TUGAS BESAR DATA MINING

EVALUATION CAR MENGGUNAKAN METODE CLASSIFICATION AND REGRESSION TREESS (CART)

DISUSUN OLEH :

Erna Nadira - 3311901008

Ana Jannatu Uzlifat - 3311901011

Tasya Selvia Ulfa - 3311901018

POLITEKNIK NEGERI BATAM

JURUSAN TEKNIK INFORMATIKA

PRODI TEKNIK INFORMATIKA

2021

- Alat :

  R dan R Studio

- Metode

  CART (*Classification and Regression Tree*) merupakan metode eksplorasi yang digunakan untuk melihat hubungan antara variabel respon dengan variabel bebas yang meliputi variabel nominal, ordinal, maupun kontinu. Metode ini meliputi metode pohon klasifikasi dan pohon regresi.

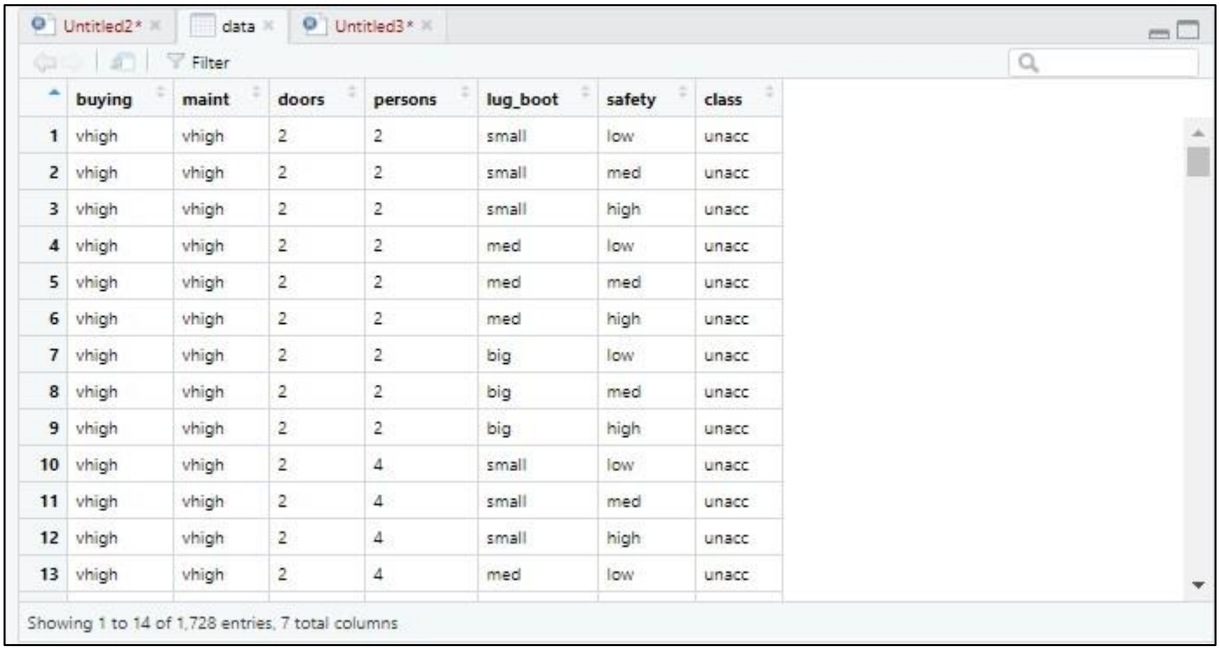
  CART merupakan pengembangan dari pohon keputusan. Metode ini merupakan metode eksplorasi yang mengubah data yang sangat besar menjadi pohon keputusan yang merepresentasikan suatu aturan. CART yaitu metode yang digunakan untuk mengelompokkan data secara berulang untuk mengestimasi distribusi kondisional dari data (pada kasus ini adalah variabel respon) jika diberikan variabel bebasnya (penjelasnya).

- Dataset :

  Evaluation Car Database berasal dari model keputusan hierarki sederhana yang awalnya dikembangkan untuk demonstrasi DEX, M. Bohanec, V. Rajkovic: Sistem pakar untuk pengambilan keputusan. Sistemica 1 (1), hlm. 145-157, 1990.).

  Evaluation Car Database berisi contoh-contoh dengan informasi struktural yang dihilangkan, yaitu, secara langsung menghubungkan CAR dengan enam atribut input: buying, maint, doors, persons, lug_boot, safety . Karena struktur konsep yang mendasari diketahui, database ini mungkin sangat berguna untuk menguji induksi konstruktif dan metode penemuan struktur.

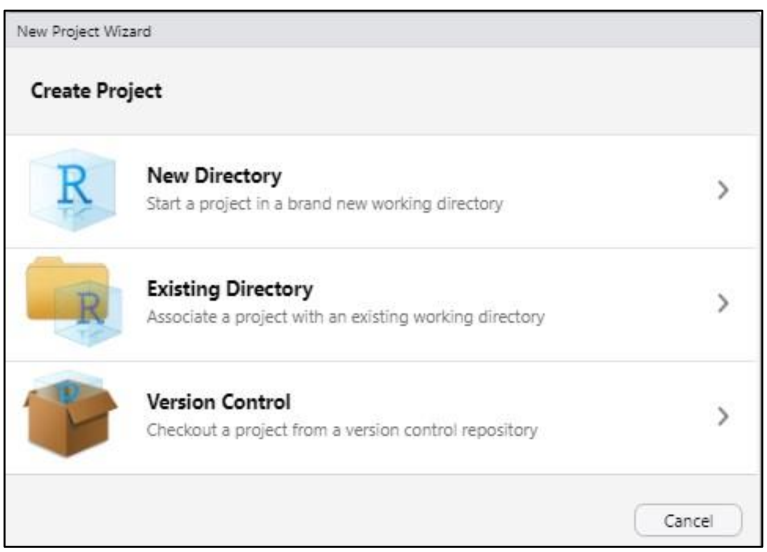
Jumlah Baris Data	: 1728
Data yang Hilang	: Tidak Ada
Jumlah Atribut	: 6 ditambah target 1 , total atribut 7
Nilai Class (target)	: unacc, acc, good, vgood
Atribut	: buying, maint, doors, persons, lug_boot, safety



