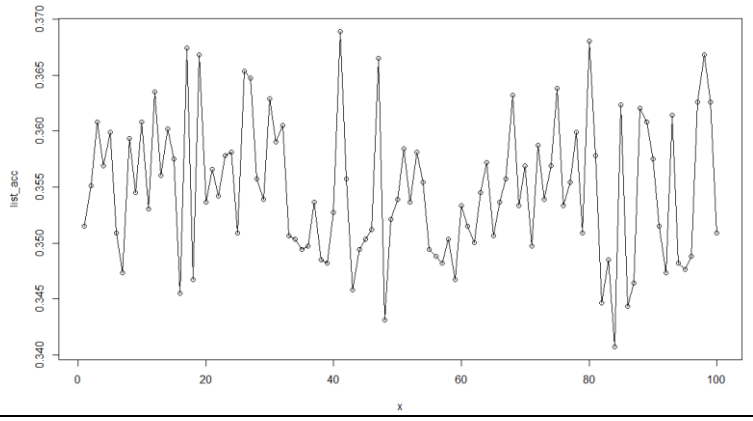
```
#holdout - C45
library(rminer)
full_accuray_tree=0
list_acc <- list()
data_sample$Rating <- as.factor(data_sample$Rating)
for(b in 1:100)
{
  H=holdout(data_sample$Rating, ratio = 2/3, mode = "random", seed=NULL)
  c45model <- J48(Rating~., data=data_sample[H$tr,])
  c45_pred <- predict(fit, type='class', newdata = data_sample[H$ts,-23])
  resultc45 <- confusionMatrix(table(c45_pred, data_sample[H$ts,]$Rating))
  accuracy <- resultc45$overall['Accuracy']
  cat("batch :",b,
      "accuracy:",accuracy,"\n")
  full_accuray_tree=full_accuray_tree+accuracy
  list_acc[[b]] <- accuracy
}
cat("Tree :",
    "accuracy:", full_accuray_tree/100, "\n")

x <- c(1:100)
plot(x, list_acc, type="o");
```

Hasil

Iterasi

```
batch : 85 accuracy: 0.3623275
batch : 86 accuracy: 0.3443311
batch : 87 accuracy: 0.3464307
batch : 88 accuracy: 0.3620276
batch : 89 accuracy: 0.3608278
batch : 90 accuracy: 0.3575285
batch : 91 accuracy: 0.3515297
batch : 92 accuracy: 0.3473305
batch : 93 accuracy: 0.3614277
batch : 94 accuracy: 0.3482304
batch : 95 accuracy: 0.3476305
batch : 96 accuracy: 0.3488302
batch : 97 accuracy: 0.3626275
batch : 98 accuracy: 0.3668266
batch : 99 accuracy: 0.3626275
batch : 100 accuracy: 0.3509298
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
Tree : accuracy: 0.354814
```

Nilai akurasi yang dihasilkan dari kedua mekanisme evaluasi kinerja tersebut, baik dengan metode CART maupun C4.5 menghasilkan nilai akurasi sekitar 30%. Namun, nilai akurasi keseluruhan yang lebih tinggi dihasilkan oleh C4.5 dengan nilai 35%.

CROSS VALIDATION (DECISION TREE)

Metode cross-validation akan menghindari tumpang tindih pada data testing. Terdiri dari 2 tahap yaitu membagi data menjadi k bagian dengan ukuran yang sama dan menggunakan masing-masing bagian untuk testing, sisanya sebagai training.

Dengan bantuan tools R, berikut merupakan syntax yang perlu dijalankan.

- **CART**

Syntax

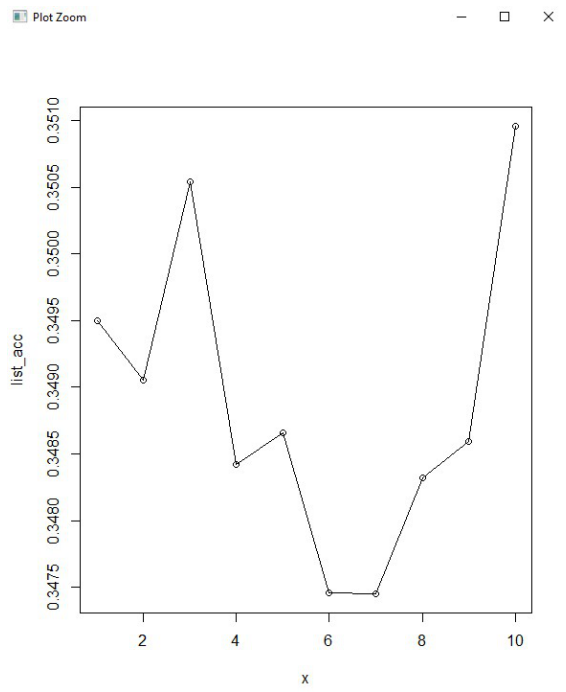
```
#cross validation - CART (gini)
folds <- cut(seq(1, nrow(train)), breaks=10, labels=FALSE)
full_acuracy_tree=0
list_acc <- list()

for (i in 1:10){
  testIndexes <- which(folds==i,arr.ind=TRUE)
  testData <- train[testIndexes, ]
  trainData <- train[-testIndexes, ]
  CARTginimodel <- rpart(Rating~., data=trainData, parms = list(split = 'gini'))
  test_pred <- predict(CARTginimodel, type='class', newdata = testData[,-23])
  resultCG <- confusionMatrix(table(test_pred, testData$Rating))
  accuracy <- resultCG$overall['Accuracy']
  cat("batch :",i,
      "accuracy:",accuracy,"\n")
  full_acuracy_tree=full_acuracy_tree+accuracy
  list_acc[[i]] <- accuracy
}
cat("Tree :",
    "accuracy:", full_acuracy_tree/10, "\n")

x <- c(1:10)
plot(x, list_acc, type="o");
```

Hasil

Iterasi

```
batch : 1 accuracy: 0.3495
batch : 2 accuracy: 0.34905
batch : 3 accuracy: 0.35054
batch : 4 accuracy: 0.34842
batch : 5 accuracy: 0.34866
batch : 6 accuracy: 0.34746
batch : 7 accuracy: 0.34745
batch : 8 accuracy: 0.34832
batch : 9 accuracy: 0.34859
batch : 10 accuracy: 0.35096
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
Tree : accuracy: 0.348895
```

- **C45**

Syntax

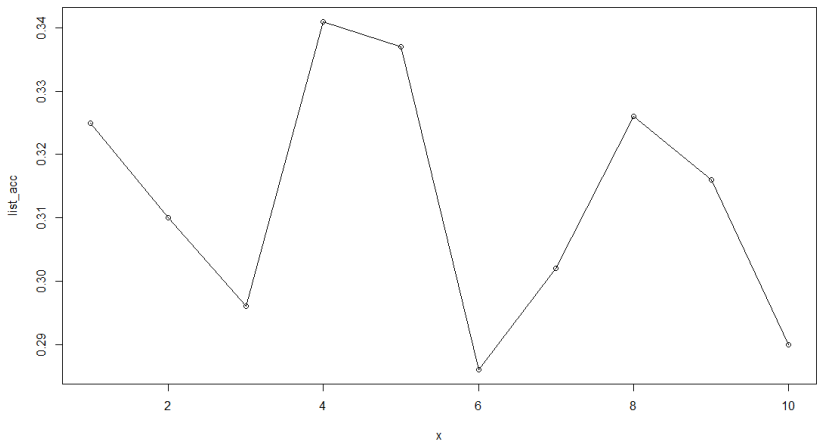
```
#cross validation - C45
data_sample$Rating <- as.factor(data_sample$Rating)
folds <- cut(seq(1, nrow(data_sample)), breaks=10, labels=FALSE)
full_accuracy_tree=0
list_acc <- list()

for (i in 1:10){
  testIndexes <- which(folds==i,arr.ind=TRUE)
  testData <- data_sample[testIndexes, ]
  trainData <- data_sample[-testIndexes, ]
  C45model <- J48(Rating~., data=trainData)
  test_pred <- predict(C45model, type='class', newdata = testData[,-23])
  resultCG <- confusionMatrix(table(test_pred, testData$Rating))
  accuracy <- resultCG$overall['Accuracy']
  cat("batch :",i,
      "accuracy:",accuracy,"\n")
  full_accuracy_tree=full_accuracy_tree+accuracy
  list_acc[[i]] <- accuracy
}
cat("Tree :",
    "accuracy:", full_accuracy_tree/10, "\n")

x <- c(1:10)
plot(x, list_acc, type="o");
```

