

# UTS Analisis Prediktif

Kelompok 1

2025-04-26

## Load Libraries

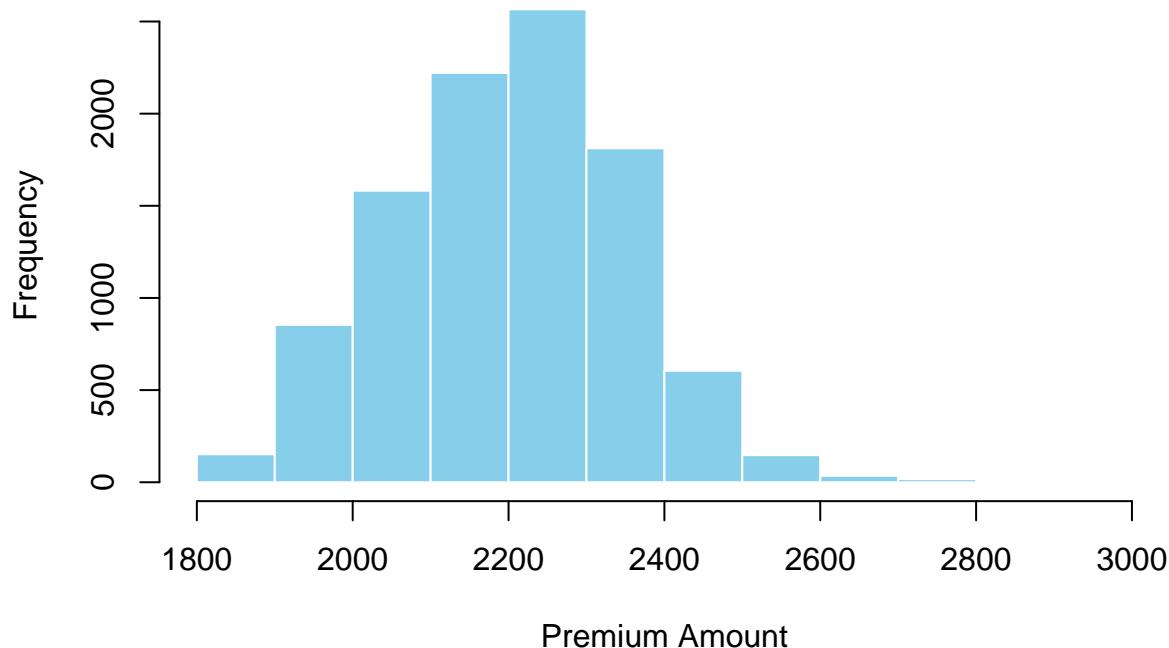
### Soal 1 - Analisis Data Eksplorasi

*Dataset* yang digunakan dalam laporan ini berisi karakteristik dan beberapa nilai historis dalam kepemilikan asuransi sebelumnya dari 10.000 pemegang polis terdahulu dari PT. Asuransi Pikagon Chandra. *Dataset* ini terdiri dari 27 variabel, dengan variabel *outputnya* adalah Premium\_Amount, 8 variabel numerik (Age, Claims\_Frequency, Claims\_Adjustment, Time\_Since\_First\_Contact, Website\_Visits, Inquiries, Time\_to\_Conversion, Credit\_Score), dan 18 variabel kategorik (Is\_Senior, Marital\_Status, Married\_Premium\_Discount, Prior\_Insurance, Prior\_Insurance\_Premium\_Adjustment, Claims\_Severity, Policy\_Type, Policy\_Adjustment, Safe\_Driver\_Discount, Multi\_Policy\_Discount, Bundling\_Discount, Total\_Discounts, Source\_of\_Lead, Conversion\_Status, Quotes\_Requested, Premium\_Adjustment\_Credit, Region, Premium\_Adjustment\_Region)

Pertama-tama, bentuk histogram dari variabel *output*, yaitu Premium\_Amount. Perhatikan gambar di bawah ini.

```
#histogram output
hist(data$Premium_Amount,
      main = "Histogram of Premium Amount",
      xlab = "Premium Amount",
      col = "skyblue",
      border = "white")
```

## Histogram of Premium Amount

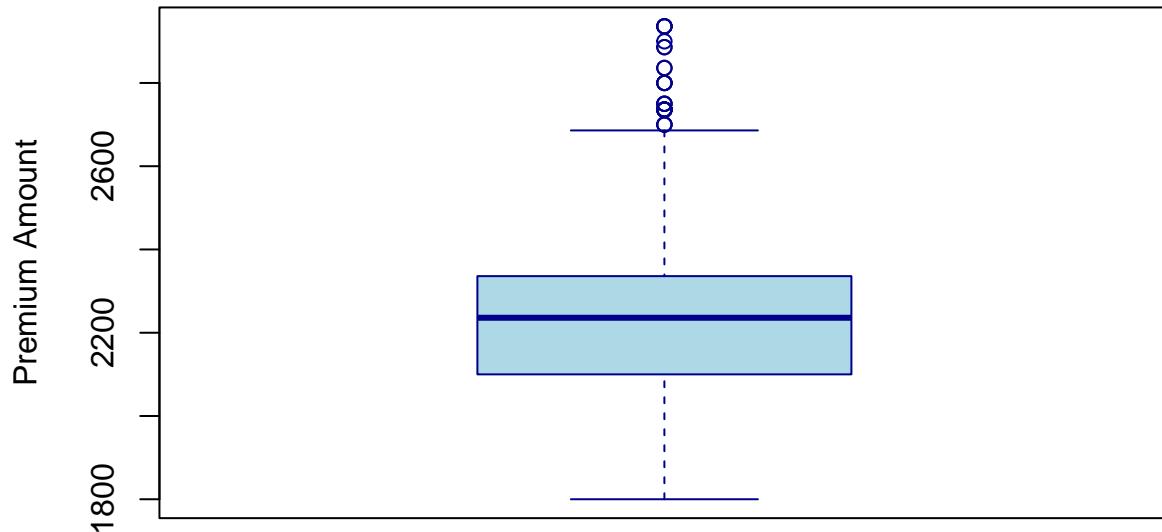


Histogram dari Premium\_Amount berbentuk simetris mendekati bentuk *bell shape*, tetapi agak menceng ke kanan. Sebagian besar data terkonsentrasi pada angka 2200 hingga 2300. Variabel Premium\_Amount memiliki bentuk yang cukup mendekati distribusi normal. Artinya, estimasi nilai dari variabel Premium\_Amount kemungkinan akan cocok dengan model regresi linear, yang memiliki asumsi normalitas (residualnya berdistribusi normal). Namun karena terdapat kemencengan, sebaiknya lakukan transformasi data agar model regresi dapat diperbaik. Opsi kedua dapat pula menggunakan model GLM (*Generalized Linear Model*) dengan distribusi Gamma.

Berikut merupakan gambar *boxplot* dari variabel Premium\_Amount.

```
#boxplot Premium_Amount
boxplot(data$Premium_Amount,
        main = "Boxplot of Premium Amount",
        ylab = "Premium Amount",
        col = "lightblue",
        border = "darkblue")
```

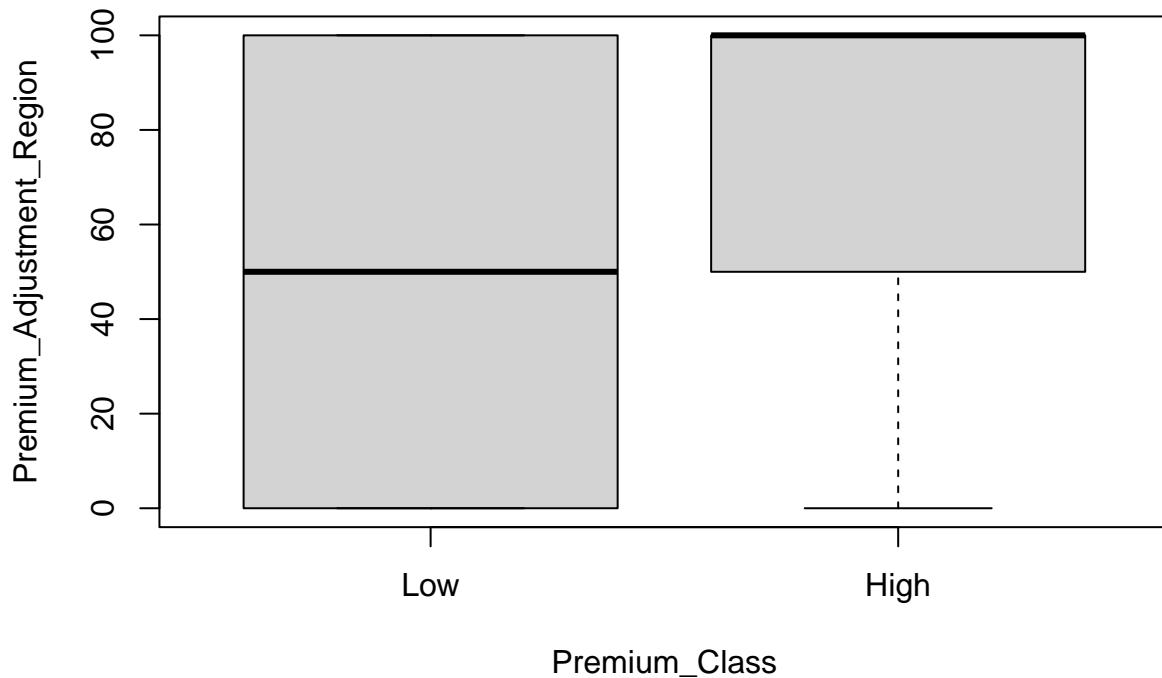
## Boxplot of Premium Amount



Kisaran harga premi berada di antara 1800 hingga 2936 dan sebagian besar harga premi berkisar di antara 2100 hingga 2336, dengan rata-rata harga premi adalah 2220 dan nilai tengahnya adalah 2236.

Selanjutnya, variabel Premium\_Amount akan diklasifikasi berdasarkan besar preminya. Jika besar premi kurang dari 2000 akan diklasifikasi sebagai "*low*" dan jika besar premi lebih besar dari 2000 akan diklasifikasi sebagai "*high*".

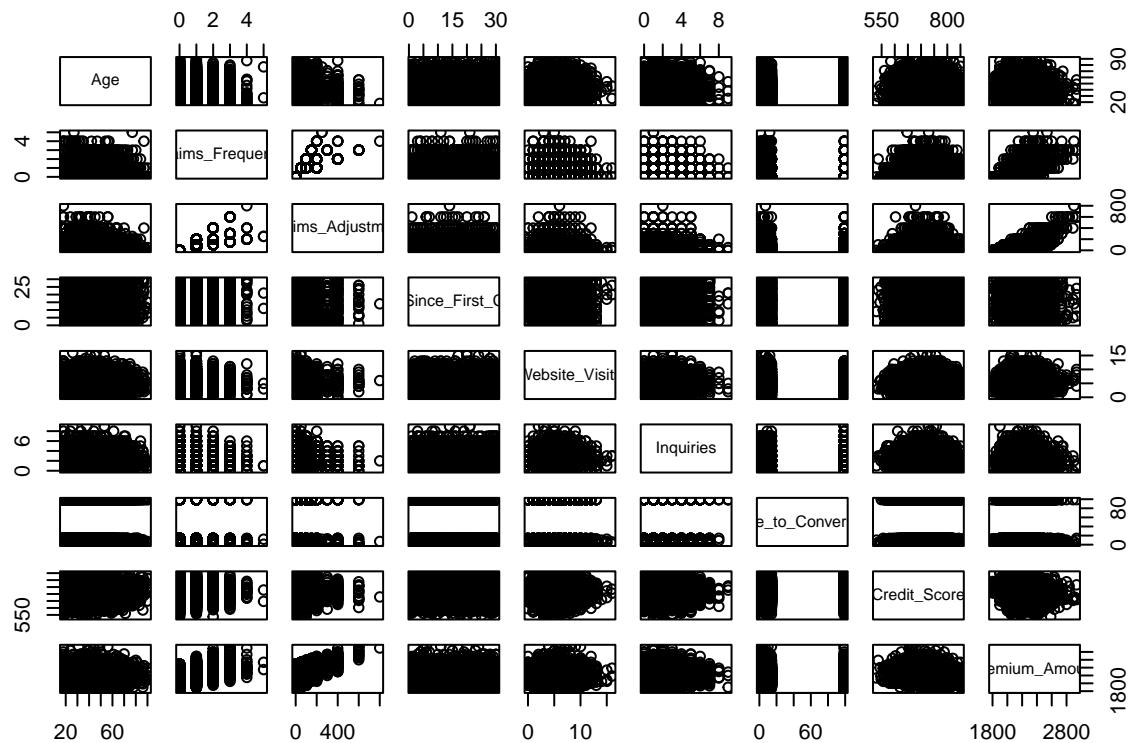
```
#Ubah Premium_Amount jadi kategorik
Premium_Class <- cut(Premium_Amount, breaks = c(0,2000,3000), labels =c("Low","High"))
boxplot(Premium_Adjustment_Region ~ Premium_Class)
```