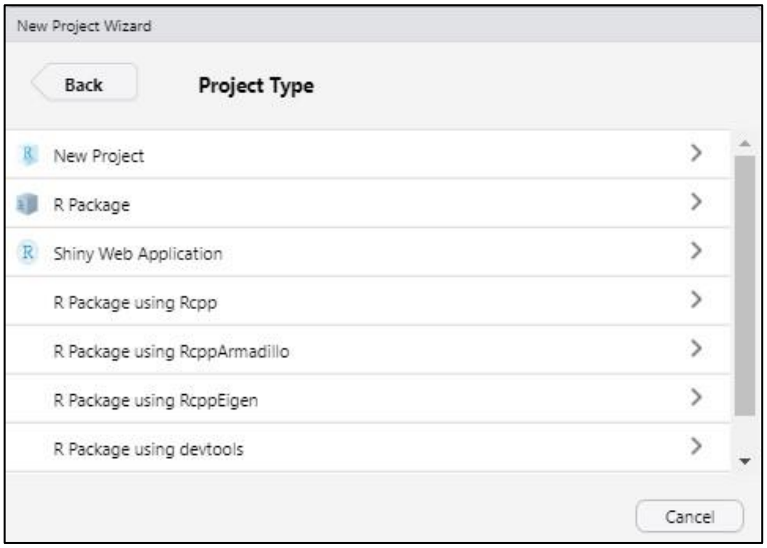
	buying	maint	doors	persons	lug_boot	safety	class
1	vhigh	vhigh	2	2	small	low	unacc
2	vhigh	vhigh	2	2	small	med	unacc
3	vhigh	vhigh	2	2	small	high	unacc
4	vhigh	vhigh	2	2	med	low	unacc
5	vhigh	vhigh	2	2	med	med	unacc
6	vhigh	vhigh	2	2	med	high	unacc
7	vhigh	vhigh	2	2	big	low	unacc
8	vhigh	vhigh	2	2	big	med	unacc
9	vhigh	vhigh	2	2	big	high	unacc
10	vhigh	vhigh	2	4	small	low	unacc
11	vhigh	vhigh	2	4	small	med	unacc
12	vhigh	vhigh	2	4	small	high	unacc
13	vhigh	vhigh	2	4	med	low	unacc