Hasil

Iterasi

```
batch : 1 accuracy: 0.325
batch : 2 accuracy: 0.31
batch : 3 accuracy: 0.296
batch : 4 accuracy: 0.341
batch : 5 accuracy: 0.337
batch : 6 accuracy: 0.286
batch : 7 accuracy: 0.302
batch : 8 accuracy: 0.326
batch : 9 accuracy: 0.316
batch : 10 accuracy: 0.29
```

Plot Nilai Akurasi



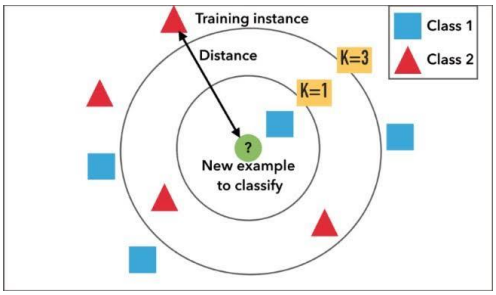
Nilai Akurasi Keseluruhan

```
Tree : accuracy: 0.3129
```

Berbeda dengan hasil repeated holdout sebelumnya, dengan mekanisme cross-validation didapatkan akurasi yang paling tinggi adalah dengan metode CART dengan nilai 34,8%.

K-NEAREST NEIGHBOR

Algoritma k-nearest neighbor (k-NN atau KNN) adalah sebuah metode untuk melakukan klasifikasi terhadap objek berdasarkan data pembelajaran yang jaraknya paling dekat dengan objek tersebut. Ilustrasi algoritma yang akan dilakukan adalah sebagai berikut :



PROSES KLASIFIKASI (KNN)

Dengan menggunakan tools R, berikut merupakan syntax yang dapat dijalankan untuk menggunakan algoritma knn dengan memanggil function train() pada library caret.

Syntax

```
#model
train3$Rating <- factor(train3$Rating)
fit.knn <- train(Rating~., data=train3, method="knn",
                 preProcess=c("center", "scale"))
#summarize fit
print(fit.knn)
fit.knn$finalModel

#makeprediction
knnpredict2 <- predict(fit.knn, newdata=test3[,1:22])
View(knnpredict2)

#accuracy
table(knnpredict2, test3$Rating)
library(caret)
confusionMatrix(table(knnpredict2, test3$Rating))
```

Hasil

Pemilihan Model

```
> print(fit.knn)
k-Nearest Neighbors

7000 samples
  22 predictor
   5 classes: '1', '2', '3', '4', '5'

Pre-processing: centered (22), scaled (22)
Resampling: Bootstrapped (25 reps)
Summary of sample sizes: 7000, 7000, 7000, 7000, 7000, 7000, ...
Resampling results across tuning parameters:

  k  Accuracy   Kappa
  5  0.2917908  0.02657921
  7  0.2970119  0.02766644
  9  0.2995769  0.02635638

Accuracy was used to select the optimal model using the largest value.
The final value used for the model was k = 9.
```

Distribusi Outcome untuk k=9

```
> fit.knn$finalModel
9-nearest neighbor model
Training set outcome distribution:

   1    2    3    4    5
 431  697 1848 2489 1535
```

Confusion Matrix

```
> confusionMatrix(table(knnpredict2, test3$Rating))
Confusion Matrix and Statistics


knnpredict2   1   2   3   4   5
          1   1   8   5  15   4
          2  13  26  29  33   8
          3  55 119 222 267 158
          4  84 150 373 546 340
          5  20  32 119 206 167

Overall Statistics

               Accuracy : 0.3207
                 95% CI : (0.304, 0.3377)
    No Information Rate : 0.3557
    P-Value [Acc > NIR] : 1

                  Kappa : 0.042

 Mcnemar's Test P-Value : <2e-16

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity          0.0057803 0.077612   0.2968   0.5117  0.24668
Specificity          0.9886806 0.968856   0.7340   0.5101  0.83771
Pos Pred Value       0.0303030 0.238532   0.2704   0.3657  0.30699
Neg Pred Value       0.9420290 0.893117   0.7586   0.6543  0.79235
Prevalence           0.0576667 0.111667   0.2493   0.3557  0.22567
Detection Rate       0.0003333 0.008667   0.0740   0.1820  0.05567
Detection Prevalence 0.0110000 0.036333   0.2737   0.4977  0.18133
Balanced Accuracy    0.4972305 0.523234   0.5154   0.5109  0.54219
```

Dengan dijalankannya syntax pertama, maka sistem akan melakukan komputasi untuk beberapa nilai "k". Dalam kasus ini, k yang digunakan adalah 5, 7 dan 9. Dari hasil akurasi yang muncul, k bernilai 9 memiliki nilai akurasi yang paling tinggi. Maka dari itu, model dibentuk dengan menggunakan k = 9. Kemudian dilakukan prediksi dan keluarlah hasil akurasi untuk model tersebut.

EVALUASI MODEL (KNN)

Evaluasi kinerja dari model pengklasifikasi dilakukan dengan menggunakan repeated hold-out sebanyak 100 kali putaran dan 10-fold cross validation .

REPEATED HOLDOUT (KNN)

Metode holdout dapat dibuat lebih handal dengan mengulang proses untuk sub sampel yang berbeda. Pada setiap iterasi, suatu proporsi tertentu dipilih secara acak untuk training. Kinerja pada setiap iterasi dirata-rata untuk mendapatkan estimasi kinerja total.

Untuk melakukan mekanisme evaluasi kinerja tersebut, maka dapat menggunakan library rminer dengan membagi data sebesar 2 banding 3. Kemudian dijalankan syntax sebagai berikut.

Syntax

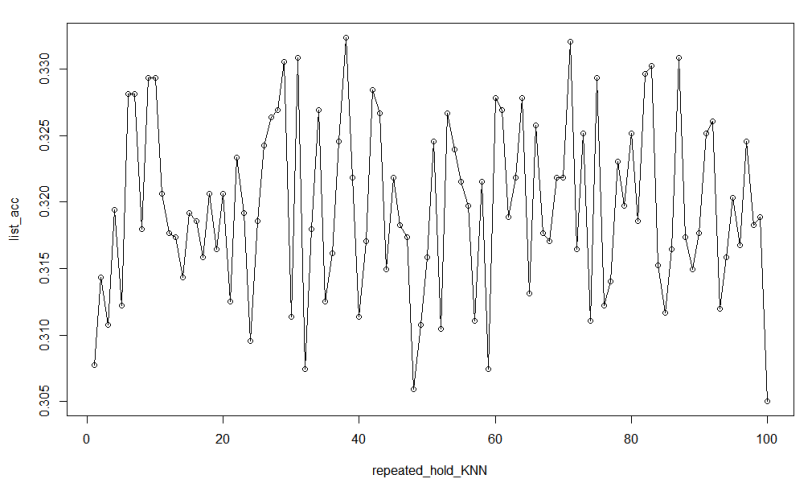
```
#repeated hold 100--knn
library(caret)
library(mlbench)
library(rminer)
full_accuracy  = 0
list_acc <- list()
for (b in 1:100) {
  H = holdout(data_sample$Rating, ratio = 2/3 , mode="random", seed=NULL)

  data_sample$Rating <- factor(data_sample$Rating)
  fit.knn <- train(Rating~., data=data_sample [H$tr,], method="knn",
                   preProcess=c("center", "scale"))

  #makeprediction
  knnpredict2 <- predict(fit.knn, newdata=data_sample[H$ts,-23])

  #accuracy
  result <- confusionMatrix(table(knnpredict2, data_sample[H$ts,]$Rating))
  accuracy <- result$overall['Accuracy']

  cat("batch :",b,
      "accuracy:",accuracy,"\n")
  full_accuracy  = full_accuracy + accuracy
  list_acc[[b]] <- accuracy

}
cat("KNN :", "accuracy:", full_accuracy /100, "\n")

repeated_hold_KNN <- c(1:100)
plot(repeated_hold_KNN, list_acc, type = "o")
```

Hasil

Iterasi

```
batch : 84 accuracy: 0.315237
batch : 85 accuracy: 0.3116377
batch : 86 accuracy: 0.3164367
batch : 87 accuracy: 0.3308338
batch : 88 accuracy: 0.3173365
batch : 89 accuracy: 0.314937
batch : 90 accuracy: 0.3176365
batch : 91 accuracy: 0.325135
batch : 92 accuracy: 0.3260348
batch : 93 accuracy: 0.3119376
batch : 94 accuracy: 0.3158368
batch : 95 accuracy: 0.3203359
batch : 96 accuracy: 0.3167367
batch : 97 accuracy: 0.3245351
batch : 98 accuracy: 0.3182364
batch : 99 accuracy: 0.3188362
batch : 100 accuracy: 0.305039
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
KNN : accuracy: 0.3196041
```

CROSS VALIDATION (KNN)

Metode cross-validation akan menghindari tumpang tindih pada data testing. Terdiri dari 2 tahap yaitu membagi data menjadi k bagian dengan ukuran yang sama dan menggunakan masing-masing bagian untuk testing, sisanya sebagai training.