Akan diperiksa hubungan antara variabel numerik dengan variabel numerik lainnya dengan menggunakan koefisien korelasi. Perhatikan *heatmap* pada gambar di bawah.

```
#periksa hubungan antar variabel numerik (koefisien korelasi)
data_numeric_manual <- data[, c("Age",
                                "Claims_Frequency",
                                "Claims_Adjustment",
                                "Time_Since_First_Contact",
                                "Website_Visits",
                                "Inquiries",
                                "Time_to_Conversion",
                                "Credit_Score",
                                "Premium_Amount")]

pairs(data_numeric_manual)
```



```
cor_matrix <- cor(data_numeric_manual, use = "complete.obs")
print(cor_matrix)
```

```
##                                     Age Claims_Frequency Claims_Adjustment
## Age                               1.000000000 -0.005682722 -0.007991319
## Claims_Frequency                  -0.005682722  1.000000000  0.803950488
## Claims_Adjustment                 -0.007991319  0.803950488  1.000000000
## Time_Since_First_Contact        0.012381982  0.001808578 -0.007773662
## Website_Visits                   -0.026850544  0.005437130  0.005136939
## Inquiries                         0.004740369  0.004282879  0.005831315
## Time_to_Conversion                -0.011567825  0.024032392  0.021283140
## Credit_Score                      0.002005075  0.002091579  0.005915483
## Premium_Amount                    -0.029540663  0.355371167  0.439129923
##                                     Time_Since_First_Contact Website_Visits   Inquiries
## Age                                0.012381982 -0.026850544  0.004740369
## Claims_Frequency                   0.001808578  0.005437130  0.004282879
## Claims_Adjustment                  -0.007773662  0.005136939  0.005831315
## Time_Since_First_Contact          1.000000000 -0.002829162  0.004622716
## Website_Visits                     -0.002829162  1.000000000 -0.002313379
## Inquiries                          0.004622716 -0.002313379  1.000000000
## Time_to_Conversion                 0.010437416 -0.024611929  0.007048466
## Credit_Score                       0.013877530 -0.006572704 -0.025561939
## Premium_Amount                     -0.001182656  0.024757612  0.002993340
##                                     Time_to_Conversion Credit_Score Premium_Amount
## Age                               -0.011567825  0.002005075 -0.029540663
```

```

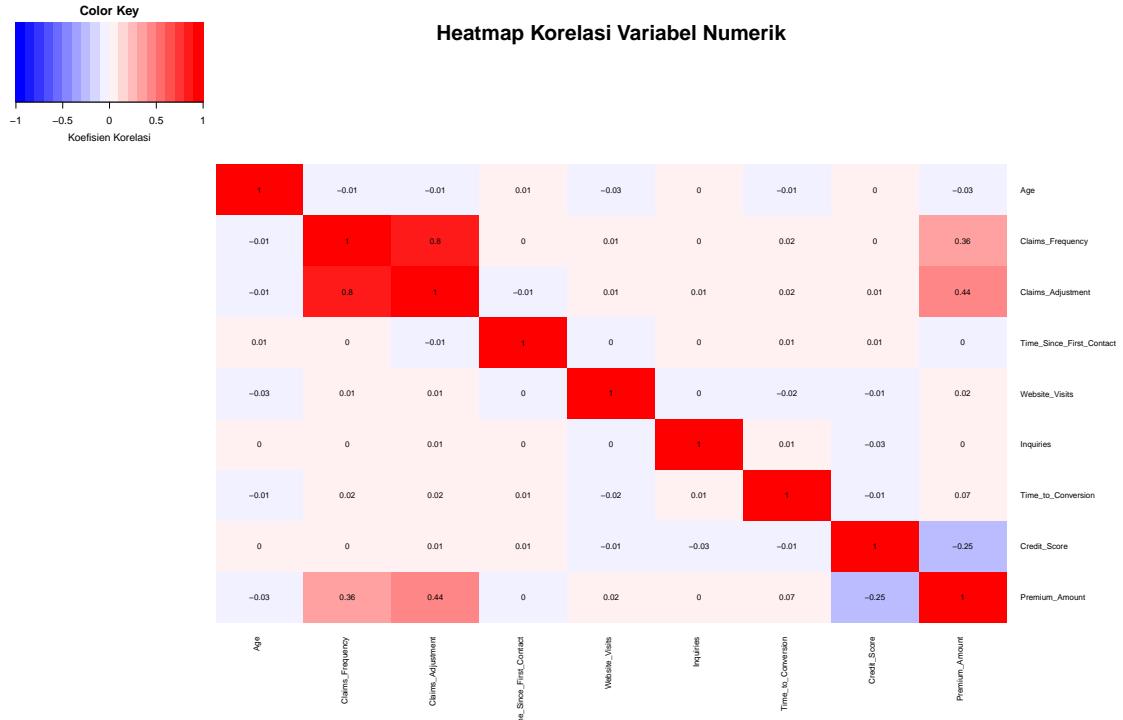
## Claims_Frequency          0.024032392  0.002091579  0.355371167
## Claims_Adjustment         0.021283140  0.005915483  0.439129923
## Time_Since_First_Contact 0.010437416  0.013877530 -0.001182656
## Website_Visits           -0.024611929 -0.006572704  0.024757612
## Inquiries                  0.007048466 -0.025561939  0.002993340
## Time_to_Conversion        1.000000000 -0.010305748  0.074710412
## Credit_Score               -0.010305748  1.000000000 -0.251238364
## Premium_Amount             0.074710412 -0.251238364  1.000000000

heatmap.2(cor_matrix,
  main = "Heatmap Korelasi Variabel Numerik",
  col = colorRampPalette(c("blue", "white", "red"))(20),
  symm = TRUE,
  trace = "none",
  density.info = "none",
  key.xlab = "Koefisien Korelasi",
  key.ylab = NULL,
  keyszie = 1,
  margins = c(15, 15), # Tingkatkan nilai margin
  cexRow = 0.7,
  cexCol = 0.7,
  Rowv = FALSE,
  Colv = FALSE,
  cellnote = round(cor_matrix, 2),
  notecl = "black",
  notecex = 0.7)

## Warning in heatmap.2(cor_matrix, main = "Heatmap Korelasi Variabel Numerik", :
## Discrepancy: Rowv is FALSE, while dendrogram is 'both'. Omitting row dendrogram.

## Warning in heatmap.2(cor_matrix, main = "Heatmap Korelasi Variabel Numerik", :
## Discrepancy: Colv is FALSE, while dendrogram is 'column'. Omitting column
## dendrogram.

```



Heatmap di atas menyatakan korelasi antar variabel numerik. Dari heatmap ini, terlihat bahwa variabel-variabel yang memiliki korelasi kuat (nilai lebih dari 0,5) adalah variabel Claims\_Frequency dan Claims\_Adjustment (dengan korelasi sebesar 0,8). Kedua variabel ini berhubungan secara positif. Semakin tinggi frekuensi klaim yang diajukan seseorang, maka semakin mahal premi yang harus dibayar orang tersebut. Artinya, kedua variabel ini saling berhubungan dan mempengaruhi nilai satu sama lain (secara positif jika nilainya lebih besar dari 0 dan secara negatif jika nilainya kurang dari 0). Dari heatmap ini, dapat disimpulkan bahwa sebagian variabel numerik saling tidak mempengaruhi satu sama lain, serta output (Premium\_Amount) tidak berkorelasi kuat oleh variabel numerik tertentu. Output hanya berkorelasi sedang dengan skor kredit (korelasi negatif), Claim\_Adjustment (korelasi positif), dan Claim\_Frequency (korelasi positif).

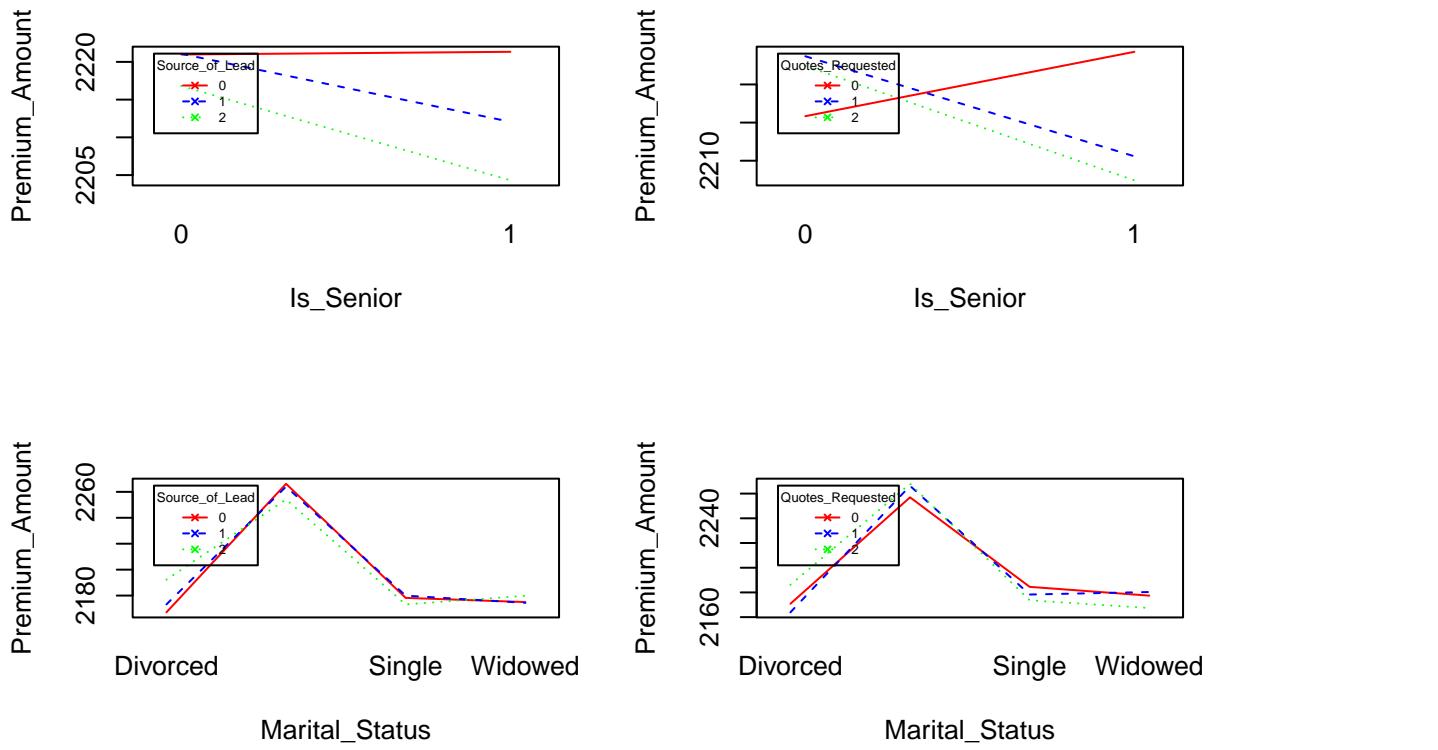
Selanjutnya, akan dianalisis hubungan antara variabel kategorik dengan variabel kategorik lainnya dengan menggunakan *interaction plot*. Berikut akan diberikan beberapa gambar *interaction plot* dari beberapa variabel kategorik, yaitu

- Is\_Senior dan Source\_of\_Lead
- Is\_Senior dan Quotes\_Requested
- Marital\_Status dan Source\_of\_Lead
- Marital\_Status dan Quotes\_Requested
- Married\_Premium\_Discount dan Source\_of\_Lead
- Married\_Premium\_Discount dan Quotes\_Requested
- Prior\_Insurance dan Source\_of\_Lead
- Prior\_Insurance dan Quotes\_Requested

- Prior\_Insurance\_Premium\_Adjustment dan Source\_of\_Lead
- Prior\_Insurance\_Premium\_Adjustment dan Quotes\_Requested
- Claims\_Severity dan Source\_of\_Lead
- Claims\_Severity dan Quotes\_Requested
- Policy\_Type dan Source\_of\_Lead
- Policy\_Adjustment dan Source\_of\_Lead
- Safe\_Driver\_Discount dan Source\_of\_Lead
- Multi\_Policy\_Discount dan Quotes\_Requested
- Bundling\_Discount dan Quotes\_Requested
- Total\_Discounts dan Source\_of\_Lead
- Total\_Discounts dan Quotes\_Requested
- Source\_of\_Lead dan Quotes\_Requested
- Conversion\_Status dan Quotes\_Requested

```
#Periksa hubungan antar variabel kategorik
par(mfrow=c(2,2))
```

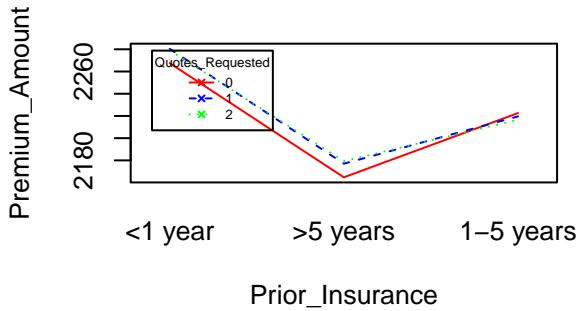
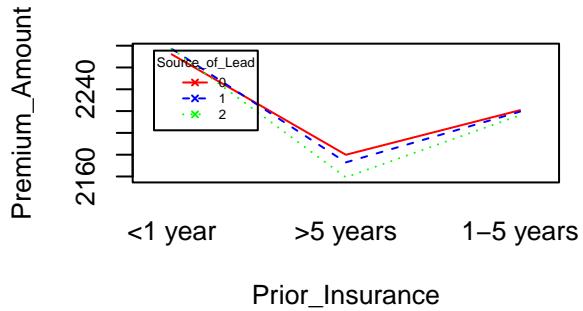
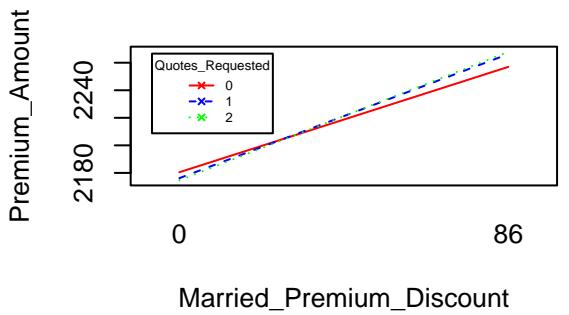
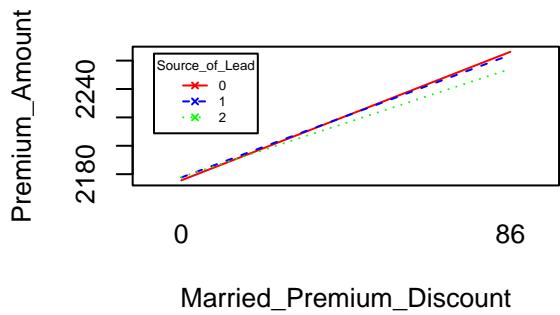
```
interaction.plot(Is_Senior, Source_of_Lead, Premium_Amount, xlab = "Is_Senior", ylab = "Premium_Amount"
legend("topleft",inset = .05, c("0","1","2"),col=c("red","blue","green","black"),lty=c(1,2,3,4),pch=4, -)
interaction.plot(Is_Senior, Quotes_Requested, Premium_Amount, xlab = "Is_Senior", ylab = "Premium_Amount"
legend("topleft",inset = .05, c("0","1","2"),col=c("red","blue","green","black"),lty=c(1,2,3,4),pch=4, -)
interaction.plot(Marital_Status, Source_of_Lead, Premium_Amount, xlab = "Marital_Status", ylab = "Premium_Amount"
legend("topleft",inset = .05, c("0","1","2"),col=c("red","blue","green","black"),lty=c(1,2,3,4),pch=4, -)
interaction.plot(Marital_Status, Quotes_Requested, Premium_Amount, xlab = "Marital_Status", ylab = "Premium_Amount"
legend("topleft",inset = .05, c("0","1","2"),col=c("red","blue","green","black"),lty=c(1,2,3,4),pch=4, -)
```



```

interaction.plot(Married_Premium_Discount, Source_of_Lead, Premium_Amount, xlab = "Married_Premium_Discount", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")
interaction.plot(Married_Premium_Discount, Quotes_Requested, Premium_Amount, xlab = "Married_Premium_Discount", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")
interaction.plot(Prior_Insurance, Source_of_Lead, Premium_Amount, xlab = "Prior_Insurance", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")
interaction.plot(Prior_Insurance, Quotes_Requested, Premium_Amount, xlab = "Prior_Insurance", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")

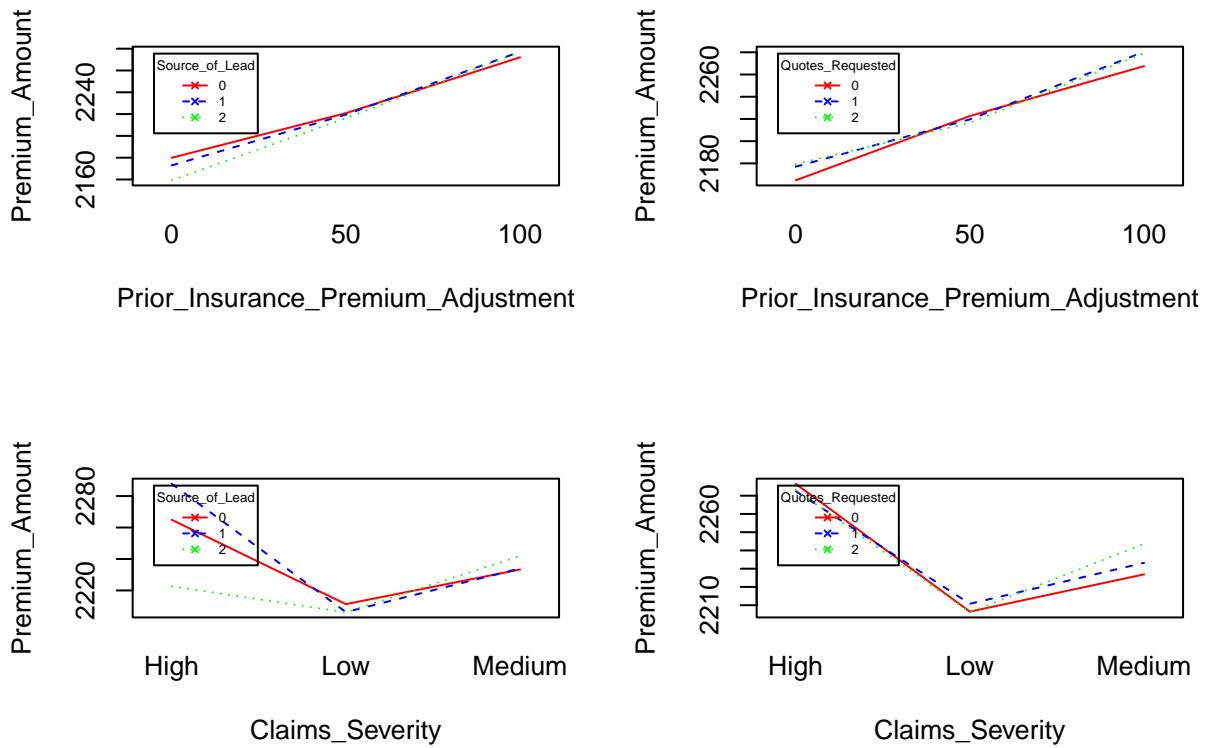
```



```

interaction.plot(Prior_Insurance_Premium_Adjustment, Source_of_Lead, Premium_Amount, xlab = "Prior_Insurance_Premium_Adjustment", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")
interaction.plot(Prior_Insurance_Premium_Adjustment, Quotes_Requested, Premium_Amount, xlab = "Prior_Insurance_Premium_Adjustment", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")
interaction.plot(Claims_Severity, Source_of_Lead, Premium_Amount, xlab = "Claims_Severity", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")
interaction.plot(Claims_Severity, Quotes_Requested, Premium_Amount, xlab = "Claims_Severity", ylab = "Premium_Amount", legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4, type="l")

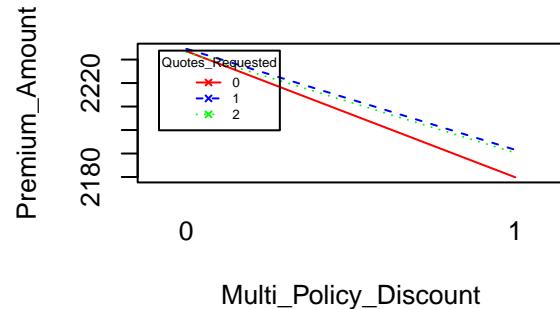
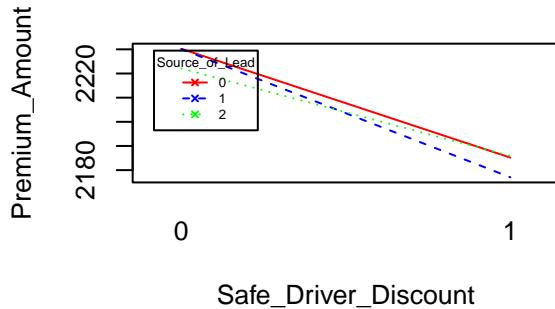
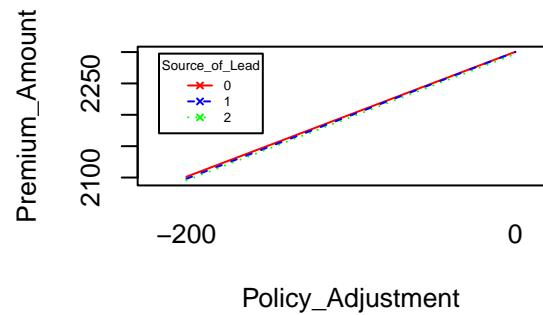
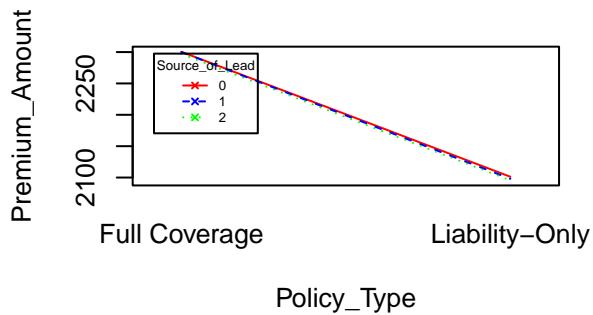
```



```

interaction.plot(Policy_Type, Source_of_Lead, Premium_Amount, xlab = "Policy_Type", ylab = "Premium_Amount",
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, -)
interaction.plot(Policy_Adjustment, Source_of_Lead, Premium_Amount, xlab = "Policy_Adjustment", ylab = "Premium_Amount",
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, -)
interaction.plot(Safe_Driver_Discount, Source_of_Lead, Premium_Amount, xlab = "Safe_Driver_Discount", ylab = "Premium_Amount",
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, -)
interaction.plot(Multi_Policy_Discount, Quotes_Requested, Premium_Amount, xlab = "Multi_Policy_Discount", ylab = "Premium_Amount",
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, -)

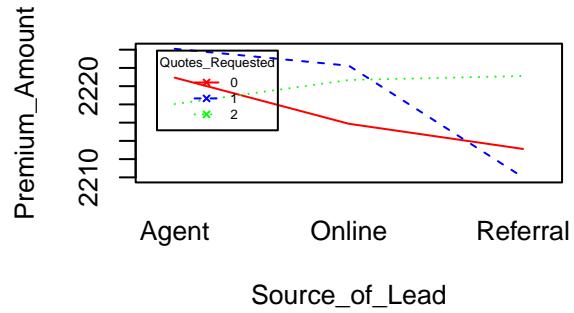
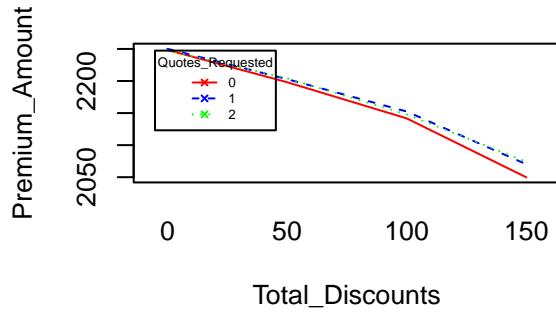
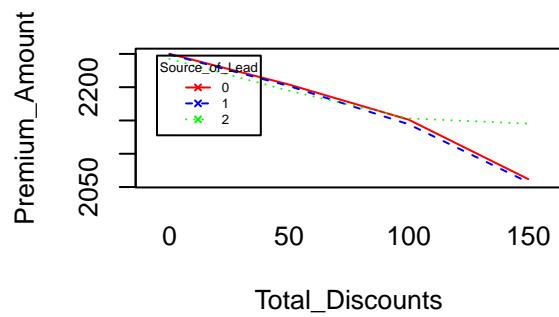
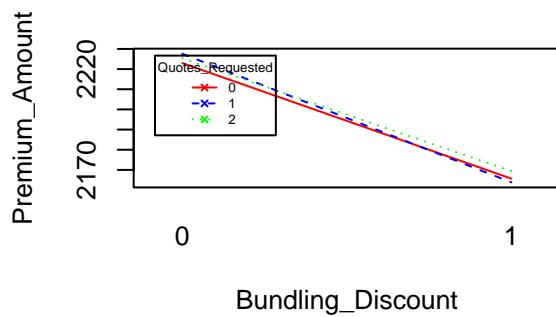
```