a. Persiapkan Rstudio dan Dataset

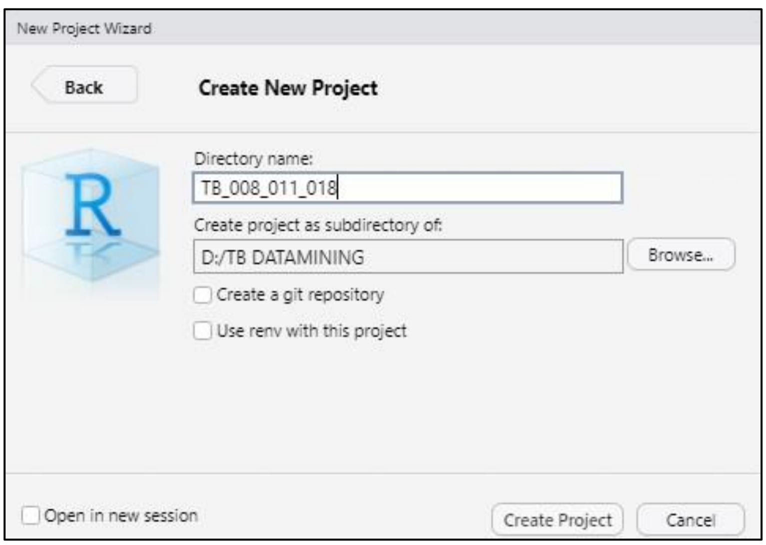
1. Buat sebuah project baru



2. Pilih New Directory

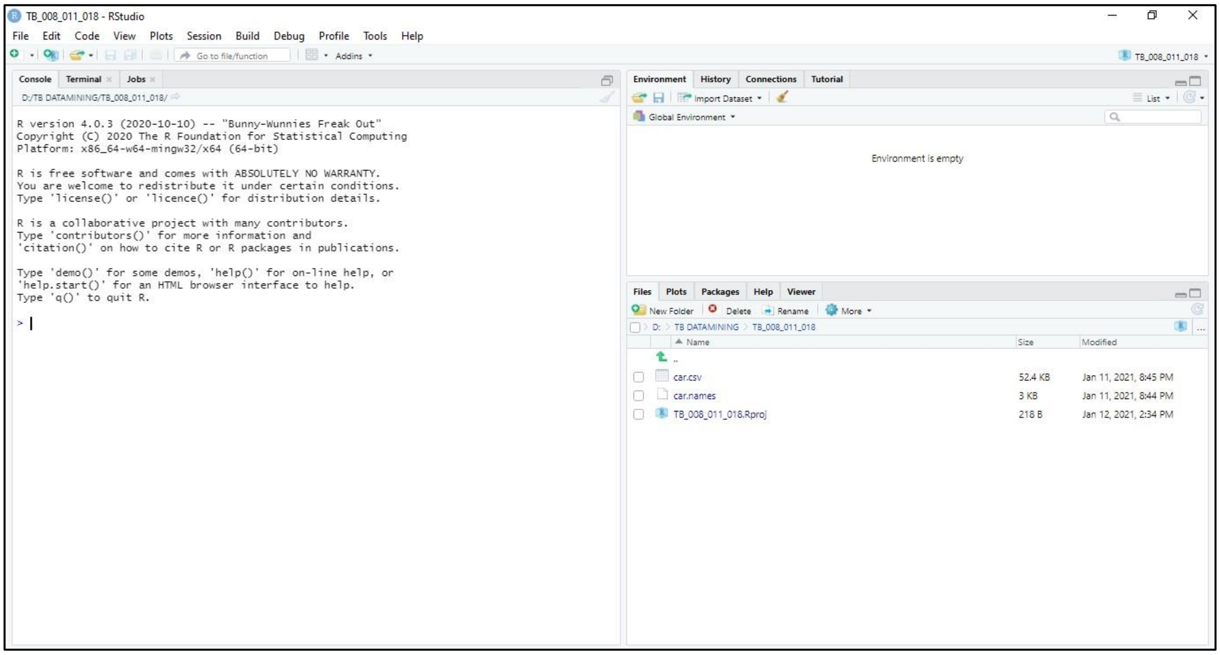


3. Lalu pilih New Project , beri nama direktori , lalu simpan di D:/TB DATAMINING



b. Lakukan preprosesing data

1. Buat script baru



2. Lihat lokasi direktori dengan perintah **`getwd()`**

```
> getwd()
[1] "D:/TB DATAMINING/TB_008_011_018"
>
```

3. Import dataset dengan perintah **`data.car <- read.csv("car.csv", sep = ",")`**

   `data.car = nama untuk data`

   `"car.csv" = nama file data yang akan kita import`

   `sep = "," artinya setiap kolom dibatasi oleh koma (sebagai pembeda)`

```
getwd()
data.car <- read.csv("car.csv", sep = ",")
```

4. data.car berhasil di import , lihat pada panel kanan akan menampilkan :

```
Data
data.car                        1728 obs. of 7 variables
```

5. lihat jumlah baris data dan kolom dengan perintah **`dim(data.car)`**

```
> dim(data.car)
[1] 1728    7
> |
```

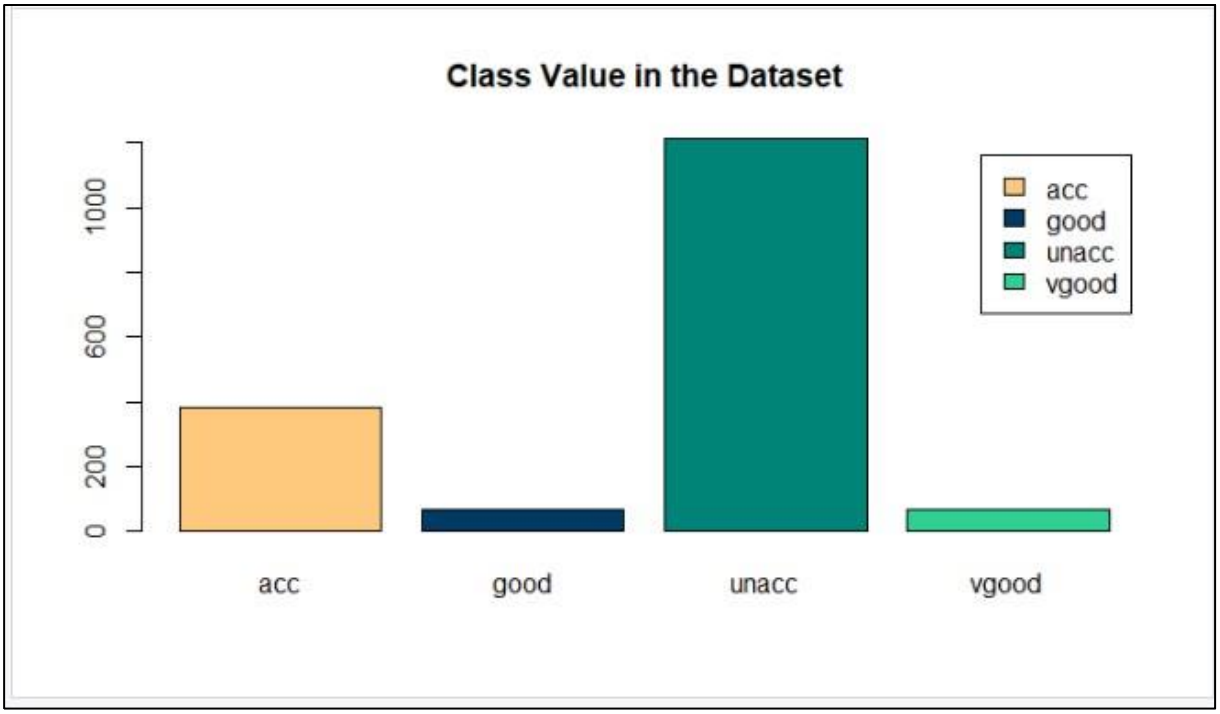
6. lihat informasi data dengan perintah str(data.car) maka akan menunjukan jumlah baris data , jumlah kolom (variable) , nama kolom (variable) , tipe data kolom(variable) , dan beberapa contoh isi dari masing – masing kolom (variable)

```
> str(data.car)
'data.frame':   1728 obs. of  7 variables:
 $ buying  : chr  "vhigh" "vhigh" "vhigh" "vhigh" ...
 $ maint   : chr  "vhigh" "vhigh" "vhigh" "vhigh" ...
 $ doors   : chr  "2" "2" "2" "2" ...
 $ persons : chr  "2" "2" "2" "2" ...
 $ lug_boot: chr  "small" "small" "small" "med" ...
 $ safety  : chr  "low" "med" "high" "low" ...
 $ class   : chr  "unacc" "unacc" "unacc" "unacc" ...
> |
```