Dengan bantuan tools R, berikut merupakan syntax yang perlu dijalankan.

Syntax

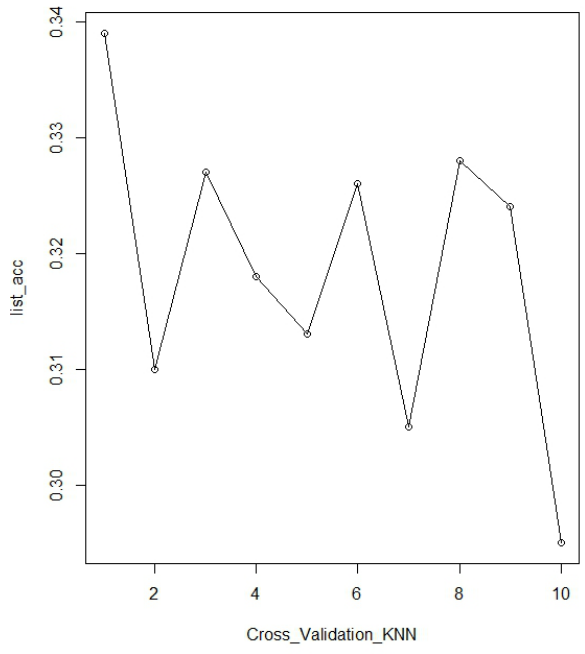
```
#cross validation - KNN
folds <- cut(seq(1, nrow(train)), breaks=10, labels=FALSE)
full_accuracy =0
list_acc <- list()
data_sample$Rating <- factor(data_sample$Rating)
for (i in 1:10){
  testIndexes <- which(folds==i,arr.ind=TRUE)
  testData <- data_sample[testIndexes, ]
  trainData <- data_sample[-testIndexes, ]

  fit.knn <- train(Rating~., data=trainData, method="knn",
                   preProcess=c("center", "scale"))
  knnpredict2 <- predict(fit.knn, newdata = trainData[,-23], type ="class")
  result <- confusionMatrix(table(knnpredict2, testData$Rating))
  accuracy <- result$overall['Accuracy']
  cat("nbbatch :",i,
      "accuracy:",accuracy,"\n")
  full_accuracy  =full_accuracy + accuracy
  list_acc[[i]] <- accuracy
}
cat("KNN :",
    "accuracy:", full_accuracy /10, "\n")

Cross_Validation_KNN <- c(1:10)
plot(Cross_Validation_KNN, list_acc, type="o")
```

Hasil

Iterasi

```
batch : 1 accuracy: 0.339
batch : 2 accuracy: 0.31
batch : 3 accuracy: 0.327
batch : 4 accuracy: 0.318
batch : 5 accuracy: 0.313
batch : 6 accuracy: 0.326
batch : 7 accuracy: 0.305
batch : 8 accuracy: 0.328
batch : 9 accuracy: 0.324
batch : 10 accuracy: 0.295
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
KNN : accuracy: 0.3185
```

Untuk metode KNN, nilai akurasi pada repeadted holdout dan cross validation tidak memiliki selisih yang signifikan. Namun, pada holdout nilai akurasi yangdihasilkan lebih besar 0,001 yaitu 0,319.

BAYESIAN CLASSIFIER

Naive Bayes classifier (NBC) merupakan salah satu metoda pemelajaran mesin yang memanfaatkan perhitungan probabilitas dan statistik yang dikemukakan oleh ilmuwan Inggris Thomas Bayes, yaitu memprediksi probabilitas di masa depan berdasarkan pengalaman di masa sebelumnya.

PROSES KLASIFIKASI (NAÏVE BAYES)

Dalam tools R, naïve bayes bisa dijalankan dengan menggunakan fungsi naiveBayes() sebagai berikut.

Syntax

```
#----------NAIVES BAYES----------#
library(e1071)
train3 <- traincb
test3 <- testcb
View(traincb)

#fitmodel
train3$Rating <- factor(train3$Rating)
nbmodel <- naiveBayes(Rating~., data=train3)
print(nbmodel)

#make prediction
nbpredict <- predict(nbmodel, newdata=test3[, 1:22], type="class")
View(nbpredict)
#accuracy
library(caret)
confusionMatrix(table(nbpredict, test3$Rating))
```

Hasil

Model – Probabilitas Apriori & Konsional

```
> print(nbmodel)

Naive Bayes Classifier for Discrete Predictors

Call:
naiveBayes.default(x = X, y = Y, laplace = laplace)

A-priori probabilities:
Y
        1         2         3         4         5
0.06157143 0.09957143 0.26400000 0.35557143 0.21928571

Conditional probabilities:
   Action
Y        [,1]      [,2]
  1 0.2853828 0.4521214
  2 0.2955524 0.4566183
  3 0.2743506 0.4463072
  4 0.2651667 0.4415107
  5 0.2371336 0.4254635

  Adventure
Y        [,1]      [,2]
  1 0.1600928 0.3671182
  2 0.1621234 0.3688286
  3 0.1439394 0.3511233
  4 0.1321816 0.3387561
  5 0.1218241 0.3271891

  Animation
Y        [,1]      [,2]
  1 0.02784223 0.1647118
  2 0.04017217 0.1965039
  3 0.03733766 0.1896392
  4 0.04660506 0.2108338
  5 0.04755700 0.2128964

  children's
Y        [,1]      [,2]
  1 0.09744780 0.2969112
  2 0.08608321 0.2806883
  3 0.07413420 0.2620601
  4 0.06347931 0.2438721
  5 0.05667752 0.2313007

  F
Y        [,1]      [,2]
  1 0.2645012 0.4415798
  2 0.2295552 0.4208488
  3 0.2445887 0.4299594
  4 0.2470872 0.4314045
  5 0.2495114 0.4328713

  M
Y        [,1]      [,2]
  1 0.7354988 0.4415798
  2 0.7704448 0.4208488
  3 0.7554113 0.4299594
  4 0.7529128 0.4314045
  5 0.7504886 0.4328713

  Mystery
Y        [,1]      [,2]
  1 0.02088167 0.1431544
  2 0.04304161 0.2030965
  3 0.04383117 0.2047747
  4 0.05182804 0.2217243
  5 0.04951140 0.2170039

  Romance
Y        [,1]      [,2]
  1 0.1392111 0.3465690
  2 0.1377331 0.3448672
  3 0.1450216 0.3522180
  4 0.1542788 0.3612884
  5 0.1335505 0.3402796

  Sci-Fi
Y        [,1]      [,2]
  1 0.1647332 0.3713706
  2 0.1836442 0.3874718
  3 0.1666667 0.3727789
  4 0.1394134 0.3464470
  5 0.1511401 0.3583021

  Thriller
Y        [,1]      [,2]
  1 0.1716937 0.3775523
  2 0.1850789 0.3886405
  3 0.2153680 0.4111886
  4 0.2093210 0.4069057
  5 0.1876221 0.3905374
```

```
Comedy
Y        [,1]      [,2]
  1 0.3967517 0.4897922
  2 0.3701578 0.4831935
  3 0.3598485 0.4800857
  4 0.3467256 0.4760231
  5 0.3153094 0.4647904

Crime
Y        [,1]      [,2]
  1 0.05104408 0.2203434
  2 0.06312769 0.2433671
  3 0.08441558 0.2780853
  4 0.08155886 0.2737464
  5 0.09381107 0.2916607

Documentary
Y         [,1]       [,2]
  1 0.004640371 0.06804101
  2 0.004304161 0.06551177
  3 0.008116883 0.08975165
  4 0.009642427 0.09774093
  5 0.011726384 0.10768673

Drama
Y        [,1]      [,2]
  1 0.2250580 0.4181058
  2 0.2840746 0.4512964
  3 0.3279221 0.4695833
  4 0.3680193 0.4823635
  5 0.4201954 0.4937510

War
Y         [,1]       [,2]
  1 0.04640371 0.2106023
  2 0.03156385 0.1749614
  3 0.04924242 0.2164323
  4 0.07071113 0.2563932
  5 0.10293160 0.3039685

Western
Y         [,1]       [,2]
  1 0.01392111 0.1172998
  2 0.01578192 0.1247204
  3 0.02110390 0.1437696
  4 0.02249900 0.1483295
  5 0.02475570 0.1554304

Age
Y      [,1]     [,2]
  1 27.68213 11.37331
  2 27.82353 10.89681
  3 29.61472 11.50899
  4 30.18321 11.56365
  5 30.27622 12.01848

Occupation
Y        [,1]     [,2]
  1 7.835267 6.652383
  2 7.565280 6.280475
  3 7.848485 6.496077
  4 8.125753 6.579212
  5 7.946580 6.305335

Fantasy
Y         [,1]       [,2]
  1 0.05800464 0.2340239
  2 0.03443329 0.1824703
  3 0.04112554 0.1986343
  4 0.03776617 0.1906685
  5 0.03192182 0.1758493

Film-Noir
Y          [,1]       [,2]
  1 0.002320186 0.04816831
  2 0.005738881 0.07559196
  3 0.017316017 0.13048136
  4 0.022498996 0.14832947
  5 0.037133550 0.18915062

Horror
Y         [,1]       [,2]
  1 0.17169374 0.3775523
  2 0.09756098 0.2969332
  3 0.08225108 0.2748212
  4 0.05383688 0.2257409
  5 0.05602606 0.2300470

Musical
Y         [,1]       [,2]
  1 0.03248260 0.1774840
  2 0.03012912 0.1710653
  3 0.03192641 0.1758518
  4 0.03937324 0.1945204
  5 0.04429967 0.2058271
```