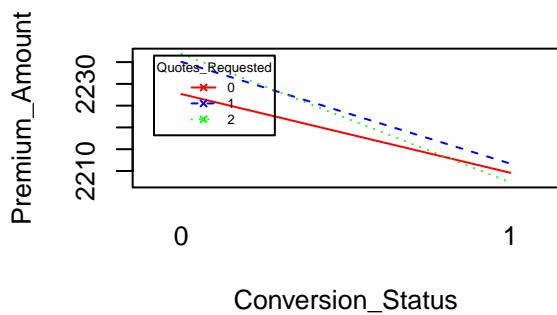
```

interaction.plot(Bundling_Discount, Quotes_Requested, Premium_Amount, xlab = "Bundling_Discount", ylab =
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, 
interaction.plot(Total_Discounts, Source_of_Lead, Premium_Amount, xlab = "Total_Discounts", ylab = "Premium_Amount"
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, 
interaction.plot(Total_Discounts, Quotes_Requested, Premium_Amount, xlab = "Total_Discounts", ylab = "Premium_Amount"
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, 
interaction.plot(Source_of_Lead, Quotes_Requested, Premium_Amount, xlab = "Source_of_Lead", ylab = "Premium_Amount"
legend("topleft", inset = .05, c("0","1","2"), col=c("red","blue","green","black"), lty=c(1,2,3,4), pch=4, 

```



```
interaction.plot(Conversion_Status, Quotes_Requested, Premium_Amount, xlab = "Conversion_Status", ylab =
legend("topleft", inset = .05, c("0", "1", "2"), col=c("red", "blue", "green", "black"), lty=c(1,2,3,4), pch=4,
```



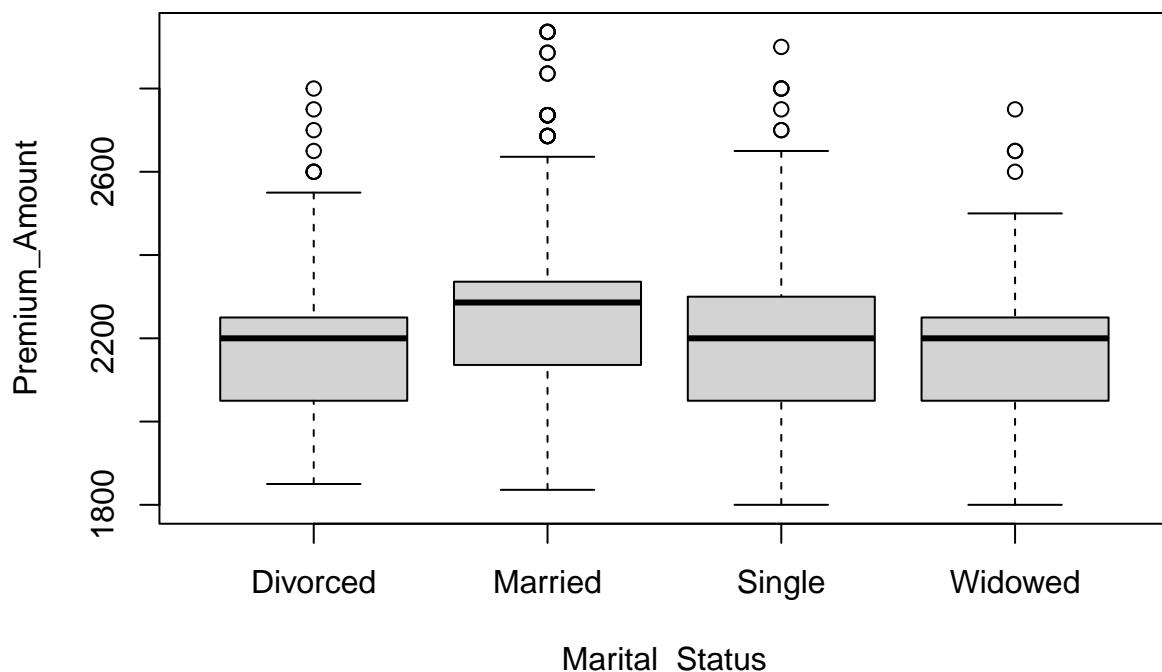
Dari gambar di atas, berikut beberapa variabel kategorik yang saling berhubungan

- Is\_Senior dan Source\_of\_Lead
- Is\_Senior dan Quotes\_Requested
- Marital\_Status dan Source\_of\_Lead
- Marital\_Status dan Quotes\_Requested
- Married\_Premium\_Discount dan Source\_of\_Lead
- Married\_Premium\_Discount dan Quotes\_Requested
- Prior\_Insurance dan Source\_of\_Lead
- Prior\_Insurance dan Quotes\_Requested
- Prior\_Insurance\_Premium\_Adjustment dan Source\_of\_Lead
- Prior\_Insurance\_Premium\_Adjustment dan Quotes\_Requested
- Claims\_Severity dan Source\_of\_Lead
- Claims\_Severity dan Quotes\_Requested
- Policy\_Type dan Source\_of\_Lead
- Policy\_Adjustment dan Source\_of\_Lead
- Safe\_Driver\_Discount dan Source\_of\_Lead

- Multi\_Policy\_Discount dan Quotes\_Requested
- Bundling\_Discount dan Quotes\_Requested
- Total\_Discounts dan Source\_of\_Lead
- Total\_Discounts dan Quotes\_Requested
- Source\_of\_Lead dan Quotes\_Requested
- Conversion\_Status dan Quotes\_Requested

Selanjutnya, akan dianalisis hubungan antara suatu variabel kategorik dengan *output* Premium\_Amount dengan menggunakan metode ANOVA.

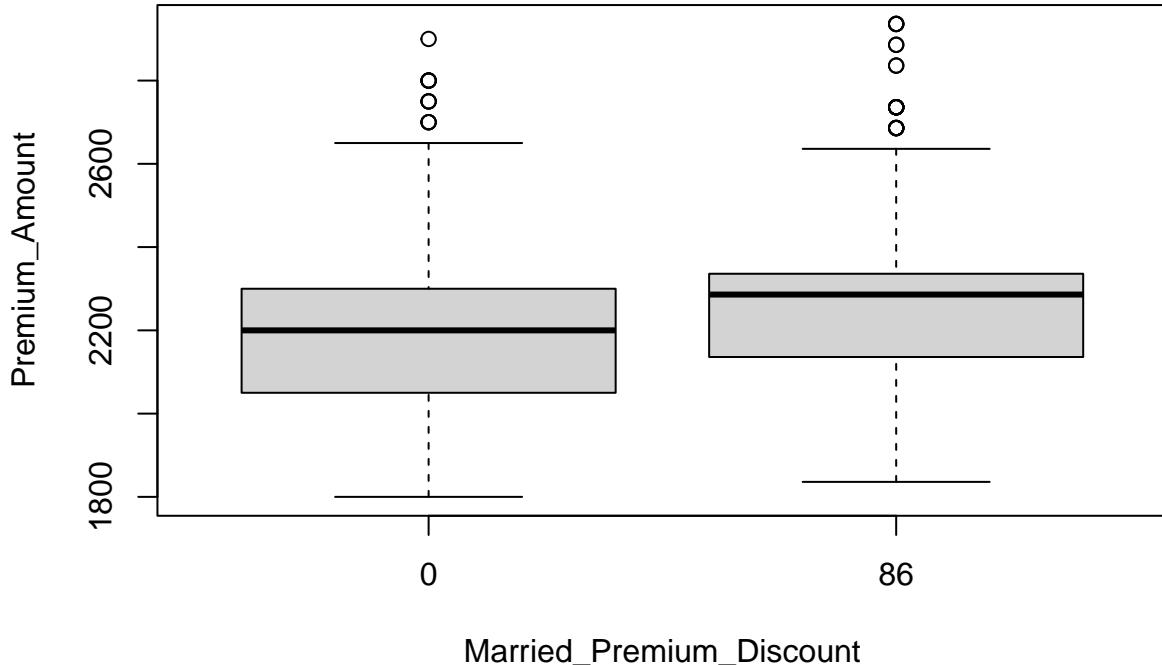
```
#periksa hubungan output dgn variabel kategorik (anova)
boxplot(Premium_Amount ~ Marital_Status)
```



```
anova2 <- aov(Premium_Amount ~ Marital_Status)
summary(anova2)
```

```
##          Df   Sum Sq Mean Sq F value Pr(>F)
## Marital_Status     3 18779068 6259689   310.1 <2e-16 ***
## Residuals      9996 201784139    20186
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

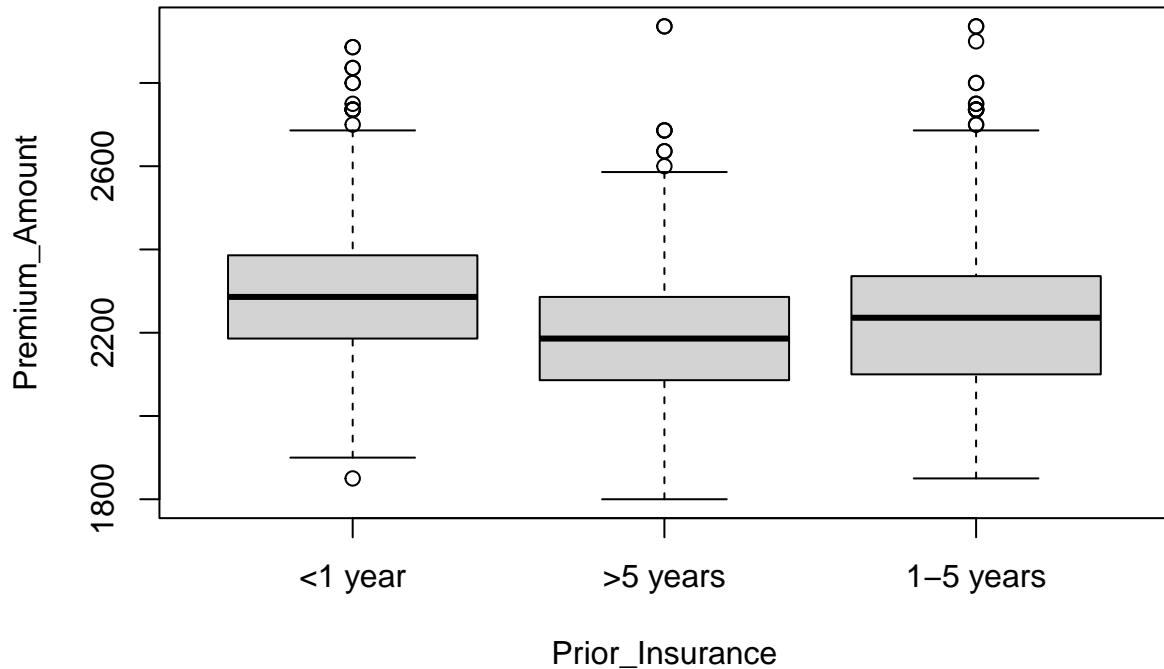
```
boxplot(Premium_Amount ~ Married_Premium_Discount)
```