7. Untuk mengetahui jumlah tiap tiap target yang ada pada variable class yaitu dengan perintah **`counts <- table(data.car$class)`**

```
counts <- table(data.car$class)
barplot(counts,col=c("darkblue","red"),
        legend = rownames(counts),
        main = "Class Value in the Dataset")
```

8. Selanjutnya untuk melihat hasil perhitungan perintah tersebut dengan perintah **`barplot(counts,col=c("#ffc97d","#003b66","#008478","#2ed194"),legend = rownames(counts), main = "Class Value in the Dataset")`**



c. Mulai melakukan pembuatan model dan pohon keputusan

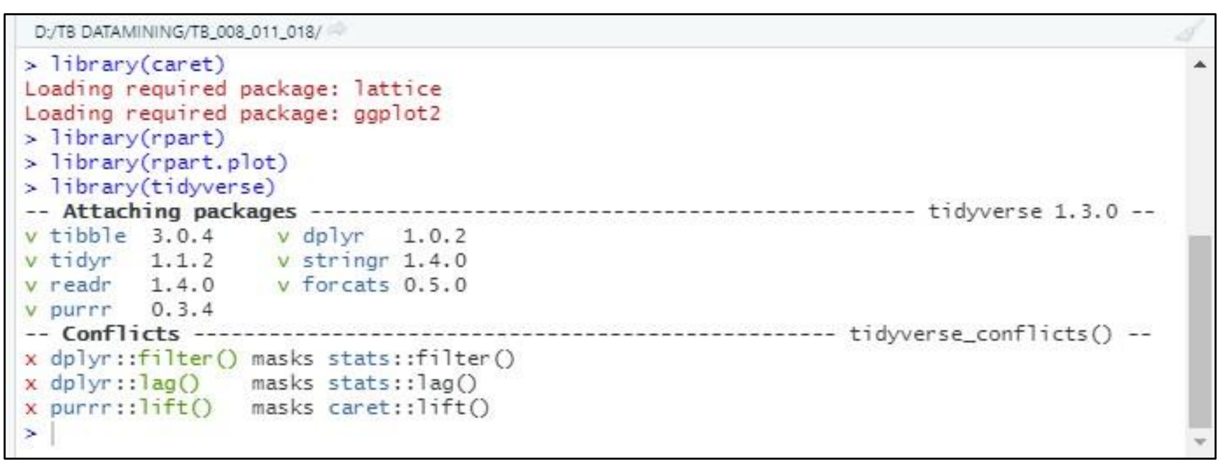
1. Tambahkan library untuk mendukung perintah yang akan digunakan pada pemprosesan data yang ada . libarary yang di butuhkan :

   library(caret)

   library(rpart)

   library(rpart.plot)

   library(tidyverse)



```
D:/TB DATAMINING/TB_008_011_018/
> library(caret)
Loading required package: lattice
Loading required package: ggplot2
> library(rpart)
> library(rpart.plot)
> library(tidyverse)
-- Attaching packages --------------------------------------------- tidyverse 1.3.0 --
v tibble  3.0.4     v dplyr   1.0.2
v tidyr   1.1.2     v stringr 1.4.0
v readr   1.4.0     v forcats 0.5.0
v purrr   0.3.4
-- Conflicts ------------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
x purrr::lift()   masks caret::lift()
>
```

2. Lihat beberapa sample data yang ada dengan perintah **`sample_n(data.car,10)`**

```
> sample_n(data.car,10)
   buying maint doors persons lug_boot safety class
1   vhigh   low 5more    more    small    low unacc
2     low  high     3       4      med   high   acc
3     low   med     4       4      big    low unacc
4     low  high     4       2    small    med unacc
5    high   low 5more       2    small    med unacc
6   vhigh vhigh     3       4    small   high unacc
7     low vhigh     4    more    small    med unacc
8   vhigh vhigh     3       2    small    med unacc
9    high   med     4    more      med    med   acc
10   high   low     4    more      med    low unacc
> |
```

3. Acak sample data dengen perintah **`set.seed(123)`**
4. Bagi data sample yang ada menjadi 80 % untuk data train dan 20 % untuk data test dengan perintah **`training.samples <- data.car$class %>%`**

    `createDataPartition(p = 0.8, list = FALSE)`

    `train.data <- data.car[training.samples, ]`

    `test.data <- data.car[-training.samples, ]`

```
set.seed(123)
training.samples <- data.car$class %>%
  createDataPartition(p = 0.8, list = FALSE)
train.data <- data.car[training.samples, ]
test.data <- data.car[-training.samples, ]
```