Confusion Matrix

```
Confusion Matrix and Statistics


nbpredict   1   2   3   4   5
        1  64 107 186 267 138
        2  69 111 270 332 173
        3   4  22  46  55  35
        4  23  64 171 264 183
        5  13  31  75 149 148

Overall Statistics

               Accuracy : 0.211
                 95% CI : (0.1965, 0.226)
    No Information Rate : 0.3557
    P-Value [Acc > NIR] : 1

                  Kappa : 0.0395

 Mcnemar's Test P-Value : <2e-16

Statistics by Class:

                     Class: 1 Class: 2 Class: 3 Class: 4 Class: 5
Sensitivity           0.36994   0.3313  0.06150   0.2474  0.21861
Specificity           0.75310   0.6833  0.94849   0.7719  0.88463
Pos Pred Value        0.08399   0.1162  0.28395   0.3745  0.35577
Neg Pred Value        0.95130   0.8905  0.75264   0.6501  0.79528
Prevalence            0.05767   0.1117  0.24933   0.3557  0.22567
Detection Rate        0.02133   0.0370  0.01533   0.0880  0.04933
Detection Prevalence  0.25400   0.3183  0.05400   0.2350  0.13867
Balanced Accuracy     0.56152   0.5073  0.50499   0.5096  0.55162
```

Dapat dilihat dari hasil diatas bahwa model yang terbentuk mempertimbangkan beberapa kemungkinan apriori & kondisional, dimana kemungkinan-kemungkinan (probalitiy) tersebut akan membuat sebuah model prediksi klasifikasi. Misal, pada Kemungkinan Apriori, disebutkan bahwa kemungkinan data berlabel kelas 1 (rating bernilai 1) adalah 0,061, data berlabel 2 adalah 0,09 dan seterusnya. Sedangkan kemungkinan kondisional atau *conditional probabilities* menyatakan kemungkinan yang akan terjadi jika dibenturkan dengan kondisi-kondisi tertentu sesuai dengan variabel-variabel independent (Age, Gender, dll) yang digunakan.

EVALUASI MODEL (NAÏVE BAYES)

Evaluasi kinerja dari model pengklasifikasi dilakukan dengan menggunakan repeated hold-out sebanyak 100 kali putaran dan 10-fold cross validation .

REPEATED HOLDOUT (NAÏVE BAYES)

Metode holdout dapat dibuat lebih handal dengan mengulang proses untuk sub sampel yang berbeda. Pada setiap iterasi, suatu proporsi tertentu dipilih secara acak untuk training. Kinerja pada setiap iterasi dirata-rata untuk mendapatkan estimasi kinerja total.

Untuk melakukan mekanisme evaluasi kinerja tersebut, maka dapat menggunakan library rminer dengan membagi data sebesar 2 banding 3. Kemudian dijalankan syntax sebagai berikut.

Syntax

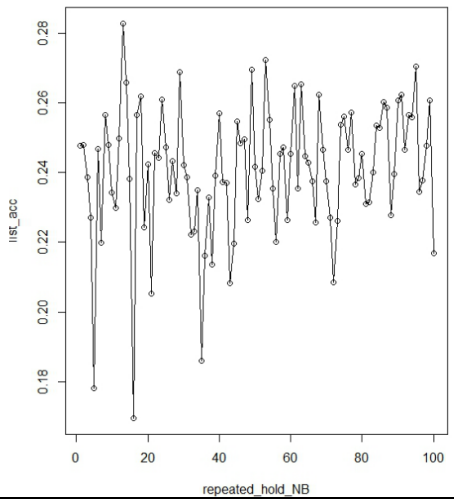
```
##repeated hold 100--Naives Bayes
library(klaR)
library(caret)
library(mlbench)
library(rminer)
library(dplyr)
library(e1071)

full_accuracy  = 0
list_acc <- list()
for (b in 1:100) {
  H = holdout(data_sample$Rating, ratio = 2/3 , mode="random", seed=NULL)


  data_sample$Rating <- factor(data_sample$Rating)
  nbmodel <- naiveBayes(Rating~., data=data_sample [H$tr,])

  #makeprediction
  nbpredict <- predict(nbmodel, newdata=data_sample[H$ts,-23], type="class")

   #accuracy
   result <- confusionMatrix(table(nbpredict, data_sample[H$ts,]$Rating))
   accuracy <- result$overall['Accuracy']

   cat("batch :",b,
       "accuracy:",accuracy,"\n")
   full_accuracy = full_accuracy +accuracy
   list_acc[[b]] <- accuracy

}
cat("Naive Bayes :", "accuracy:", full_accuracy /100, "\n")

repeated_hold_NB <- c(1:100)
plot(repeated_hold_NB, list_acc, type = "o")
```

Hasil

Iterasi

```
batch : 89 accuracy: 0.2396521
batch : 90 accuracy: 0.2606479
batch : 91 accuracy: 0.2624475
batch : 92 accuracy: 0.2465507
batch : 93 accuracy: 0.2564487
batch : 94 accuracy: 0.2558488
batch : 95 accuracy: 0.2705459
batch : 96 accuracy: 0.2345531
batch : 97 accuracy: 0.2378524
batch : 98 accuracy: 0.2477504
batch : 99 accuracy: 0.2606479
batch : 100 accuracy: 0.2168566
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
Naive Bayes : accuracy: 0.2406989
```

CROSS VALIDATION (NAÏVE BAYES)

Metode cross-validation akan menghindari tumpang tindih pada data testing. Terdiri dari 2 tahap yaitu membagi data menjadi k bagian dengan ukuran yang sama dan menggunakan masing-masing bagian untuk testing, sisanya sebagai training.

Dengan bantuan tools R, berikut merupakan syntax yang perlu dijalankan.