```
anova3 <- aov(Premium_Amount ~ Married_Premium_Discount)
summary(anova3)
```

```
##                               Df Sum Sq Mean Sq F value Pr(>F)
## Married_Premium_Discount     1 18753731 18753731    929.1 <2e-16 ***
## Residuals                     9998 201809476      20185
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

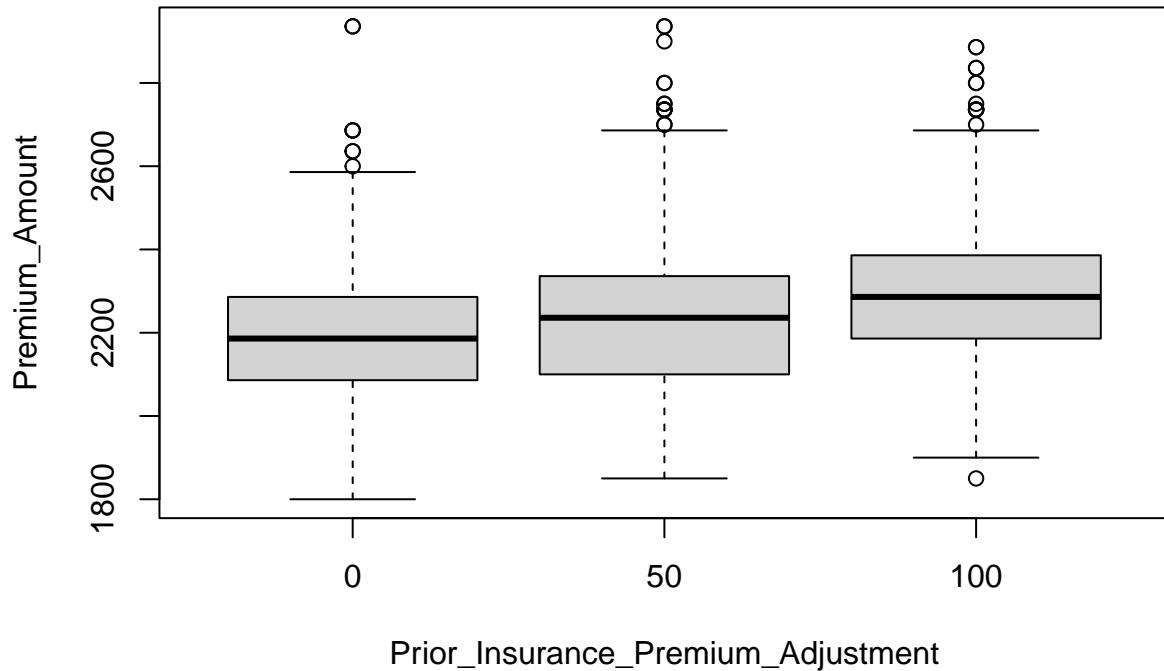
```
boxplot(Premium_Amount ~ Prior_Insurance)
```



```
anova4 <- aov(Premium_Amount ~ Prior_Insurance)
summary(anova4)
```

```
##          Df Sum Sq Mean Sq F value Pr(>F)
## Prior_Insurance  2 12196237 6098118   292.6 <2e-16 ***
## Residuals      9997 208366970    20843
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

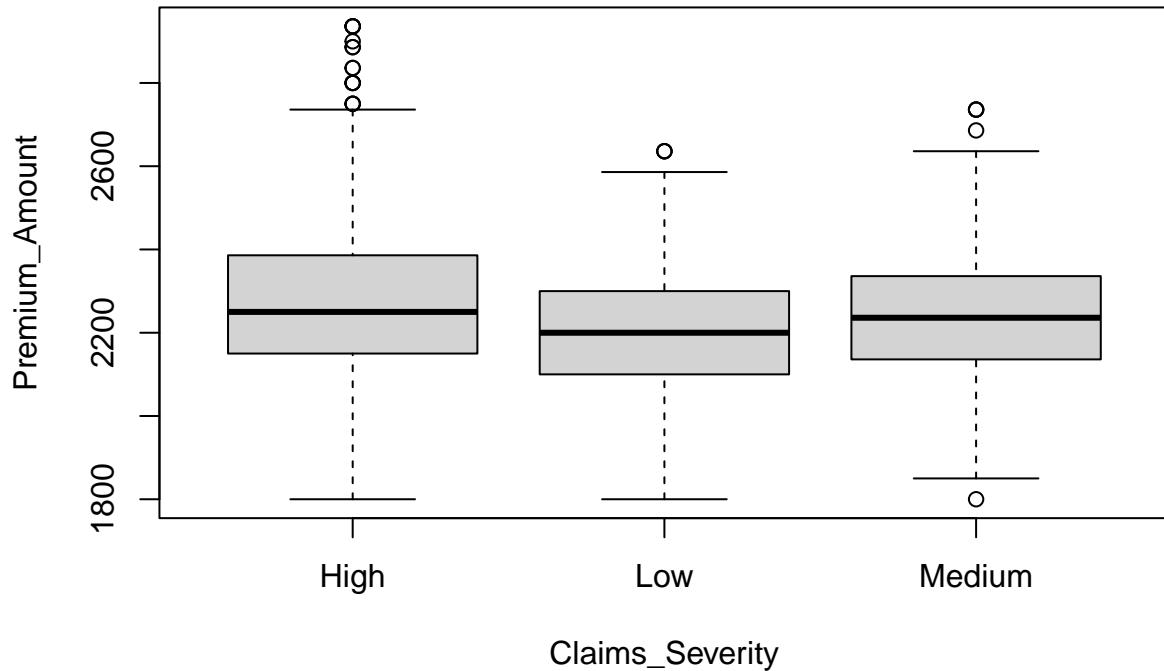
```
boxplot(Premium_Amount ~ Prior_Insurance_Premium_Adjustment)
```



```
anova5 <- aov(Premium_Amount ~ Prior_Insurance_Premium_Adjustment)
summary(anova5)
```

```
##                                     Df Sum Sq Mean Sq F value Pr(>F)
## Prior_Insurance_Premium_Adjustment  1 12133050 12133050      582 <2e-16 ***
## Residuals                          9998 208430157    20847
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

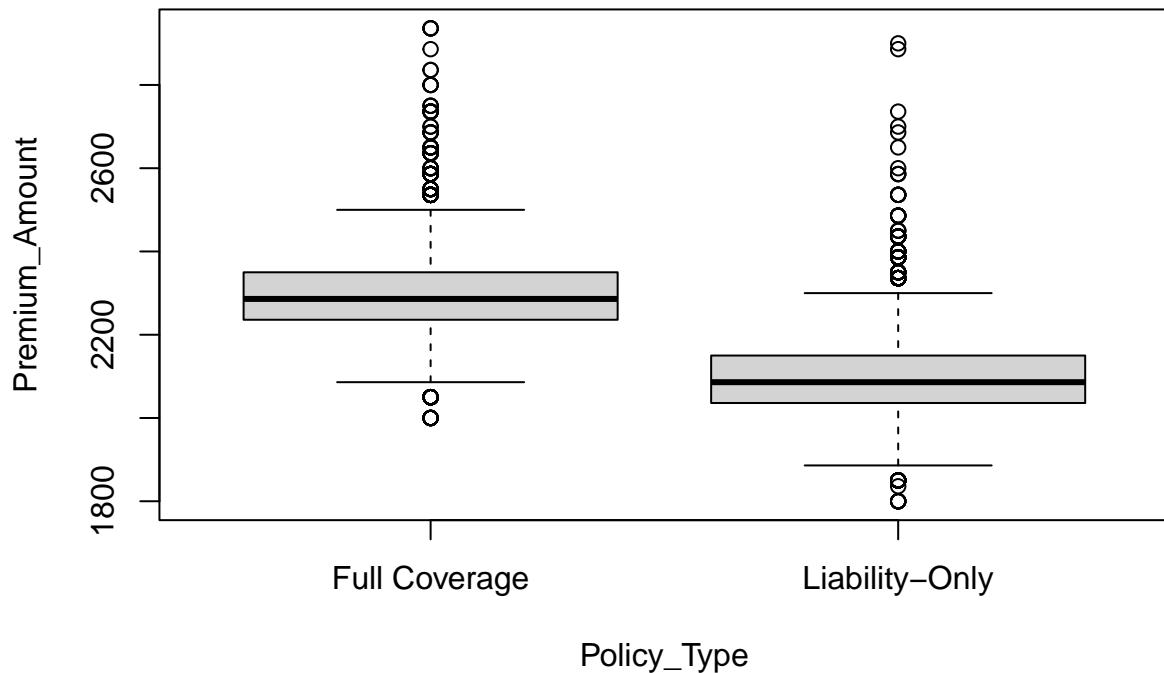
```
boxplot(Premium_Amount ~ Claims_Severity)
```



```
anova6 <- aov(Premium_Amount ~ Claims_Severity)
summary(anova6)
```

```
##          Df   Sum Sq Mean Sq F value Pr(>F)
## Claims_Severity    2 4276847 2138424  98.84 <2e-16 ***
## Residuals      9997 216286360    21635
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

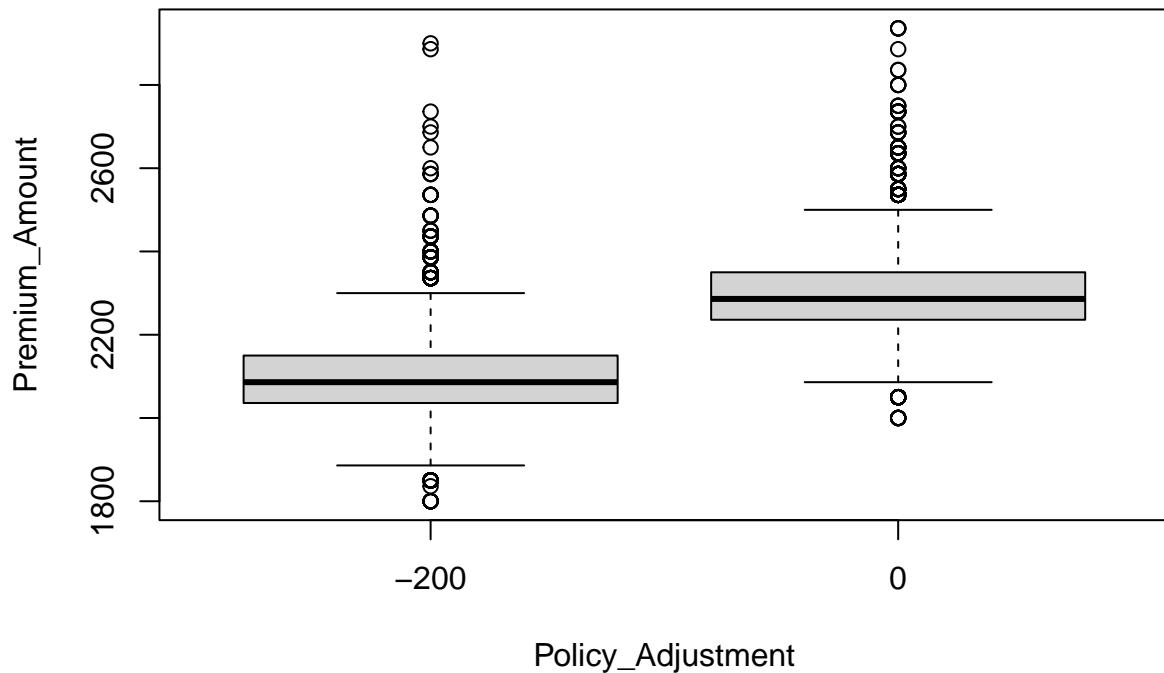
```
boxplot(Premium_Amount ~ Policy_Type)
```



```
anova7 <- aov(Premium_Amount ~ Policy_Type)
summary(anova7)
```

```
##              Df    Sum Sq  Mean Sq F value Pr(>F)
## Policy_Type     1 97062076 97062076    7858 <2e-16 ***
## Residuals   9998 123501131     12353
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

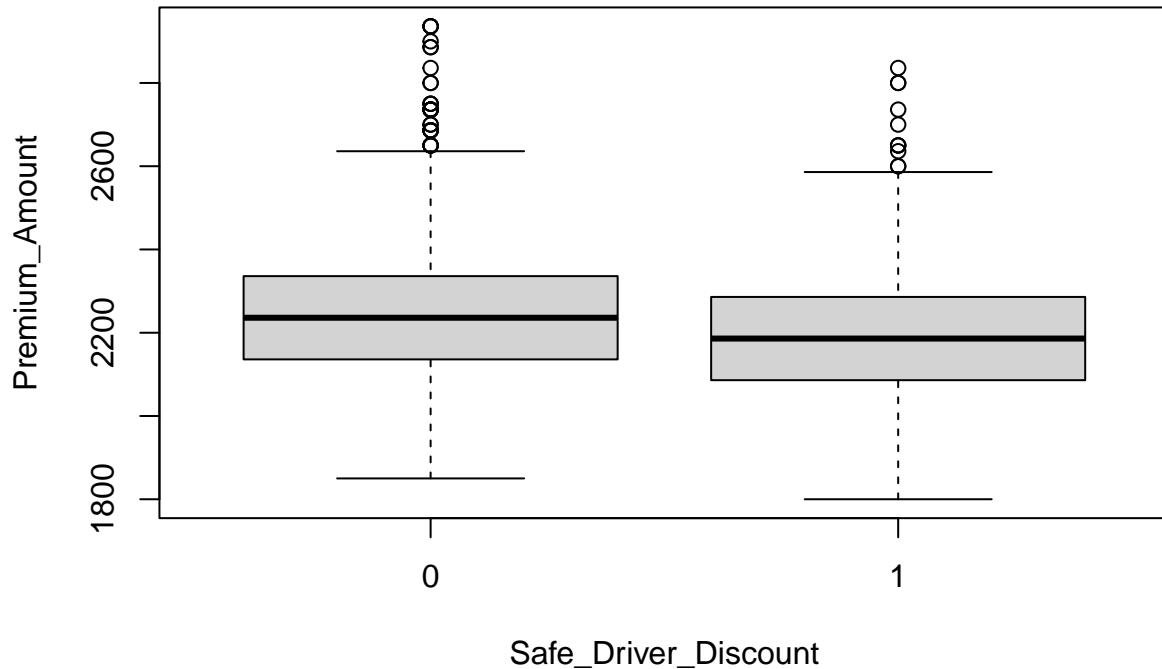
```
boxplot(Premium_Amount ~ Policy_Adjustment)
```



```
anova8 <- aov(Premium_Amount ~ Policy_Adjustment)
summary(anova8)
```

```
##                                Df    Sum Sq  Mean Sq F value Pr(>F)
## Policy_Adjustment      1 97062076 97062076     7858 <2e-16 ***
## Residuals             9998 123501131     12353
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

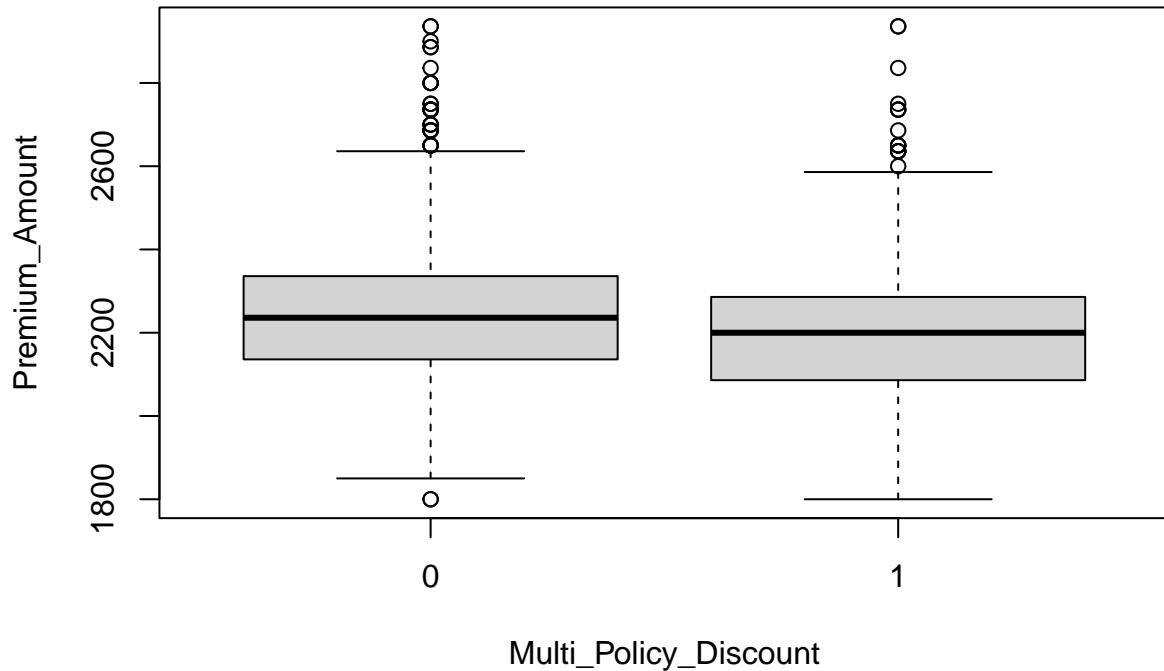
```
boxplot(Premium_Amount ~ Safe_Driver_Discount)
```