5. Lihat data training dan data testing

`view(train.data)` untuk melihat data train , data train memiliki jumlah 1384 dan 7 variable atau data train memiliki 80% data dari data sample yang ada car_008_011_018.r* | train.data

Filter

	buying	maint	doors	persons	lug_boot	safety	class
1	vhigh	vhigh	2	2	small	low	unacc
2	vhigh	vhigh	2	2	small	med	unacc
4	vhigh	vhigh	2	2	med	low	unacc
5	vhigh	vhigh	2	2	med	med	unacc
6	vhigh	vhigh	2	2	med	high	unacc
8	vhigh	vhigh	2	2	big	med	unacc
9	vhigh	vhigh	2	2	big	high	unacc
11	vhigh	vhigh	2	4	small	med	unacc
12	vhigh	vhigh	2	4	small	high	unacc
14	vhigh	vhigh	2	4	med	med	unacc
17	vhigh	vhigh	2	4	big	med	unacc
18	vhigh	vhigh	2	4	big	high	unacc
19	vhigh	vhigh	2	more	small	low	unacc
20	vhigh	vhigh	2	more	small	med	unacc
21	vhigh	vhigh	2	more	small	high	unacc
23	vhigh	vhigh	2	more	med	med	unacc
24	vhigh	vhigh	2	more	med	high	unacc
25	vhigh	vhigh	2	more	big	low	unacc

Showing 1 to 19 of 1,384 entries, 7 total columns

view(test.data) untuk melihat data test , data test memiliki jumlah 344 dan 7 variable atau data test memiliki 20% data dari data sample yang ada

car_008_011_018.r* | test.data | train.data | datatesting

Filter

	buying	maint	doors	persons	lug_boot	safety	class
3	vhigh	vhigh	2	2	small	high	unacc
7	vhigh	vhigh	2	2	big	low	unacc
10	vhigh	vhigh	2	4	small	low	unacc
13	vhigh	vhigh	2	4	med	low	unacc
15	vhigh	vhigh	2	4	med	high	unacc
16	vhigh	vhigh	2	4	big	low	unacc
22	vhigh	vhigh	2	more	med	low	unacc
27	vhigh	vhigh	2	more	big	high	unacc
31	vhigh	vhigh	3	2	med	low	unacc
43	vhigh	vhigh	3	4	big	low	unacc
47	vhigh	vhigh	3	more	small	med	unacc
50	vhigh	vhigh	3	more	med	med	unacc
53	vhigh	vhigh	3	more	big	med	unacc
57	vhigh	vhigh	4	2	small	high	unacc
62	vhigh	vhigh	4	2	big	med	unacc
65	vhigh	vhigh	4	4	small	med	unacc
66	vhigh	vhigh	4	4	small	high	unacc
85	vhigh	vhigh	5more	2	med	low	unacc

Showing 1 to 19 of 344 entries, 7 total columns

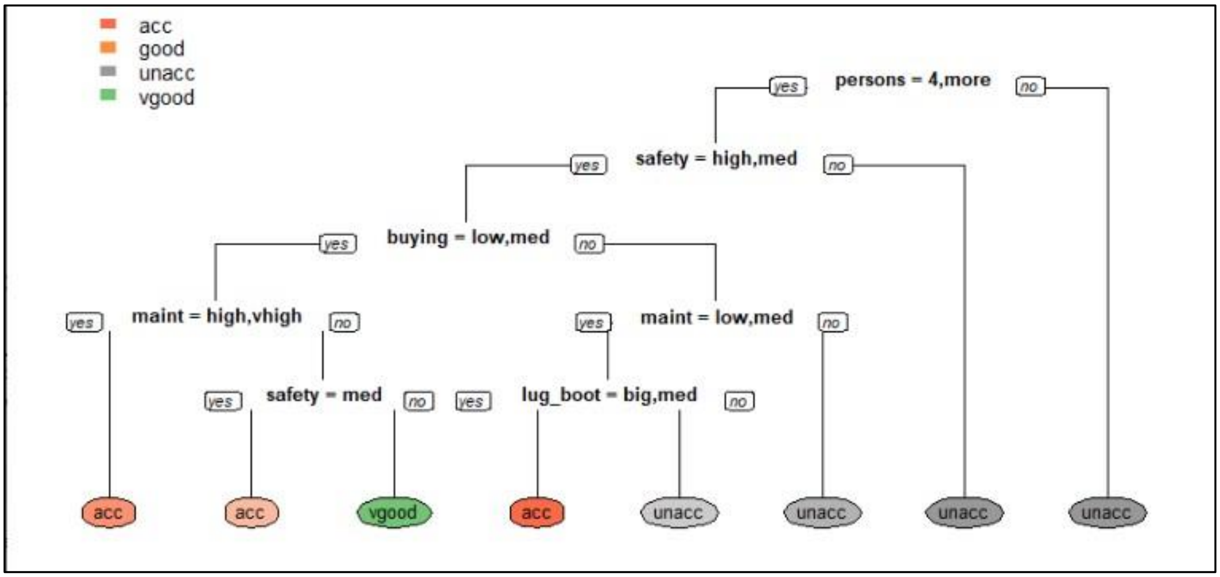
6. Buat model untuk membentuk decision tree (pohon keputusan) dengan menggunakan data train yang ada dan gunakan package rpart untk membentuk model tersebut .