Syntax

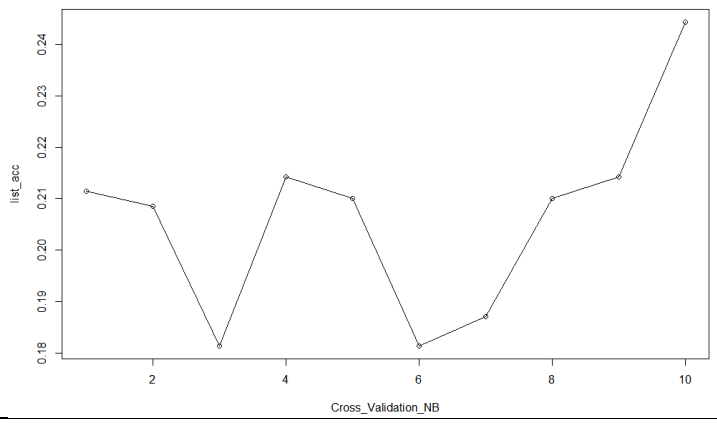
```
#cross validation - NAIVE BAYES
folds <- cut(seq(1, nrow(train)), breaks=10, labels=FALSE)
full_accuracy =0
list_acc <- list()
data_sample$Rating <- factor(data_sample$Rating)
for (i in 1:10){
  testIndexes <- which(folds==i,arr.ind=TRUE)
  testData <- data_sample[testIndexes, ]
  trainData <- data_sample[-testIndexes, ]
  nbmodel <- naiveBayes(Rating~., data=trainData)
  nbpredict <- predict(nbmodel, newdata = testData[,-23], type="class")
  result <- confusionMatrix(table(nbpredict, testData$Rating))
  accuracy <- result$overall['Accuracy']
  cat("batch :",i,
      "accuracy:",accuracy,"\n")
  full_accuracy  =full_accuracy + accuracy
  list_acc[[i]] <- accuracy
}
cat("Naive Bayes :",
    "accuracy:", full_accuracy /10, "\n")

Cross_Validation_NB <- c(1:10)
plot(Cross_Validation_NB, list_acc, type="o")
```

Hasil

Iterasi

```
batch : 1 accuracy: 0.2114286
batch : 2 accuracy: 0.2085714
batch : 3 accuracy: 0.1814286
batch : 4 accuracy: 0.2142857
batch : 5 accuracy: 0.21
batch : 6 accuracy: 0.1814286
batch : 7 accuracy: 0.1871429
batch : 8 accuracy: 0.21
batch : 9 accuracy: 0.2142857
batch : 10 accuracy: 0.2442857
```

Plot Nilai Akurasi



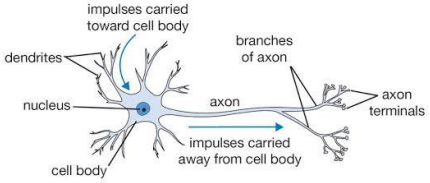
Nilai Akurasi Keseluruhan

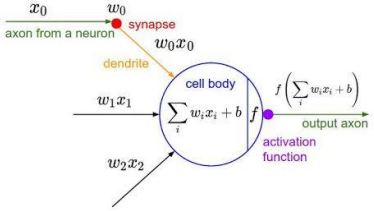
```
Naive Bayes : accuracy: 0.2062857
```

Setelah dilakukan pengujian seperti terlampir diatas, maka didapatkan nilai angka akurasi yang lebih tinggi untuk model naïve bayes adalah dengan menggunakan repeated holdout yaitu sebesar 24%. Meskipun, dua angka (akurasi pada repeated holdout dan cross validation) yang keluar tidak sebagus metode lainnya.

NEURAL NETWORK

Jaringan saraf tiruan (JST) (Bahasa Inggris: artificial neural network (ANN), atau juga disebut simulated neural network (SNN), atau umumnya hanya disebut neural network (NN)), adalah jaringan dari sekelompok unit pemroses kecil yang dimodelkan berdasarkan sistem saraf manusia. JST merupakan sistem adaptif yang dapat mengubah strukturnya untuk memecahkan masalah berdasarkan informasi eksternal maupun internal yang mengalir melalui jaringan tersebut. Oleh karena sifatnya yang adaptif, JST juga sering disebut dengan jaringan adaptif. Ilustrasinya adalah sebagai berikut :





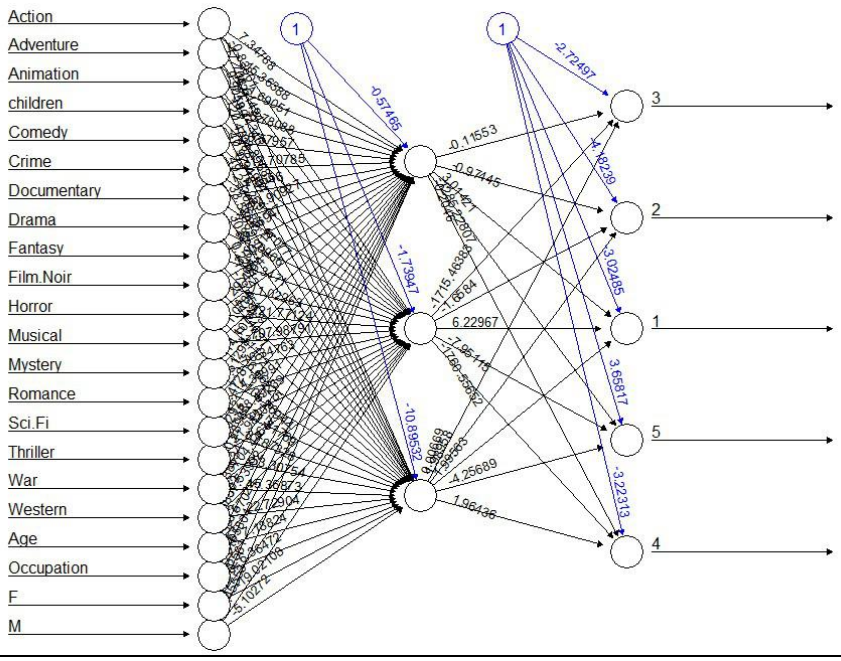
PROSES KLASIFIKASI (NN)

Dengan menggunakan tools R, library yang bisa digunakan untuk membuat NN adalah nnet() dan neuralnet(). Neural Network yang terbentuk dibawah memiliki jumlah hidden node sebanyak 3. Input pada jaringan tersebut adalah variabel-variebel independent yang digunakan(Genre, Gender, Age, Occupation), dan ouputnya berupa nilai rating baik 1, 2, 3, 4 maupun 5.

Syntax

```
#---------------NEURAL NETWORK---------#
install.packages("neuralnet")
library(nnet)
library(neuralnet)
nn <- nnet(Rating~., data=train3, size=2, maxit = 500)
view(nn)
#npredict <- predict(nn, test3, type = 'c')
library(dplyr)
names(train3)[names(train3) == "Children's"] <- "children"
names(train3)[names(train3) == "Film-Noir"] <- "Film.Noir"
names(train3)[names(train3) == "Sci-Fi"] <- "Sci.Fi"
nn1 <- neuralnet(Rating~., train3, hidden = 3, linear.output = F)
plot(nn1)
```

Hasil



EVALUASI MODEL (NN)

Evaluasi kinerja dari model pengklasifikasi dilakukan dengan menggunakan repeated hold-out sebanyak 100 kali putaran dan 10-fold cross validation .

REPEATED HOLDOUT (NN)

Metode holdout dapat dibuat lebih handal dengan mengulang proses untuk sub sampel yang berbeda. Pada setiap iterasi, suatu proporsi tertentu dipilih secara acak untuk training. Kinerja pada setiap iterasi dirata-rata untuk mendapatkan estimasi kinerja total.

Untuk melakukan mekanisme evaluasi kinerja tersebut, maka dapat menggunakan library rminer dengan membagi data sebesar 2 banding 3. Kemudian dijalankan syntax sebagai berikut.