```
anova9 <- aov(Premium_Amount ~ Safe_Driver_Discount)
summary(anova9)
```

```
##                                Df    Sum Sq Mean Sq F value Pr(>F)
## Safe_Driver_Discount      1 3839829 3839829   177.1 <2e-16 ***
## Residuals                  9998 216723378   21677
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

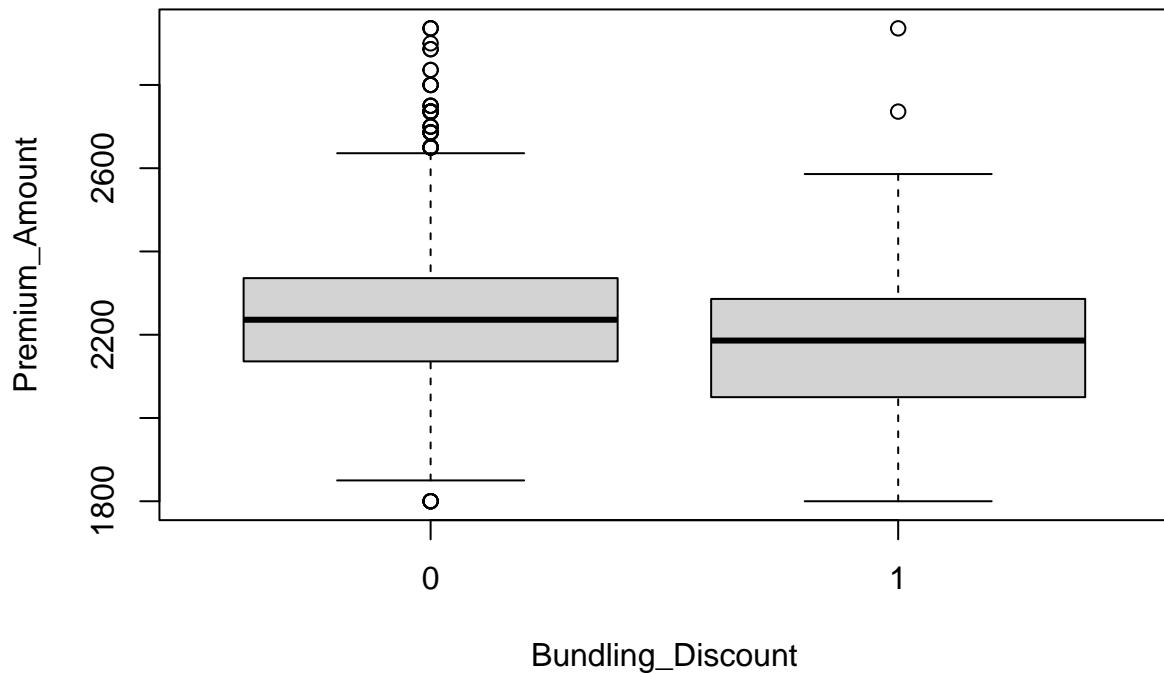
```
boxplot(Premium_Amount ~ Multi_Policy_Discount)
```



```
anova10 <- aov(Premium_Amount ~ Multi_Policy_Discount)
summary(anova10)
```

```
##                                Df Sum Sq Mean Sq F value Pr(>F)
## Multi_Policy_Discount     1 4597473 4597473   212.8 <2e-16 ***
## Residuals                  9998 215965734    21601
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

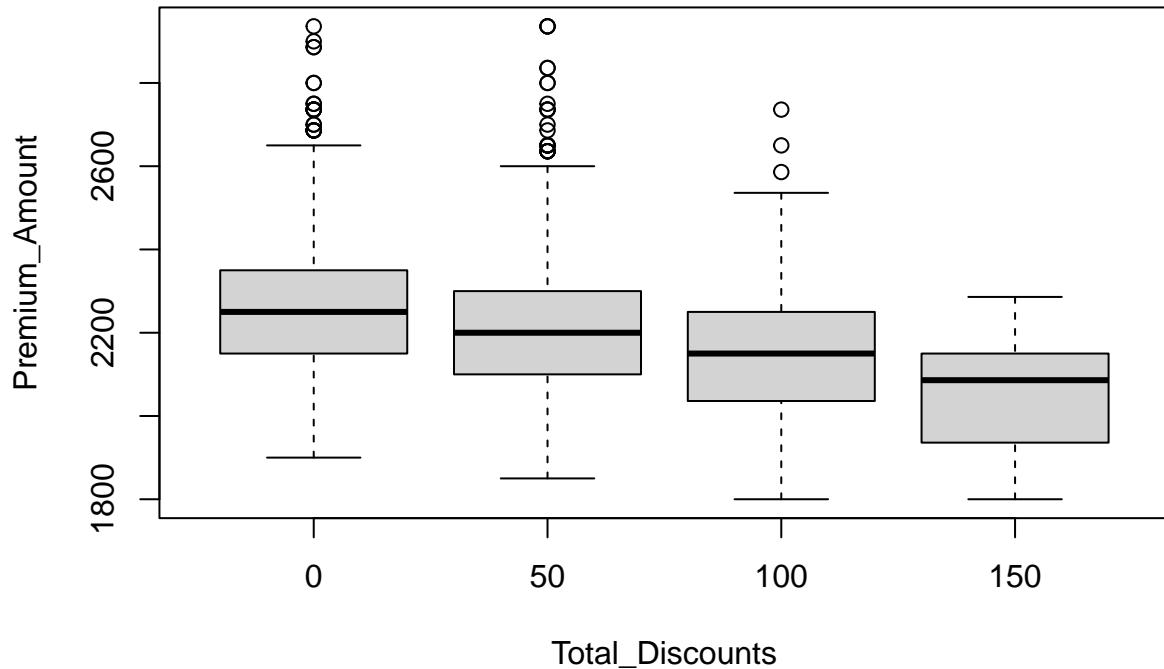
```
boxplot(Premium_Amount ~ Bundling_Discount)
```



```
anova11 <- aov(Premium_Amount ~ Bundling_Discount)
summary(anova11)
```

```
##                                Df    Sum Sq Mean Sq F value Pr(>F)
## Bundling_Discount      1 3051070 3051070   140.2 <2e-16 ***
## Residuals              9998 217512137   21756
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

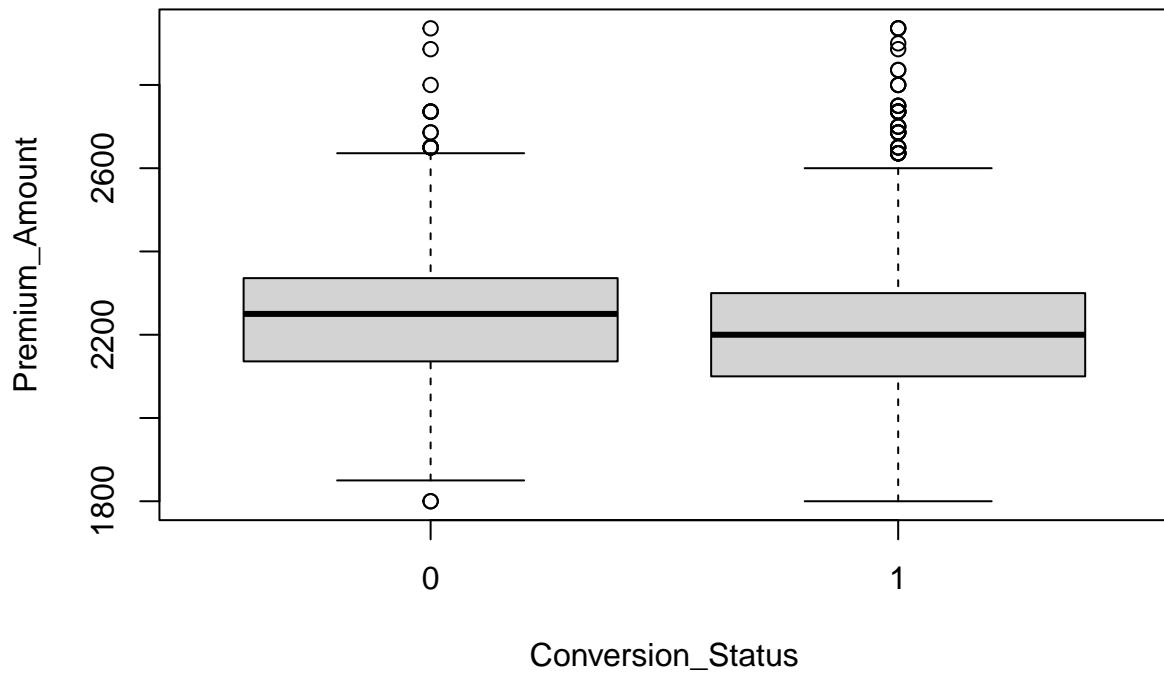
```
boxplot(Premium_Amount ~ Total_Discounts)
```



```
anova12 <- aov(Premium_Amount ~ Total_Discounts)
summary(anova12)
```

```
##              Df Sum Sq Mean Sq F value Pr(>F)
## Total_Discounts     1 11535781 11535781    551.8 <2e-16 ***
## Residuals         9998 209027426      20907
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

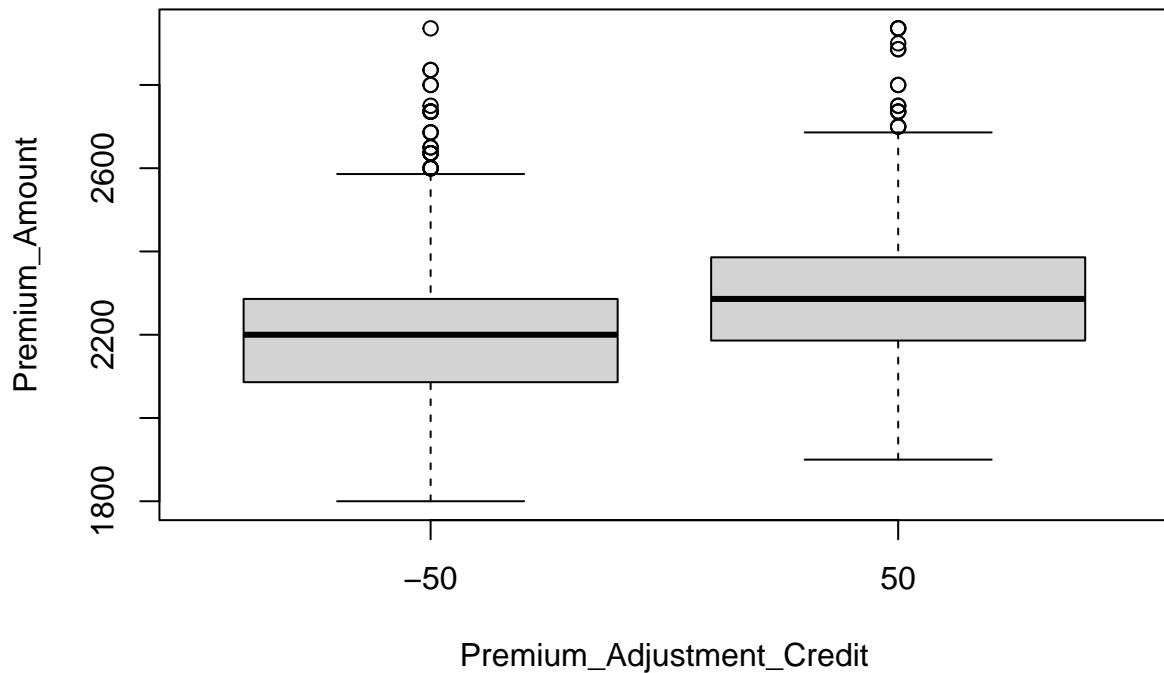
```
boxplot(Premium_Amount ~ Conversion_Status)
```



```
anova14 <- aov(Premium_Amount ~ Conversion_Status)
summary(anova14)
```

```
##                                Df    Sum Sq Mean Sq F value    Pr(>F)
## Conversion_Status      1 1368363 1368363   62.41 3.08e-15 ***
## Residuals            9998 219194844    21924
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