model.class <- rpart(class ~., data = train.data, method = "class", control = rpart.control(minsplit = 100, cp =0))

```
D:/TB DATAMINING/TB_008_011_018/
> model.class <- rpart(class ~., data = train.data, method = "class",
+                      control = rpart.control(minsplit = 100, cp =0))
> rpart.plot(model.class, yesno = 2, type = 0, extra = 0)
> |
```

7. Untuk melihat hasil dari model yang sudah terbentuk dengan perintah

rpart.plot(model.class, yesno = 2, type = 0, extra = 0)



8. Melihat Root Node Eror dengan perintah **printcp(model.class)**

```
D:/TB DATAMINING/TB_008_011_018/
> printcp(model.class)

Classification tree:
rpart(formula = class ~ ., data = train.data, method = "class",
    control = rpart.control(minsplit = 100, cp = 0))

Variables actually used in tree construction:
[1] buying   lug_boot maint    persons  safety

Root node error: 416/1384 = 0.30058

n= 1384

        CP nsplit rel error  xerror     xstd
1 0.137019      0   1.00000 1.00000 0.041004
2 0.111779      2   0.72596 0.77885 0.037867
3 0.039663      4   0.50240 0.53606 0.032878
4 0.024038      6   0.42308 0.43990 0.030293
5 0.000000      7   0.39904 0.42788 0.029938
> |
```

9. Untuk melihat penjelasan mengenai model yang telah dibuat sebelumnya dengan perintah

   summary(model.class)

```
D:/TB DATAMINING/TB_008_011_018/
> rpart.plot(model.class, yesno = 2, type = 0, extra = 4)
> summary(model.class)
Call:
rpart(formula = class ~ ., data = train.data, method = "class",
    control = rpart.control(minsplit = 100, cp = 0))
  n= 1384

          CP nsplit rel error    xerror       xstd
1 0.13701923      0 1.0000000 1.0000000 0.04100369
2 0.11177885      2 0.7259615 0.7788462 0.03786728
3 0.03966346      4 0.5024038 0.5336538 0.03281846
4 0.02403846      6 0.4230769 0.4567308 0.03077638
5 0.00000000      7 0.3990385 0.4254808 0.02986613

Variable importance
  safety  persons    maint   buying lug_boot    doors
      42       25       18        9        5        1

Node number 1: 1384 observations,    complexity param=0.1370192
  predicted class=unacc  expected loss=0.300578  P(node) =1
    class counts:   308    56   968    52
   probabilities: 0.223 0.040 0.699 0.038
  left son=2 (917 obs) right son=3 (467 obs)
  Primary splits:
      persons  splits as  RLL,  improve=100.735200, (0 missing)
      safety   splits as  LRL,  improve=100.410000, (0 missing)
      buying   splits as  RLLR, improve= 17.656390, (0 missing)
      maint    splits as  RLLR, improve= 11.344240, (0 missing)
      lug_boot splits as  LLR,  improve=  8.731054, (0 missing)
```

```
D:/TB DATAMINING/TB_008_011_018/
Node number 2: 917 observations,    complexity param=0.1370192
  predicted class=unacc  expected loss=0.4536532  P(node) =0.6625723
    class counts:   308    56   501    52
   probabilities: 0.336 0.061 0.546 0.057
  left son=4 (610 obs) right son=5 (307 obs)
  Primary splits:
      safety   splits as  LRL,  improve=150.248200, (0 missing)
      buying   splits as  RLLR, improve= 26.817870, (0 missing)
      maint    splits as  RLLR, improve= 16.728220, (0 missing)
      lug_boot splits as  LLR,  improve= 12.733500, (0 missing)
      doors    splits as  RLLL, improve=  1.388815, (0 missing)

Node number 3: 467 observations
  predicted class=unacc  expected loss=0  P(node) =0.3374277
    class counts:     0     0   467     0
   probabilities: 0.000 0.000 1.000 0.000
```

```
Node number 4: 610 observations,    complexity param=0.1117788
  predicted class=acc    expected loss=0.495082  P(node) =0.4407514
    class counts:   308    56   194    52
   probabilities: 0.505 0.092 0.318 0.085
  left son=8 (308 obs) right son=9 (302 obs)
  Primary splits:
      buying   splits as  RLLR, improve=36.931360, (0 missing)
      maint    splits as  LLLR, improve=25.670540, (0 missing)
      lug_boot splits as  LLR,  improve=16.638600, (0 missing)
      safety   splits as  R-L,  improve= 9.254405, (0 missing)
      doors    splits as  RLLL, improve= 2.314267, (0 missing)
  Surrogate splits:
      doors    splits as  LRLR, agree=0.526, adj=0.043, (0 split)
      maint    splits as  LLRR, agree=0.521, adj=0.033, (0 split)
      safety   splits as  R-L,  agree=0.511, adj=0.013, (0 split)
      lug_boot splits as  LLR,  agree=0.507, adj=0.003, (0 split)
```