Syntax

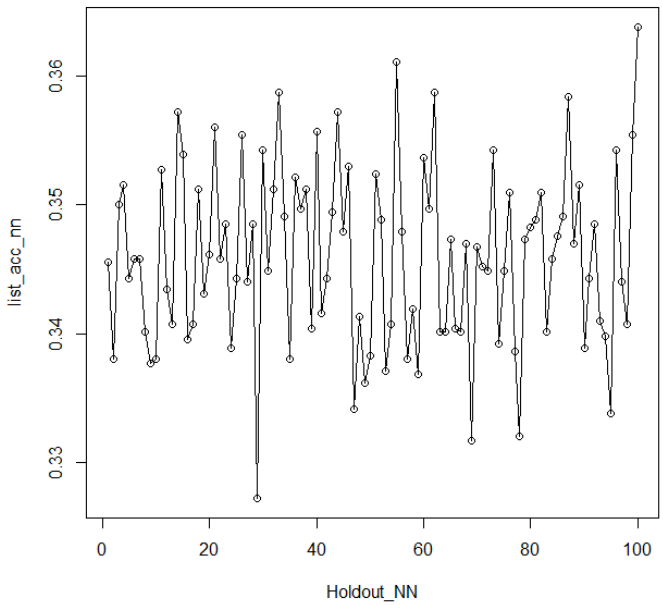
```
#Hold out NeuralNet
sample_data$Rating <- as.factor(sample_data$Rating)
sample_data$Gender <- as.numeric(sample_data$Gender)
sample_data$Age <- as.numeric(sample_data$Age)
sample_data$Occupation <- as.numeric(sample_data$Occupation)
library(nnet)
full_accuracy_nn = 0
list_acc_nn <- list()
for(b in 1:100){
  H = holdout(data_sample$Rating, ratio = 2/3, mode = 'random', seed = NULL)
  modelnn <- nnet(Rating~., data=data_sample[H$tr,], size = 3, maxit = 200, trace = FALSE)
  modelnn_predict <- predict(modelnn, data_sample[H$ts, -23], type='class')
  nnresult <- confusionMatrix(
    factor(modelnn_predict, levels = 1:5),
    factor(data_sample[H$ts,]$Rating, levels = 1:5))
  accuracy <- nnresult$overall['Accuracy']
  cat("Batch :",b,
      "accuracy :",accuracy,"\n")
  full_accuracy_nn = full_accuracy_nn+accuracy
  list_acc_nn[[b]] <- accuracy
  View(list_acc_nn)
}

cat("NN :",
    "accuracy:", full_accuracy_nn /100, "\n")

Holdout_NN <- c(1:100)
plot(Holdout_NN, list_acc_nn, type="o")
```

Hasil

Iterasi

```
Batch : 81 accuracy : 0.3488302
Batch : 82 accuracy : 0.3509298
Batch : 83 accuracy : 0.340132
Batch : 84 accuracy : 0.3458308
Batch : 85 accuracy : 0.3476305
Batch : 86 accuracy : 0.3491302
Batch : 87 accuracy : 0.3584283
Batch : 88 accuracy : 0.3470306
Batch : 89 accuracy : 0.3515297
Batch : 90 accuracy : 0.3389322
Batch : 91 accuracy : 0.3443311
Batch : 92 accuracy : 0.3485303
Batch : 93 accuracy : 0.3410318
Batch : 94 accuracy : 0.339832
Batch : 95 accuracy : 0.3338332
Batch : 96 accuracy : 0.3542292
Batch : 97 accuracy : 0.3440312
Batch : 98 accuracy : 0.3407319
Batch : 99 accuracy : 0.3554289
Batch : 100 accuracy : 0.3638272
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
NN: Accuracy : 0.3459448
```

CROSS VALIDATION (NN)

Metode cross-validation akan menghindari tumpang tindih pada data testing. Terdiri dari 2 tahap yaitu membagi data menjadi k bagian dengan ukuran yang sama dan menggunakan masing-masing bagian untuk testing, sisanya sebagai training.

Dengan bantuan tools R, berikut merupakan syntax yang perlu dijalankan.

Syntax

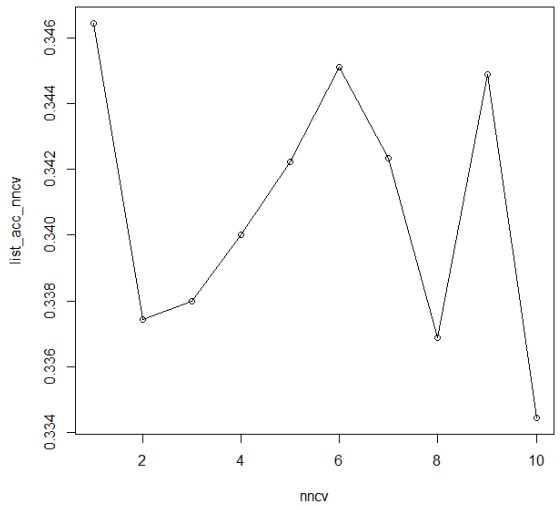
```
#10 fold Cross Validation NeuralNet
str(data_sample)
data_sample$Rating <- as.factor(data_sample$Rating)
foldnn <- cut(seq(1,nrow(data_sample)), breaks=10, labels = FALSE)
full_accuracy_nn_cv = 0
list_acc_nncv <- list()
for(i in 1:10) {
  testIndexes <- which(foldnn==i, arr.ind = TRUE)
  trainData <- data_sample[testIndexes, ]
  testData <- data_sample[-testIndexes, ]
  nnmodel <- nnet(Rating~., data=trainData, size = 3, maxit = 200, trace = FALSE)
  nnpredict <- predict(nnmodel, testData[,-23], type="class")
  result_nn <- confusionMatrix(
    factor(nnpredict, levels = 1:5),
    factor(testData$Rating, levels = 1:5))
  accuracy <- result_nn$overall['Accuracy']
  cat("batch :",i,
      "accuracy :",accuracy,"\n")
  full_accuracy_nn_cv = full_accuracy_nn_cv+accuracy
  list_acc_nncv[[i]] <- accuracy
}
cat("neural net cv :",
    "Accuracy :",full_accuracy_nn_cv/10,"\n")

nncv <- c(1:10)
plot(nncv, list_acc_nncv, type = "o")
```

Hasil

Iterasi

```
batch : 1 accuracy : 0.3464444
batch : 2 accuracy : 0.3374444
batch : 3 accuracy : 0.338
batch : 4 accuracy : 0.34
batch : 5 accuracy : 0.3422222
batch : 6 accuracy : 0.3451111
batch : 7 accuracy : 0.3423333
batch : 8 accuracy : 0.3368889
batch : 9 accuracy : 0.3448889
batch : 10 accuracy : 0.3344444
```

Plot Nilai Akurasi



Nilai Akurasi Keseluruhan

```
neural net cv : Accuracy : 0.3407778
```

Hasil dari keduanya (repeated holdout dan cross validation) tidak memilki perbedaan yang terlalu signifikan pada nilai akurasi yang dihasilkan, yaitu sekitar 34%.

BAB IV: HASIL PENGUJIAN MODEL

PERHITUNGAN

Dari model yang telah terbentuk kemudian dilakukan perhitungan nilai *accuracy, recall, precision* dan *F-measure*. Perhitungan tersebut diambil dari *confusion matrix* yang telah dihasilkan untuk masing-masing model yang terbentuk (Decision Tree(CART & C4.5), naïve bayes, k-nearest neighbor dan neural network.

DECISION TREE

CART

Berikut merupakan hasil perhitungan nilai accuracy, recall, precision dan F-measure untuk model decision tree yang dibuat dengan metode CART.

Syntax

Accuracy

```
cmcartG <- confusionMatrix(table(test_pred_cartG, test2$Rating))
cmcartG$overall[1]
```

Precision, Recall, F-Measure

```
cmcartG$byClass
```

Hasil

Accuracy	Precision	Recall	F-Measure
Accuracy 0.363	Precision NA NA NA 0.363 NA	Recall 0 0 0 1 0	F1 NA NA NA 0.5326486 NA

Hasil akurasi yang dihasilkan untuk model tersebut adalah **0,3225.** Sedangkan nilai precision, recall, dan f-measure ditampilkan pada tabel terlampir. Nilai NA dan NaN yang keluar pada precision dan F-Measure menunjukkan bahwa data yang diklasifikasi *unbalanced*. Yang dimaksud data *unbalanced* adalah adanya kelas yang kosong, tidak ada objek yang masuk di dalam kelas hasil klasifikasi tersebut.

C4.5

Berikut merupakan hasil perhitungan nilai accuracy, recall, precision dan F-measure untuk model decision tree yang dibuat dengan metode C4.5.

Syntax

Accuracy

```
cmc45 <- confusionMatrix(table(c45predict, test$Rating))
cmc45$overall[1]
```

Precision, Recall, F-Measure

```
cmc45$byClass
```

Hasil

Accuracy	Precision	Recall	F-Measure
Accuracy 0.3176667	Precision 0.1235955 0.1390374 0.2649007 0.3672026 0.3274336	Recall 0.07236842 0.07902736 0.21361816 0.52433425 0.27165932	F1 0.09128631 0.10077519 0.23651146 0.43192133 0.29695024

Hasil akurasi yang dihasilkan untuk model tersebut adalah **0,317667.** Sedangkan nilai precision, recall, dan f-measure ditampilkan pada tabel terlampir.