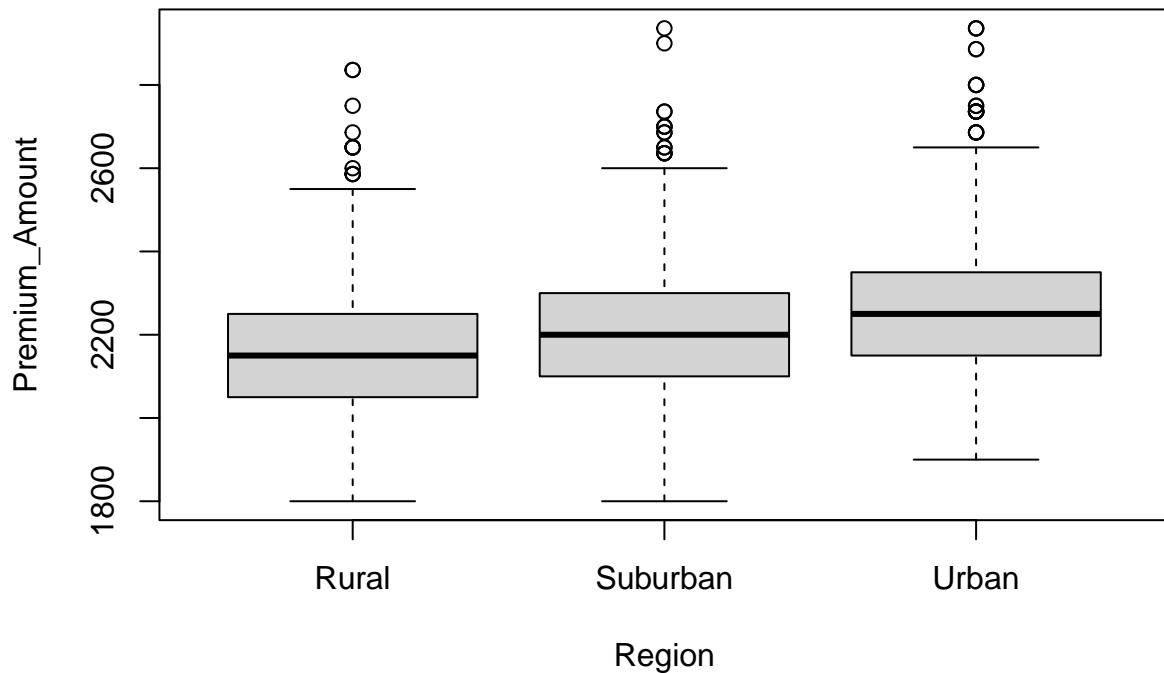
```
boxplot(Premium_Amount ~ Premium_Adjustment_Credit)
```



```
anova16 <- aov(Premium_Amount ~ Premium_Adjustment_Credit)
summary(anova16)
```

```
##                               Df   Sum Sq Mean Sq F value Pr(>F)
## Premium_Adjustment_Credit    1 23418345 23418345    1188 <2e-16 ***
## Residuals                  9998 197144862    19718
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

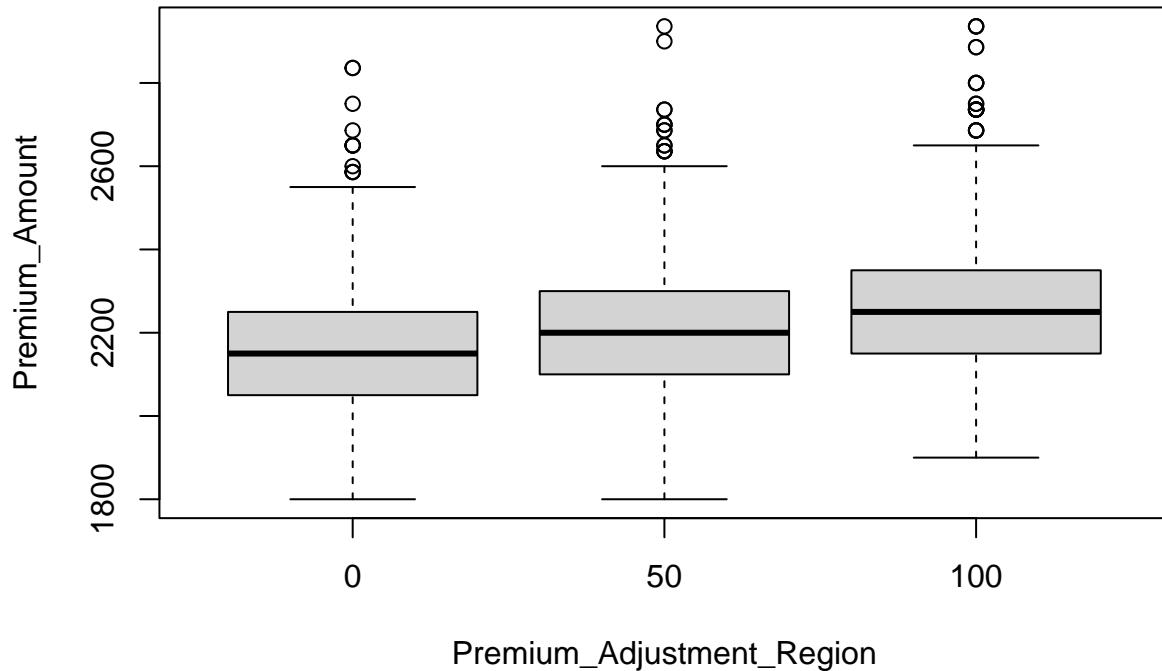
```
boxplot(Premium_Amount ~ Region)
```



```
anova17 <- aov(Premium_Amount ~ Region)
summary(anova17)
```

```
##           Df Sum Sq Mean Sq F value Pr(>F)
## Region      2 15632513 7816257   381.3 <2e-16 ***
## Residuals  9997 204930694    20499
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
boxplot(Premium_Amount ~ Premium_Adjustment_Region)
```



```
anova18 <- aov(Premium_Amount ~ Premium_Adjustment_Region)
summary(anova18)
```

```
##                               Df   Sum Sq Mean Sq F value Pr(>F)
## Premium_Adjustment_Region    1 15582143 15582143     760 <2e-16 ***
## Residuals                  9998 204981064     20502
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Dari sini, diperoleh beberapa variabel yang saling berhubungan adalah

- Premium\_Amount dan Marital\_Status

Pelanggan yang sudah menikah mendapat rata-rata harga premi yang paling tinggi, sementara untuk pelanggan yang sudah bercerai, masih single, atau janda memiliki rata-rata harga premi yang lebih rendah, dengan harga yang kurang lebih sama. Artinya, pelanggan yang hidup dengan pasangan umumnya harus membayar premi yang lebih mahal.

- Premium\_Amount dan Married\_Premium\_Discount

Variabel Married\_Premium\_Discount akan memberikan pelanggan yang sudah menikah tambahan sebesar 86 pada premi. Penambahan premi tentunya akan total premi menjadi lebih banyak.

- Premium\_Amount dan Prior\_Insurance

Pelanggan yang sebelumnya memiliki asuransi selama kurang dari 1 tahun akan membayar premi lebih mahal daripada yang sebelumnya memiliki asuransi selama 1-5 tahun atau lebih dari 5 tahun.

- Premium\_Amount dan Prior\_Insurance\_Premium\_Adjustment

Variabel Prior\_Insurance\_Premium\_Adjustment akan memberikan tambahan premi sebesar 50 jika pelanggan memiliki asuransi sebelumnya selama 1-5 tahun dan memberi tambahan premi sebesar 100 jika pelanggan memiliki asuransi sebelumnya selama kurang dari 1 tahun. Penambahan premi sebesar 50 tentunya akan menambah total premi dan penambahan premi sebesar 100 akan menambah total premi lebih banyak (seperti yang ditunjukkan dalam *boxplot*).

- Premium\_Amount dan Claims\_Severity

Pelanggan yang memiliki *severity* tinggi perlu membayar premi dengan harga yang lebih mahal daripada pelanggan yang memiliki *severity* sedang atau rendah. Pelanggan yang memiliki *severity* rendah akan diberikan harga premi yang paling murah.

- Premium\_Amount dan Policy\_Type

Pelanggan dengan tipe polis *Liability-Only* akan diberikan pengurangan premi pada variabel “Policy\_Adjustment”, sehingga harga preminya lebih murah.

- Premium\_Amount dan Policy\_Adjustment

Variabel Policy\_Adjustment akan memberikan pengurangan premi sebesar 200 jika tipe polisnya adalah *Liability-Only*. Pengurangan premi sebesar 200 tentunya akan mengurangi total premi (seperti yang ditunjukkan dalam *boxplot*).

- Premium\_Amount dan Safe\_Driver\_Discount}

Pelanggan yang mendapat diskon karena memiliki riwayat mengemudi aman cenderung membayar premi yang lebih murah daripada pelanggan yang tidak memiliki riwayat mengemudi aman.

- Premium\_Amount dan Multi\_Policy\_Discount

Pelanggan yang mendapat diskon karena memiliki lebih dari satu polis cenderung membayar premi yang lebih murah daripada pelanggan yang hanya memiliki satu polis.

- Premium\_Amount dan Bundling\_Discount

Pelanggan yang mendapat diskon karena *bundling* produk cenderung membayar premi yang lebih murah daripada pelanggan yang tidak mendapat diskon bundling produk.

- Premium\_Amount dan Total\_Discounts

Semakin tinggi total diskon, semakin murah pula premi yang harus dibayar pelanggan.

- Premium\_Amount dan Conversion\_Status

Pelanggan yang berhasil dikonversi cenderung harus membayar premi yang lebih murah daripada pelanggan yang tidak berhasil dikonversi.

- Premium\_Amount dan Premium\_Adjustment\_Credit

Variabel Premium\_Adjustment\_Credit akan memberikan tambahan premi sebesar 50 jika skor kredit bernilai kurang dari 700 dan memberi pengurangan premi sebesar 50 jika skor kredit bernilai lebih dari 700. Penambahan premi sebesar 50 tentunya akan menambah total premi dan pengurangan premi sebesar 50 akan mengurangi total premi (seperti yang ditunjukkan dalam *boxplot*).

- Premium\_Amount dan Region

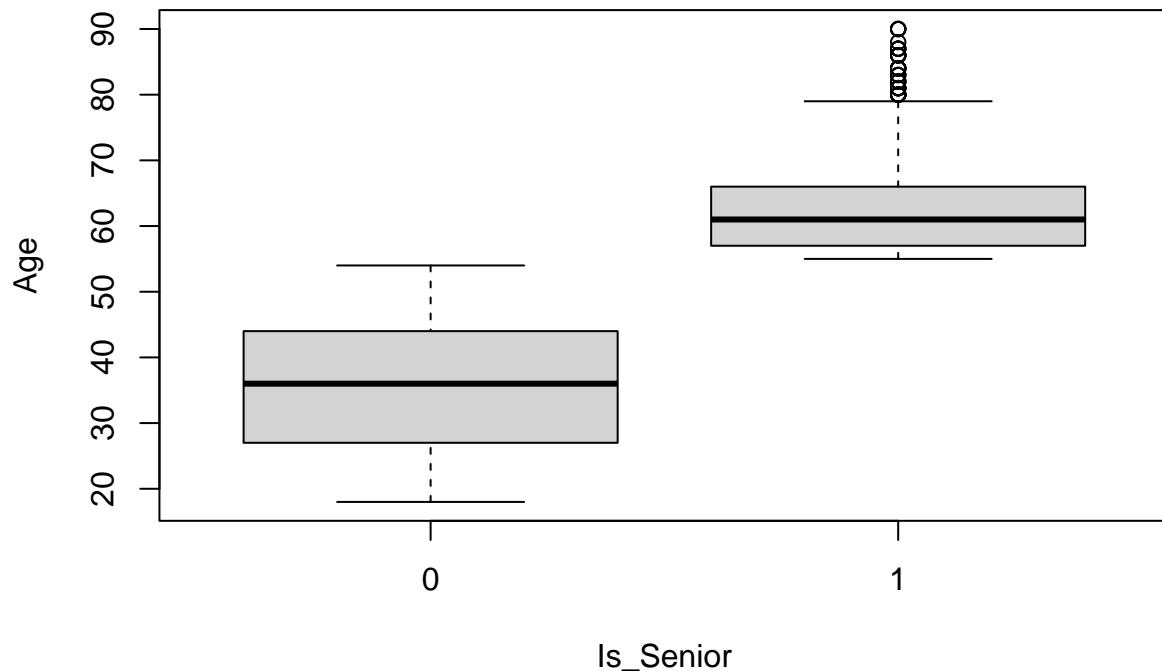
Rata-rata harga premi paling rendah hingga paling tinggi secara berturut-turut adalah daerah pedesaan, pinggiran kota, dan perkotaan.

- Premium\_Amount dan Premium\_Adjustment\_Region

Dengan melihat *boxplot* yang terbentuk, terlihat bahwa semakin besar penyesuaian premi akibat wilayah, semakin besar pula rata-rata harga premi. Artinya, wilayah dengan penyesuaian premi terbesar cenderung memberikan harga premi yang lebih mahal daripada wilayah lainnya.

Terakhir, akan dianalisis hubungan dari beberapa variabel numerik dengan beberapa variabel kategorik dengan menggunakan metode ANOVA.

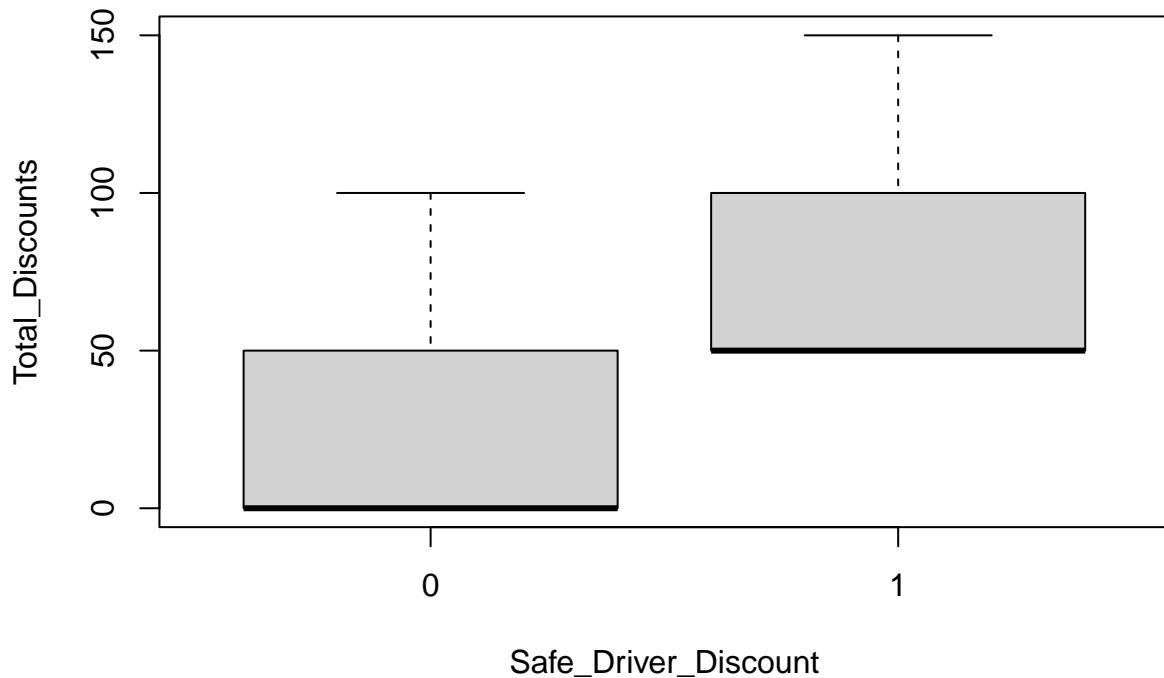
```
#periksa hubungan variabel numerik dgn variabel kategorik (anova)
boxplot(Age ~ Is_Senior)
```



```
anova19 <- aov(Age ~ Is_Senior)
summary(anova19)
```

```
##           Df  Sum Sq Mean Sq F value Pr(>F)
## Is_Senior     1  953107  953107    9335 <2e-16 ***
## Residuals  9998 1020821      102
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

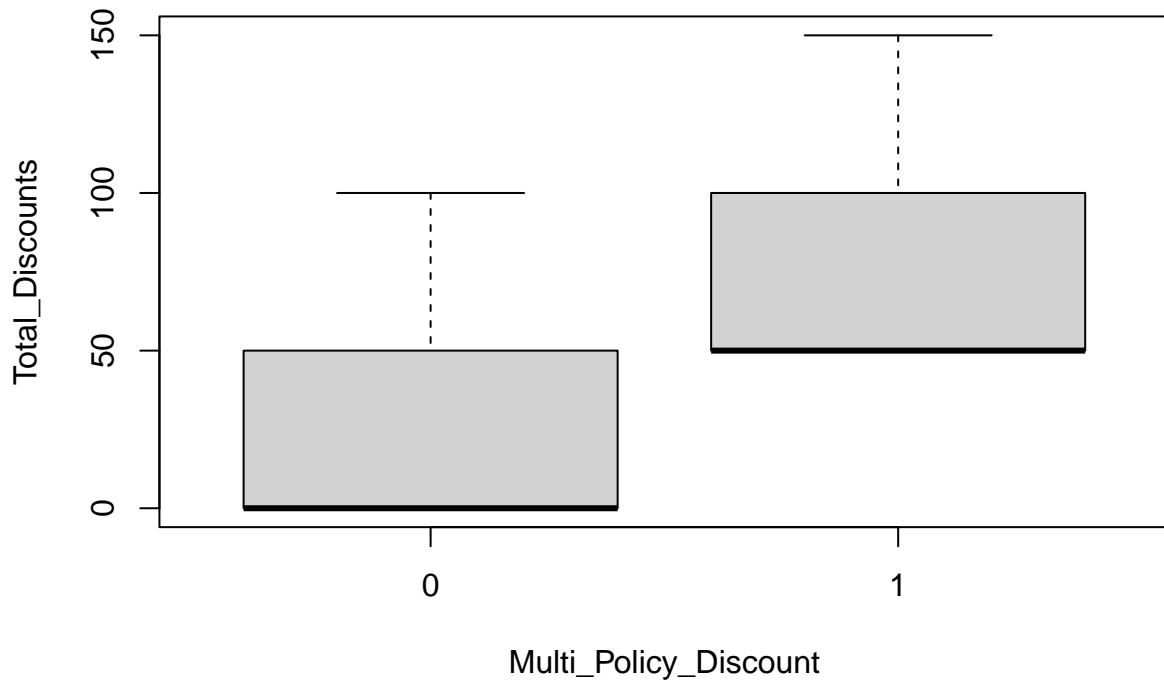
```
boxplot(Total_Discounts ~ Safe_Driver_Discount)
```