D:/TB DATAMINING/TB_008_011_018/

```
Node number 5: 307 observations
  predicted class=unacc  expected loss=0  P(node) =0.2218208
    class counts:     0     0   307     0
   probabilities: 0.000 0.000 1.000 0.000

Node number 8: 308 observations,    complexity param=0.03966346
  predicted class=acc    expected loss=0.461039  P(node) =0.2225434
    class counts:   166    56    34    52
   probabilities: 0.539 0.182 0.110 0.169
  left son=16 (151 obs) right son=17 (157 obs)
  Primary splits:
      maint    splits as  LRRL, improve=27.474650, (0 missing)
      safety   splits as  R-L,  improve=12.370240, (0 missing)
      lug_boot splits as  RRL,  improve= 9.018566, (0 missing)
      doors    splits as  RLLL, improve= 2.473101, (0 missing)
      buying   splits as  -RL-, improve= 2.259740, (0 missing)
  Surrogate splits:
      buying   splits as  -LR-, agree=0.529, adj=0.040, (0 split)
      doors    splits as  RRRL, agree=0.516, adj=0.013, (0 split)
      lug_boot splits as  LRR,  agree=0.516, adj=0.013, (0 split)

Node number 9: 302 observations,    complexity param=0.1117788
  predicted class=unacc  expected loss=0.4701987  P(node) =0.2182081
    class counts:   142     0   160     0
   probabilities: 0.470 0.000 0.530 0.000
  left son=18 (151 obs) right son=19 (151 obs)
  Primary splits:
      maint    splits as  RLLR, improve=46.728480, (0 missing)
      lug_boot splits as  LLR,  improve=12.491260, (0 missing)
      safety   splits as  L-R,  improve= 8.767991, (0 missing)
      buying   splits as  L--R, improve= 4.988893, (0 missing)
      doors    splits as  RLLL, improve= 1.067564, (0 missing)
  Surrogate splits:
      doors    splits as  RLLR, agree=0.520, adj=0.040, (0 split)
      lug_boot splits as  LRL,  agree=0.520, adj=0.040, (0 split)
      safety   splits as  R-L,  agree=0.517, adj=0.033, (0 split)
      buying   splits as  L--R, agree=0.507, adj=0.013, (0 split)
      persons  splits as  -LR,  agree=0.503, adj=0.007, (0 split)
```

D:/TB DATAMINING/TB_008_011_018/

```
Node number 16: 151 observations
  predicted class=acc    expected loss=0.2450331  P(node) =0.109104
    class counts:   114     0    26    11
   probabilities: 0.755 0.000 0.172 0.073 

Node number 17: 157 observations,    complexity param=0.03966346
  predicted class=good   expected loss=0.6433121  P(node) =0.1134393
    class counts:    52    56     8    41
   probabilities: 0.331 0.357 0.051 0.261 
  left son=34 (80 obs) right son=35 (77 obs)
  Primary splits:
      safety   splits as  R-L,  improve=19.238550, (0 missing)
      lug_boot splits as  RRL,  improve= 8.449274, (0 missing)
      maint    splits as  -RL-, improve= 4.592334, (0 missing)
      buying   splits as  -RL-, improve= 4.255466, (0 missing)
      doors    splits as  LRRR, improve= 1.666579, (0 missing)
  Surrogate splits:
      doors    splits as  LLRL, agree=0.522, adj=0.026, (0 split)
      lug_boot splits as  RLR,  agree=0.522, adj=0.026, (0 split)
      persons  splits as  -LR,  agree=0.516, adj=0.013, (0 split)

Node number 18: 151 observations,    complexity param=0.02403846
  predicted class=acc    expected loss=0.2516556  P(node) =0.109104
    class counts:   113     0    38     0
   probabilities: 0.748 0.000 0.252 0.000 
  left son=36 (99 obs) right son=37 (52 obs)
  Primary splits:
      lug_boot splits as  LLR,  improve=18.82561000, (0 missing)
      safety   splits as  L-R,  improve=15.53944000, (0 missing)
      doors    splits as  RLLL, improve= 0.75791600, (0 missing)
      maint    splits as  -LR-, improve= 0.06650693, (0 missing)
      persons  splits as  -RL,  improve= 0.06650693, (0 missing)

Node number 19: 151 observations
  predicted class=unacc  expected loss=0.192053  P(node) =0.109104
    class counts:    29     0   122     0
   probabilities: 0.192 0.000 0.808 0.000 
```

D:/TB DATAMINING/TB_008_011_018/

```
Node number 34: 80 observations
  predicted class=acc    expected loss=0.45  P(node) =0.05780347
    class counts:    44    32     4     0
   probabilities: 0.550 0.400 0.050 0.000 

Node number 35: 77 observations
  predicted class=vgood  expected loss=0.4675325  P(node) =0.05563584
    class counts:     8    24     4    41
   probabilities: 0.104 0.312 0.052 0.532 

Node number 36: 99 observations
  predicted class=acc    expected loss=0.07070707  P(node) =0.07153179
    class counts:    92     0     7     0
   probabilities: 0.929 0.000 0.071 0.000 

Node number 37: 52 observations
  predicted class=unacc  expected loss=0.4038462  P(node) =0.03757225
    class counts:    21     0    31     0
   probabilities: 0.404 0.000 0.596 0.000 
```

10. Lakukan uji data dengan memprediksi hasil dari model dan pohon keputusan yang terbentuk , data yang digunakan untuk memprediksi data adalah data test
11. Lihat hasil dari prediksi data dan jumlah tiap tiap class yang dihasilkan dari prediksi data tersebut .
12. Menunjukan bahwa jumlah tiap tiap class dapat terlihat yaitu