NAÏVE BAYES

Berikut merupakan hasil perhitungan nilai accuracy, recall, precision dan F-measure untuk model naïve bayes yang terbentuk dengan melakukan pengujian terhadap test set.

Syntax

Accuracy

```
cmnb <- confusionMatrix(table(nbpredict, test3$Rating))
cmnb$byClass
```

Precision, Recall, F-Measure

```
cmnb$byClass
```

Hasil

Accuracy	Precision	Recall	F-Measure
Accuracy 0.211	Precision 0.0839895 0.1162304 0.2839506 0.3744681 0.3557692	Recall 0.36994220 0.33134328 0.06149733 0.24742268 0.21861152	F1 0.1368984 0.1720930 0.1010989 0.2979684 0.2708143

Hasil akurasi yang dihasilkan untuk model tersebut adalah **0,211.** Sedangkan nilai precision, recall, dan f-measure ditampilkan pada tabel terlampir.

KNN

Berikut merupakan hasil perhitungan nilai accuracy, recall, precision dan F-measure untuk model KNN yang terbentuk dengan melakukan pengujian terhadap test set.

Syntax

Accuracy

```
cmknn <- confusionMatrix(table(knnpredict2, test3$Rating))
cmknn$overall[1]
```

Precision, Recall, F-Measure

```
cmknn$byClass
```

Hasil

Accuracy	Precision	Recall	F-Measure
Accuracy 0.315	Precision 0.1052632 0.1932773 0.2617047 0.3613559 0.3121495	Recall 0.02312139 0.06865672 0.29144385 0.49953140 0.24667651	F1 0.03791469 0.10132159 0.27577483 0.41935484 0.27557756

Hasil akurasi yang dihasilkan untuk model tersebut adalah **0,315.** Sedangkan nilai precision, recall, dan f-measure ditampilkan pada tabel terlampir.

NEURAL NETWORK

Berikut merupakan hasil perhitungan nilai accuracy, recall, precision dan F-measure untuk model Neural Network yang terbentuk dengan melakukan pengujian terhadap test set.

Syntax

Accuracy

```
nnresult <- confusionMatrix(
  factor(modelnn_predict, levels = 1:5),
  factor(test3$Rating, levels = 1:5))
nnresult$byClass
```

Precision, Recall, F-Measure

```
nnresult$byClass
```

Hasil

Accuracy	Precision	Recall	F-Measure
Accuracy 0.351	Precision NA 0.1666667 0.2669323 0.3590959 NA	Recall 0.000000000 0.002985075 0.089572193 0.923149016 0.000000000	F1 NA 0.005865103 0.134134134 0.517060367 NA

Hasil akurasi yang dihasilkan untuk model tersebut adalah **0,351.** Sedangkan nilai precision, recall, dan f-measure ditampilkan pada tabel terlampir. Nilai NA dan NaN yang keluar pada precision dan F-Measure menunjukkan bahwa data yang diklasifikasi *unbalanced*. Yang dimaksud data *unbalanced* adalah adanya kelas yang kosong, tidak ada objek yang masuk di dalam kelas hasil klasifikasi tersebut.

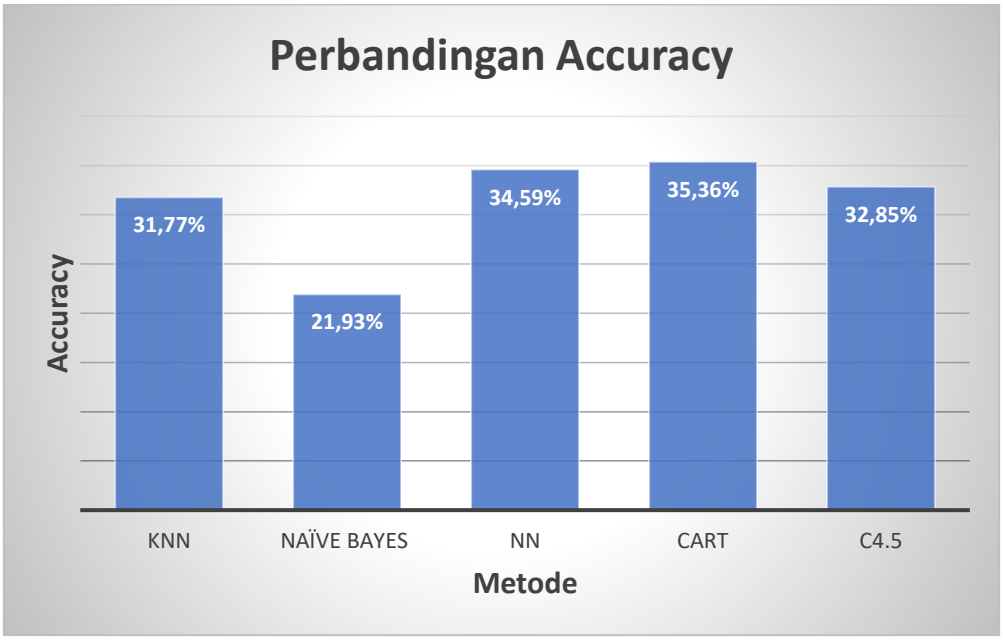
ANALISIS

Accuracy Setiap Model

Di bawah ini merupakan tabel perbandingan tingkat akurasi model yang didapatkan dari setiap metode. Nilai akurasi akhir dari setiap metode didapatkan dari perhitungan rata-rata setiap akurasi dari data testing, repeated holdout, dan cross-validation. Berdasarkan dari hasil rata-rata akurasi dari setiap metode, dapat disimpulkan bahwa metode Decision Tree (CART) memiliki tingkat akurasi yang lebih tinggi sebesar 35,36% dibandingkan dengan metode lainnya. Tingkat akurasi yang paling kecil dimiliki oleh metode Bayesian Classifier sebesar 21,93%.

	Nilai Akurasi			
	Hasil testing (70:30)	**Repeated Holdout**	**Cross-validation**	**Rata-Rata**
DECISION TREE (CART)	0,363	0,3489709	0,348895	35,36%
DECISION TREE (C45)	0,3176667	0,354814	0,3129	32,85%
KNN	0,315	0,3196041	0,3185	31,77%
BAYESIAN CLASSIFIER	0,211	0,2406989	0,2062857	21,93%
NEURAL NETWORK	0,351	0,3459448	0,3407778	34,59%

Berikut ini adalah grafik yang memperlihatkan perbandingan tingkat akurasi dari setiap metode. Semakin tinggi hasil akurasi maka dapat dikatakan bahwa semakin akurat dan bagus model yang didapatkan dari suatu metode tersebut.



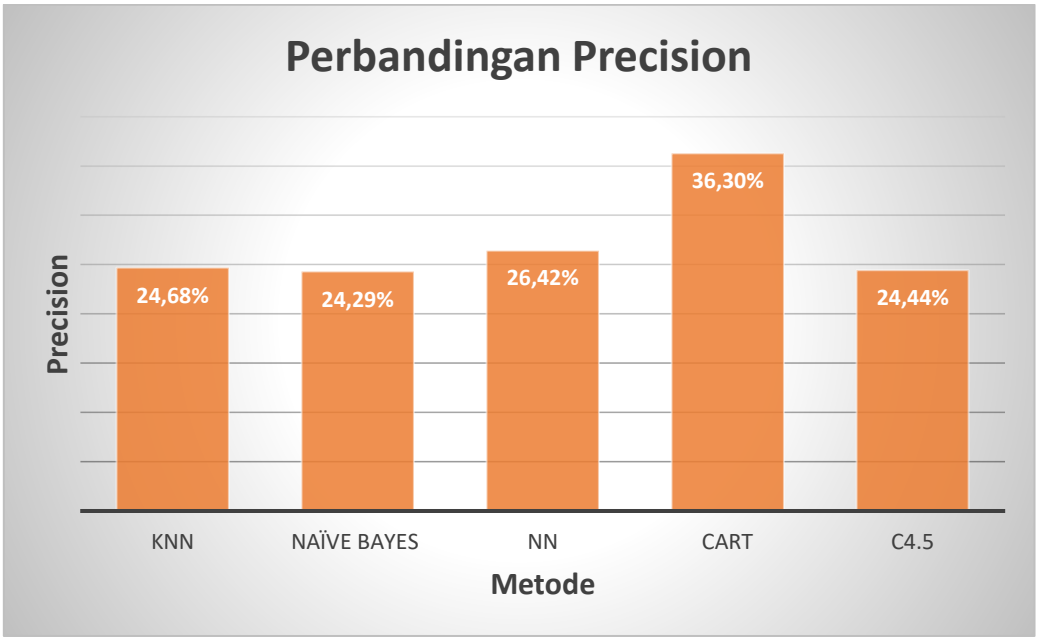
Precision Setiap Metode

Nilai precision merupakan tingkat ketepatan atau keakuratan antara informasi yang dibutuhkan dengan respon yang diberikan sistem. Di bawah ini merupakan tabel perbandingan nilai precision dari setiap metode.