```
anova20 <- aov(Total_Discounts ~ Safe_Driver_Discount)
summary(anova20)
```

```
##                               Df  Sum Sq Mean Sq F value Pr(>F)
## Safe_Driver_Discount     1 3908029 3908029    5251 <2e-16 ***
## Residuals                 9998 7440850      744
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

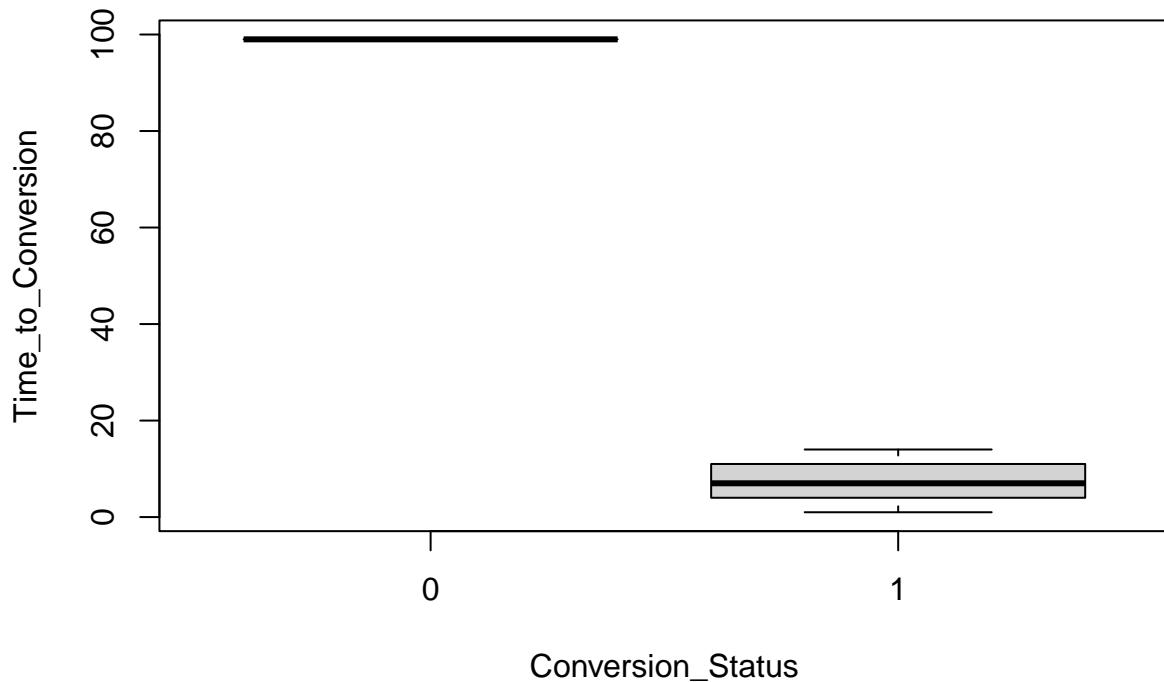
```
boxplot(Total_Discounts ~ Multi_Policy_Discount)
```



```
anova21 <- aov(Total_Discounts ~ Multi_Policy_Discount)
summary(anova21)
```

```
##                                Df  Sum Sq Mean Sq F value Pr(>F)
## Multi_Policy_Discount      1 5198582 5198582    8451 <2e-16 ***
## Residuals                  9998 6150297     615
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

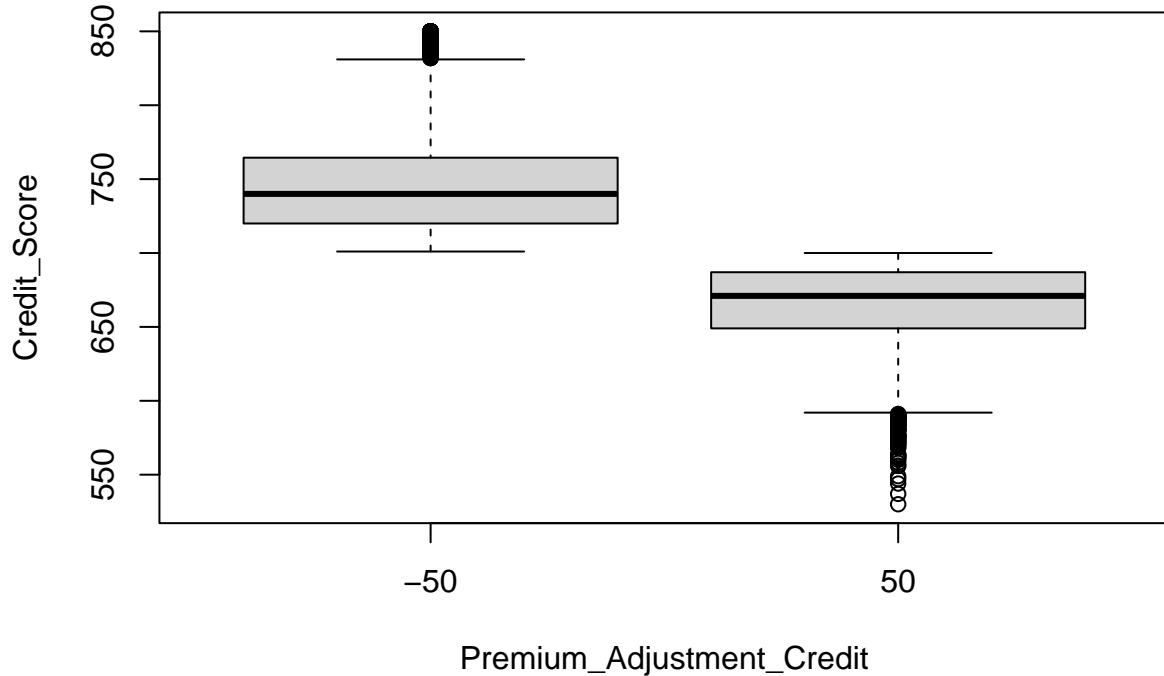
```
boxplot(Time_to_Conversion ~ Conversion_Status)
```



```
anova22 <- aov(Time_to_Conversion ~ Conversion_Status)
summary(anova22)
```

```
##                                Df  Sum Sq Mean Sq F value Pr(>F)
## Conversion_Status      1 20561254 20561254 2227288 <2e-16 ***
## Residuals            9998    92297       9
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
boxplot(Credit_Score ~ Premium_Adjustment_Credit)
```



```
anova23 <- aov(Credit_Score ~ Premium_Adjustment_Credit)
summary(anova23)
```

```
##                               Df  Sum Sq Mean Sq F value Pr(>F)
## Premium_Adjustment_Credit     1 15359866 15359866   16358 <2e-16 ***
## Residuals                  9998 9387774      939
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

par(mfrow=c(1,1))
```

Dari sini, diperoleh variabel-variabel yang saling berhubungan adalah

- Age dan Is\_Senior

Kedua variabel ini berhubungan secara positif. Seseorang yang usianya sudah tua akan diklasifikasikan sebagai lansia.

- Total\_Discounts dan Safe\_Driver\_Discount

Kedua variabel ini berhubungan secara positif. Pelanggan yang mendapat diskon karena riwayat mengemudi aman umumnya juga mendapat total diskon yang banyak.

- Total\_Discounts dan Multi\_Policy\_Discount

Kedua variabel ini berhubungan secara positif. Pelanggan yang mendapat diskon karena memiliki lebih dari satu polis umumnya juga mendapat total diskon yang banyak.

- Time\_to\_Conversion dan Conversion\_Status

Kedua variabel ini berhubungan secara negatif. Pelanggan yang tidak berhasil dikonversi umumnya memerlukan waktu yang lebih lama pelanggan untuk membeli polis.

- Premium\_Adjustment\_Credit dan Credit\_Score

Kedua variabel ini berhubungan secara negatif. Semakin tinggi skor kredit pelanggan, maka semakin tinggi pemotongan preminya. Sebaliknya, jika skor kredit pelanggan rendah, maka premi yang harus dibayar akan semakin mahal.

## Soal 2 - Decision Tree

```
# Membuat kolom Premium_Class berdasarkan Premium_Amount
data$Premium_Class <- ifelse(data$Premium_Amount > 2000, "High", "Low")
data$Premium_Class <- as.factor(data$Premium_Class)

set.seed(123) # untuk hasil yang konsisten

# Ambil indeks untuk Premium_Class == "High"
high_index <- which(data$Premium_Class == "High")
train_high <- sample(high_index, 800)

# Ambil indeks untuk Premium_Class == "Low"
low_index <- which(data$Premium_Class == "Low")
train_low <- sample(low_index, 800)

# Gabungkan indeks training
train_index <- c(train_high, train_low)

# Buat data train dan data test
data_train <- data[train_index, ]
data_test <- data[-train_index, ]
```

Tanpa melakukan validasi silang dengan menggunakan *library* rpart, maka dapat ditetapkan beberapa nilai parameter yaitu

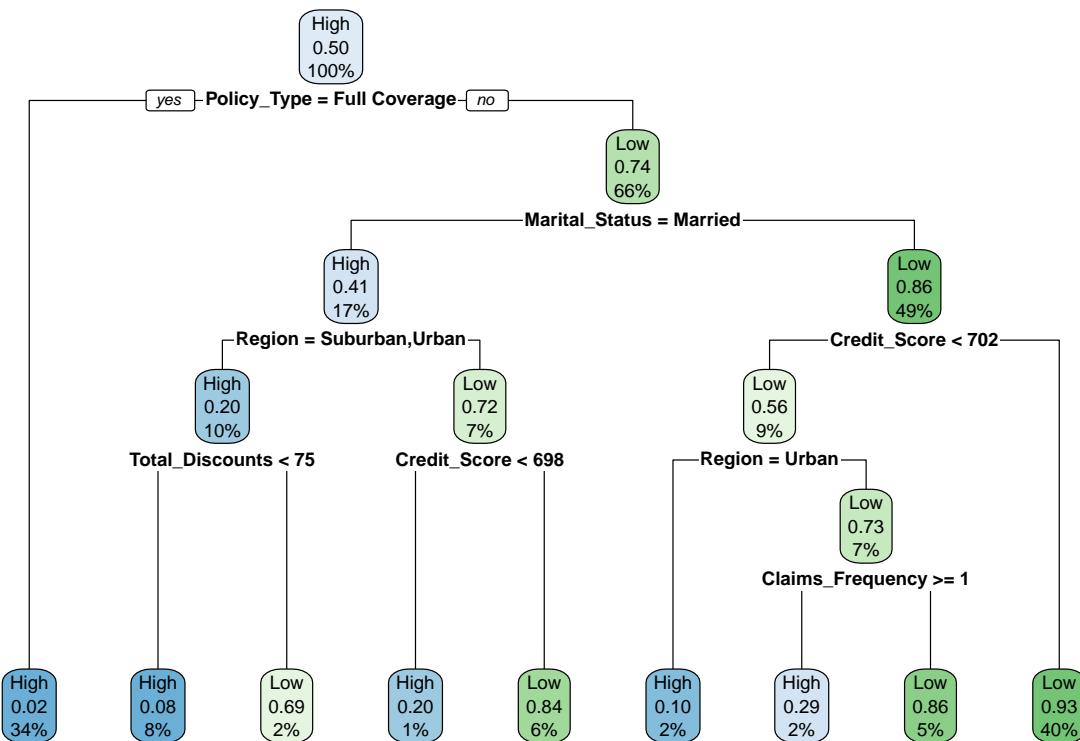
- *parms*: menentukan ukuran apa yang akan digunakan dalam penentuan *splitting*, dapat dipilih antara menggunakan *information gain* atau indeks Gini.
- *complexity* parameter (cp): nilai batas untuk menentukan apakah *splitting* akan dilakukan, sehingga pohon tidak menjadi terlalu kompleks. Nilai cp diformulasikan dalam bentuk

$$\sum_{\text{Terminal Nodes}} \text{Misclass}_i + \lambda \times (\text{Splits}).$$

- *minsplit*: nilai minimum dari banyaknya observasi pada \textit{node} sebelum *splitting* dilakukan.
- *minbucket*: minimum banyaknya observasi pada *leaf node* yang harus terpenuhi.
- *maxdepth*: batas maksimum kedalaman pohon untuk mencegah *overfitting*.

Dengan menerapkan nilai parameter-parameter di atas dan tanpa melakukan validasi silang, dapat diperoleh *decision tree*-nya (sebut sebagai model 1) adalah sebagai berikut:

```
#Membentuk decision tree
full_model1 <- rpart(Premium_Class ~ . - Premium_Amount, data=data_train, parms=list(split=c("information
    cp = 0.01, minsplit=30, minbucket=20, maxdepth=5)
rpart.plot(full_model1)
```



```
# Prediksi pada data_tes
predictions <- predict(full_model1, data_test, type = "class")
confusionMatrix(predictions, data_test$Premium_Class)
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction High   Low
##       High  7329   13
##       Low   865   193
##
##                         Accuracy : 0.8955
##                         95% CI : (0.8887, 0.9019)
##      No Information Rate : 0.9755
##      P-Value [Acc > NIR] : 1
##
##                         Kappa : 0.2756
```

```

## 
##  Mcnemar's Test P-Value : <2e-16
##
##          Sensitivity : 0.8944
##          Specificity : 0.9369
##          Pos Pred Value : 0.9982
##          Neg Pred Value : 0.1824
##          Prevalence : 0.9755
##          Detection Rate : 0.8725
##          Detection Prevalence : 0.8740
##          Balanced Accuracy : 0.9157
##
##          'Positive' Class : High
##

```

Hal ini menunjukkan bahwa variabel-variabel yang penting dalam menentukan *premium class* adalah *policy type*, *marital status*, *region*, *credit score*, *total discounts*, dan *claim frequency*. Hasil prediksi juga menunjukkan nilai metrik yang baik, khususnya *accuracy* sebesar 89,55%, *sensitivity* sebesar 89,44%, dan *specificity* sebesar 93,69% sehingga mengindikasikan model yang tidak *overfit*.

Setelah itu, akan ditampilkan hasil *variable importance plot* dengan menggunakan kode di bawah. Nilai pada plot ini diperoleh berdasarkan penjumlahan dari pengurangan *impurity* yang terjadi dari pemisahan-pemisahan pada seluruh pohon.