Acc = 74 baris data

Good = 13 baris data

Unacc = 242 baris data

Vgood = 15 baris data

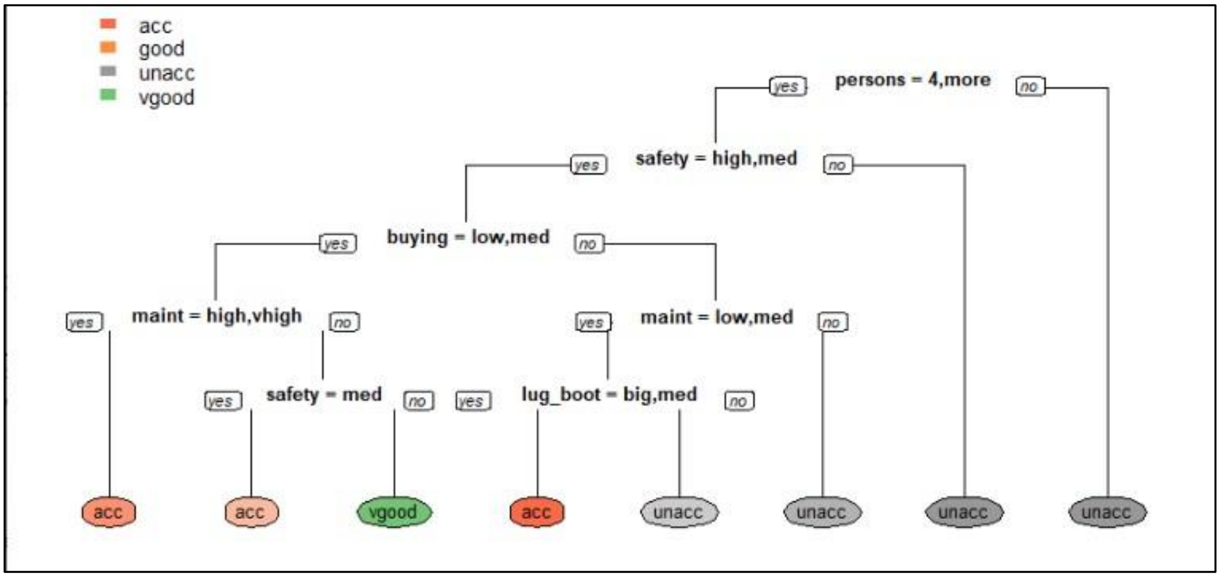
```
D:/TB DATAMINING/TB_008_011_018/
> predicted.classes <- model %>%
+   predict(test.data, type = "class")
> str(predicted.classes)
 Factor w/ 4 levels "acc","good","unacc",..: 3 3 3 3 3 3 3 3 3 3 ...
> summary(predicted.classes)
  acc  good unacc vgood
   74    13   242    15
> head(predicted.classes)
[1] unacc unacc unacc unacc unacc unacc
Levels: acc good unacc vgood
> head(test.data$class)
[1] "unacc" "unacc" "unacc" "unacc" "unacc" "unacc"
```

13. Bandingan hasil prediksi dari model dan pohon keputusan dengan data test yang ada untuk melihat keakurasian model dan pohon keputusan yang ada

Menunjukkan bahwa keakurasian dari model dan pohon keputusan = 98,8% dan tingkat terjadinya eror atau kesalahan sebesar 1,2 %

```
> mean(predicted.classes == test.data$class)
[1] 0.9883721
> mean(predicted.classes != test.data$class)
[1] 0.01162791
```

Kesimpulan



1. Aturan klasifikasi yang didapat dari pohon keputusan adalah :

- Rule 1 : IF (person “!= 4 atau more”) → **unacc**
- Rule 2 : IF (person “= 4 atau more”) ^ (safety “!= high atau med”) → **unacc**
- Rule 3 : IF (person “= 4 atau more”) ^ (safety “= high atau med”) ^ (buying “!=low atau med”) ^ (maint “!=low atau med”)→ **unacc**
- Rule 4 : IF (person “= 4 atau more”) ^ (safety “= high atau med”) ^ (buying “!=low atau med”) ^ (maint “=low atau med”) ^ (lug_boot “!=big atau med”) → **unacc**
- Rule 5 : IF (person “= 4 atau more”) ^ (safety “= high atau med”) ^ (buying “!=low atau med”) ^ (maint “=low atau med”) ^ (lug_boot “=big atau med”) → **acc**
- Rule 6 : IF (person “= 4 atau more”) ^ (safety “= high atau med”) ^ (buying “= low atau med”) ^ (maint “!= high atau highv”) ^ (safety “!= med”) → **vgood**
- Rule 7 : IF (person “= 4 atau more”) ^ (safety “= high atau med”) ^ (buying “= low atau med”) ^ (maint “!= high atau highv”) ^ (safety “= med”) → **acc**
- Rule 8 : IF (person “= 4 atau more”) ^ (safety “= high atau med”) ^ (buying “= low atau med”) ^ (maint “= high atau highv”) → **acc**

Daftar Pustaka

http://muhammadilhammubarok19.blogspot.com/2018/06/classification-and-regression-tree-cart.html

https://repository.ipb.ac.id/handle/123456789/50394#:~:text=CART%20(Classifcation%20and%20Regression%20Tree,data%20yaitu%20teknik%20pohon%20keputusan.&text=CART%20mengbasilkan%20suatu%20pobon%20klasifikasi,regresi%20jika%20peubah%20responnya%20kontinu.

https://archive.ics.uci.edu/ml/datasets/car+evaluation

https://youtu.be/qS55tp5nuuo

http://ejurnal.its.ac.id/index.php/sains_seni/article/download/10673/2395#:~:text=Metode%20CART%20(Classification%20and%20Regression,variabel%20prediktor%20(variabel%20independen)