Dapat disimpulkan bahwa nilai precision yang paling tinggi yaitu metode Decision Tree (CART) dengan tingkat akurasi sebesar 36,3%. Tingkat akurasi yang paling rendah yaitu terdapat pada metode Naïve Bayes (Bayesian Classifier) dengan nilai sebesar 24,29%.

	Precision
KNN	24,68%
Naïve Bayes	24,29%
NN	26,42%
CART	36,30%
C4.5	24,44%

Perbandingan nilai precision dari setiap metode juga ditampilkan dalam bentuk grafik dimana terlihat jelas bahwa niai precision dimiliki oleh metode Decision Tree (CART).

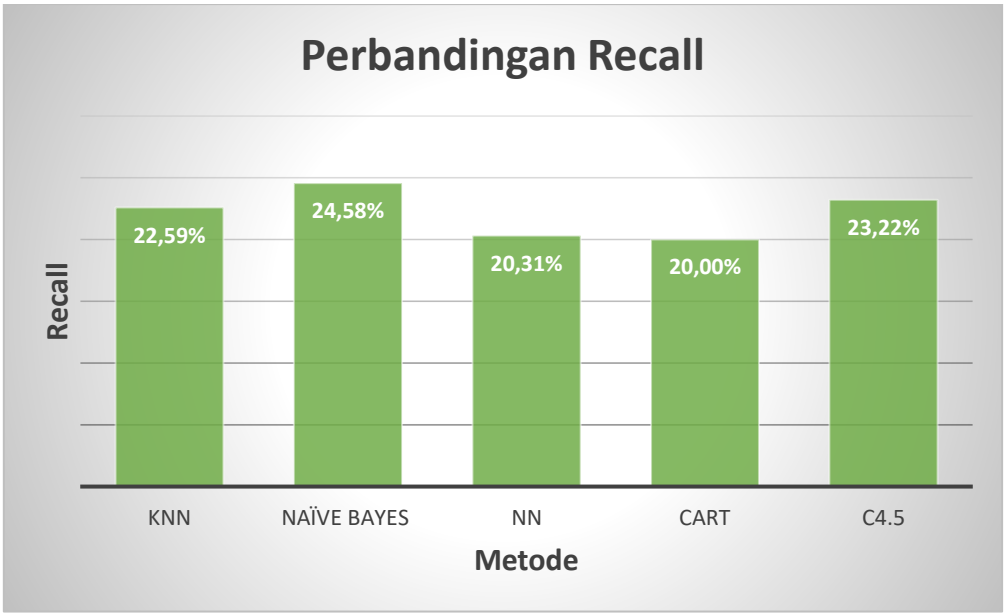


Recall Setiap Metode

Nilai recall adalah suatu nilai yang merepresentasikan tingkat keberhasilan suatu sistem dalam menemukan kembali informasi. Di bawah ini merupakan perbandingan nilai recall dari setiap metode yang dihitung dalam bentuk persentase. Dapat disimpulkan bahwa nilai recall paling tinggi dimiliki oleh metode Naïve Bayes sebesar 24,58%. Nilai recall yang paling kecil dimiliki oleh metode Decision Tree (CART) dengan nilai sebesar 20,00%.

	Recall
KNN	22,59%
Naïve Bayes	24,58%
NN	20,31%
CART	20,00%
C4.5	23,22%

Perbandingan nilai recall dari setiap metode juga ditampilkan dalam bentuk grafik dimana terlihat jelas bahwa nilai recall paling tinggi dimiliki oleh metode Naïve Bayes.



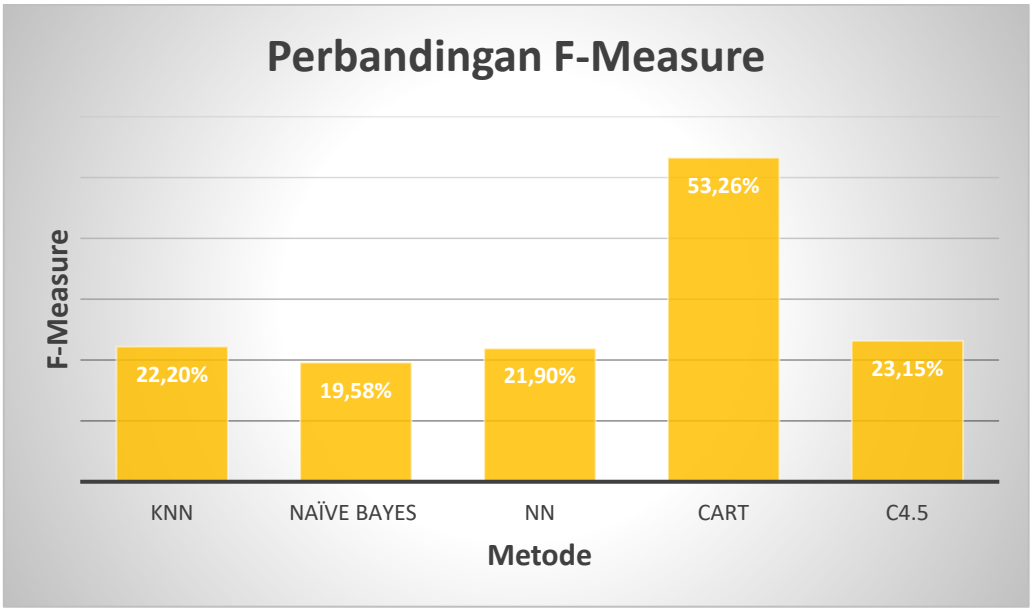
F-Measure Setiap Metode

Nilai F-Measure merupakan suatu nilai atau bobot rata-rata dari nilai precision dan recall. F-Measure biasanya digunakan untuk melakukan perhitungan evaluasi yang mengkombinasikan antara recall dan precission. Di bawah ini merupakan perbandingan nilai F-Measure dari setiap metode dalam bentuk persentase.

Dapat disimpulkan bahwa nilai F-Measure yang paling tinggi dimiliki oleh metode Decision Tree (CART) dengan nilai sebesar 53,26%. Nilai F-Measure yang paling kecil dimiliki oleh metode Naives Bayes (Bayesian Classifier) dengan nilai sebesar 19,58%.

	F-Measure
KNN	22,20%
Naïve Bayes	19,58%
NN	21,90%
CART	53,26%
C4.5	23,15%

Perbandingan nilai F-Measure dari setiap metode juga ditampilkan dalam bentuk grafik dimana terlihat jelas bahwa nilai F-Measure paling tinggi dimiliki oleh metode Decision Tree (CART).



BAB V : KESIMPULAN & SARAN

KESIMPULAN

Pada hasil pengerjaan tentang klasifikasi data ini maka dapat disimpulkan bahwa model yang dihasilkan dari metode Decision Tree (CART) memilki tingkat akurasi sebesar 35,36%. Nilai itu termasuk nilai akurasi yang paling tinggi dibandingkan dengan metode lainnya. Selain itu, model ini juga memiliki nilai precision (36,30%), dan f-measure (53,26%) yang lebih bagus dari pada metode yang lain. Namun, nilai recall untuk metode Decision Tree (CART) memiliki nilai yang paling rendah sebesar 20,00% sedangkan yang paling tinggi dimiliki oleh metode Naïve Bayes sebesar 24,58%.

SARAN

Berdasarkan hasil pengerjaan beberapa metode klasifikasi di atas, terdapat beberapa saran yaitu :

- Parameter yang ada di metode Neural Network dilakukan berbagai percobaan modifikasi di bagian hidden layer yaitu dengan cara melakukan optimasi agar tingkat akurasi yang didapatkan lenih baik lagi.
- Pada data klasifikasi tersebut perlu dilakukan analisis variabel sehingga diketahui variabel mana saja yang paling memiliki korelasi tinggi terhadap variabel Rating. Adanya analisis ini dapat meningkatkan tingkat akurasi dan memperkecil error.