```

# Menampilkan daftar variabel penting (Variable Importance)
importance <- full_model1$variable.importance
importance_sorted <- sort(importance, decreasing = TRUE)
print(importance_sorted)

```

	Policy_Adjustment	Policy_Type	Marital_Status
##	452.359723	452.359723	99.856421
##	Married_Premium_Discount	Credit_Score	Premium_Adjustment_Region
##	99.856421	80.647019	63.118723
##	Region	Premium_Adjustment_Credit	Total_Discounts
##	63.118723	62.822298	24.742617
##	Claims_Adjustment	Claims_Frequency	Website_Visits
##	23.019406	14.093037	4.904381
##	Safe_Driver_Discount	Time_Since_First_Contact	Bundling_Discount
##	4.639241	4.262120	1.546414
##	Inquiries	Age	Source_of_Lead
##	1.385442	0.692721	0.692721

Berdasarkan hasil tersebut, dapat diperoleh bahwa variabel *policy adjustment*, *policy type*, *marital status*, dan *married premium discount* termasuk dalam variabel-variabel yang paling signifikan dalam menentukan *premium class*. Akan tetapi, dapat dilihat bahwa ada beberapa variabel yang nilainya serupa atau bahkan sama persis. Untuk efisiensi penggunaan variabel, maka hanya akan dipilih satu diantara banyaknya variabel yang memiliki nilai yang sama. Oleh karena itu, terdapat beberapa variabel yang ditunjukkan signifikan oleh *variable importance plot*-nya, tetapi tidak terdapat pada *decision tree*, seperti *policy adjustment*.

Kurva *Receiver Operating Characteristic* (ROC) adalah kurva yang menggambarkan kemampuan model untuk mengatasi masalah klasifikasi dengan mencoba nilai *threshold* yang berbeda-beda. Sumbu-*x* menyatakan tingkat *specificity* dan sumbu-*y* menyatakan tingkat *sensitivity*. Untuk membandingkan performa antara dua buah model, akan digunakan metrik *Area Under Curve* (AUC) yang menyatakan luas daerah di bawah kurva ROC. Nilai AUC yang semakin mendekati 1 mengindikasikan performa model yang semakin baik. Jika AUC

bernilai 1, maka berarti terdapat suatu *threshold* di mana model dapat secara sempurna mengklasifikasikan *premium class*. Kurva ROC dan nilai AUC dari *decision tree* yang telah dibentuk ditampilkan pada plot di bawah ini.

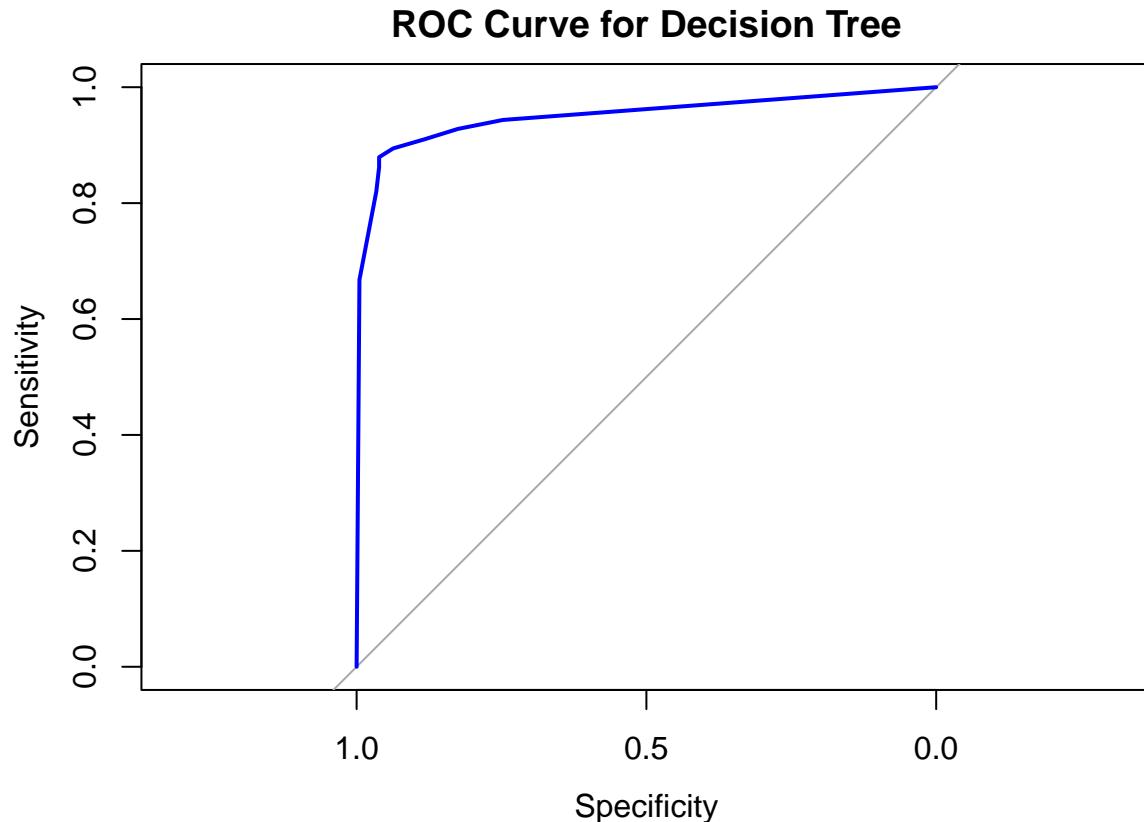
```
#Membuat kurva ROC
library(pROC) # Library untuk ROC curve

# Untuk ROC, kita butuh probabilitas prediksi, bukan class
prob_predictions <- predict(full_model1, data_test, type = "prob")

# Misal Premium_Class ada dua kelas: "Low" dan "High"
# Kita plot ROC untuk salah satu kelas (contohnya kelas "High")
roc_curve <- roc(
  response = data_test$Premium_Class,
  predictor = prob_predictions[, "High"], # Ubah "High" sesuai nama kelas kamu
  levels = rev(levels(data_test$Premium_Class))
)

## Setting direction: controls < cases

# Plot ROC curve
plot(roc_curve, col = "blue", main = "ROC Curve for Decision Tree")
```



```

auc_value <- auc(roc_curve)
print(paste("AUC:", auc_value))

```

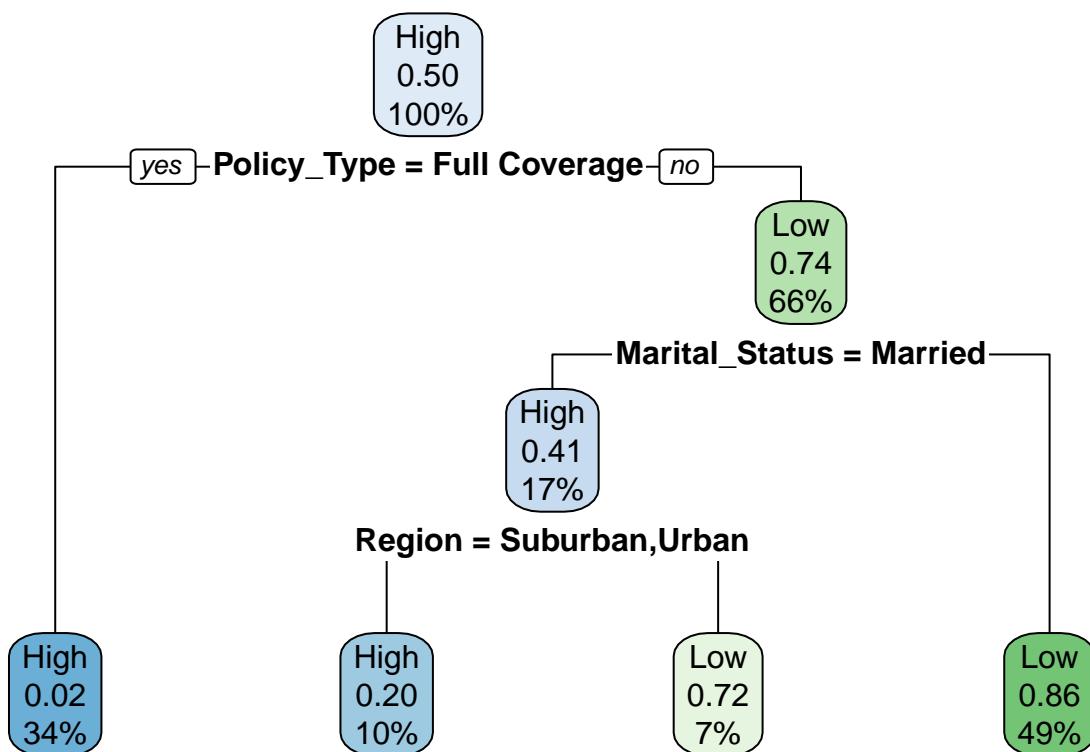
```
## [1] "AUC: 0.949732340263181"
```

Dengan nilai AUC sebesar 0,9497 serta metrik-metrik yang sudah ditampilkan dalam *confusion matrix*, maka dapat dikatakan bahwa model *decision tree* dapat secara baik mengklasifikasi *premium class*. Berikutnya, Eksplorasi parameter-parameter yang telah dijelaskan juga akan dilakukan dan divisualisasikan pada 4 plot di bawah.

```

#Membentuk decision tree
full_model2 <- rpart(Premium_Class ~ . - Premium_Amount, data_train, parms=list(split=c("information")),
                      cp = 0.02, minsplit=30, minbucket=20, maxdepth=5)
rpart.plot(full_model2)

```



```

# Prediksi pada data_tes
predictions <- predict(full_model2, data_test, type = "class")
confusionMatrix(predictions, data_test$Premium_Class)

```

```

## Confusion Matrix and Statistics
##
##             Reference
## Prediction High   Low
##           High  6847    18

```

```

##      Low   1347   188
##
##          Accuracy : 0.8375
##                95% CI : (0.8294, 0.8453)
##    No Information Rate : 0.9755
##    P-Value [Acc > NIR] : 1
##
##          Kappa : 0.1805
##
##  Mcnemar's Test P-Value : <2e-16
##
##          Sensitivity : 0.8356
##          Specificity : 0.9126
##    Pos Pred Value : 0.9974
##    Neg Pred Value : 0.1225
##          Prevalence : 0.9755
##    Detection Rate : 0.8151
##  Detection Prevalence : 0.8173
##    Balanced Accuracy : 0.8741
##
##    'Positive' Class : High
##

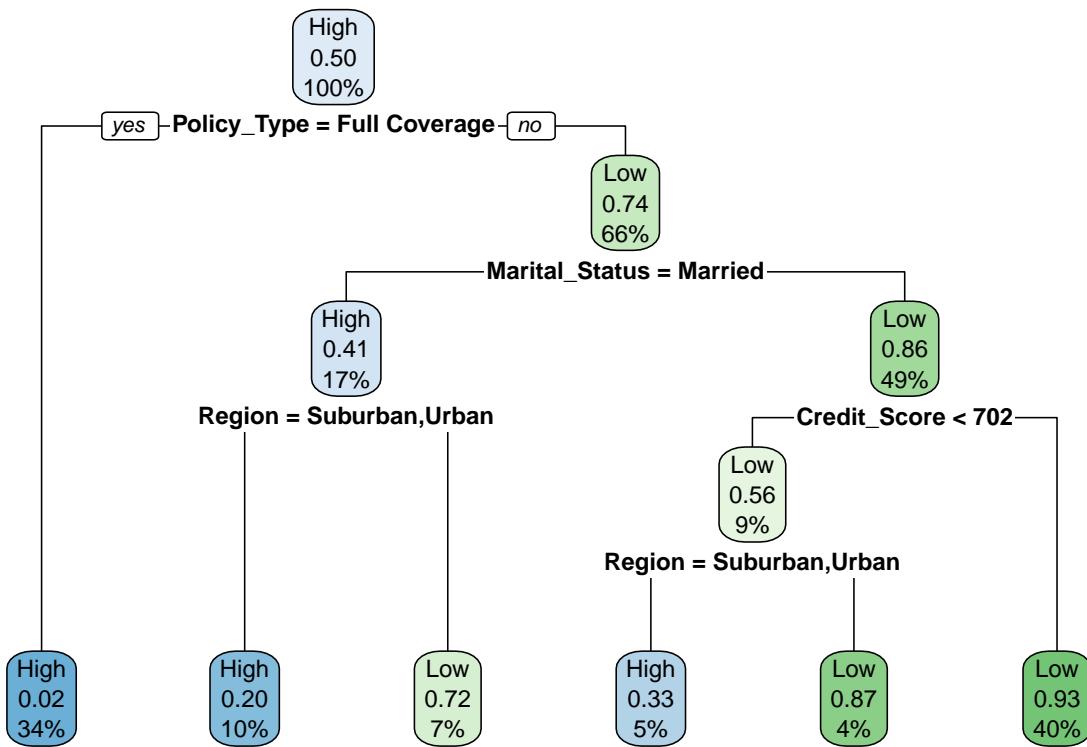
```

*#Membentuk decision tree*

```

full_model3 <- rpart(Premium_Class~. - Premium_Amount, data_train, parms=list(split=c("information")),
                      cp = 0.01, minsplit=100, minbucket=40, maxdepth=5)
rpart.plot(full_model3)

```



```

# Prediksi pada data_tes
predictions <- predict(full_model3, data_test, type = "class")
confusionMatrix(predictions, data_test$Premium_Class)

```

```

## Confusion Matrix and Statistics
##
##             Reference
## Prediction High   Low
##       High 7362    26
##       Low   832    180
##
##                 Accuracy : 0.8979
##                 95% CI : (0.8912, 0.9043)
##     No Information Rate : 0.9755
##     P-Value [Acc > NIR] : 1
##
##                 Kappa : 0.2656
##
## McNemar's Test P-Value : <2e-16
##
##                 Sensitivity : 0.8985
##                 Specificity : 0.8738
##     Pos Pred Value : 0.9965
##     Neg Pred Value : 0.1779
##     Prevalence : 0.9755
## Detection Rate : 0.8764

```

```

##      Detection Prevalence : 0.8795
##      Balanced Accuracy : 0.8861
##
##      'Positive' Class : High
##

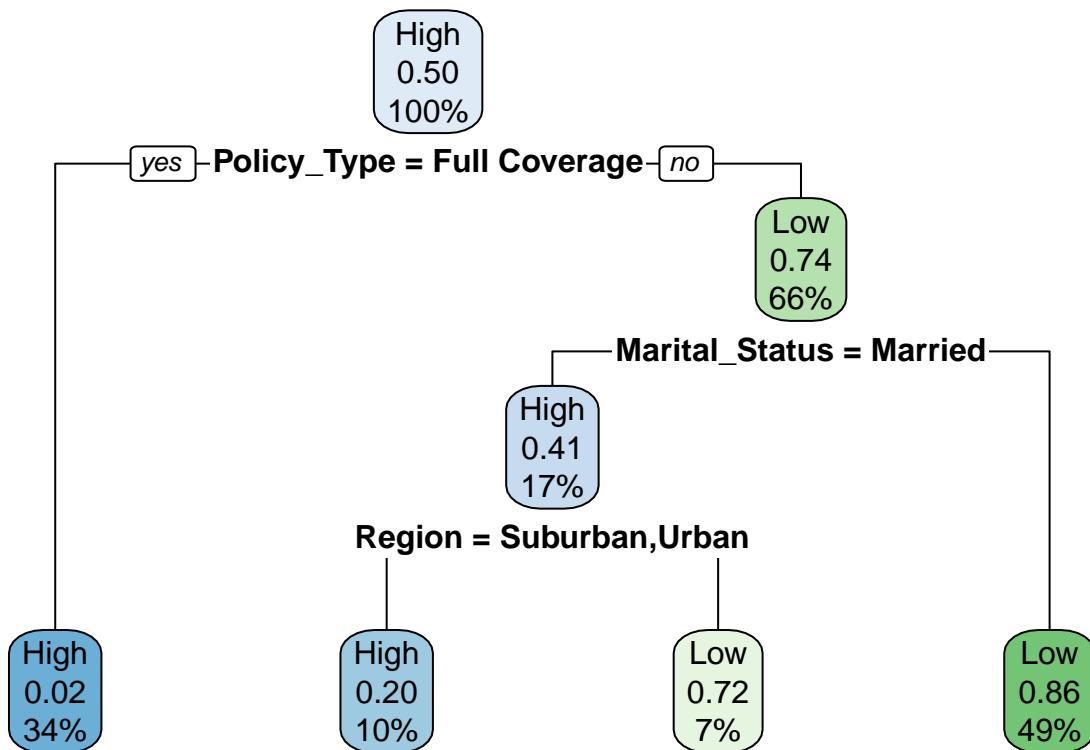
```

*#Membentuk decision tree*

```

full_model4 <- rpart(Premium_Class~. - Premium_Amount, data_train, parms=list(split=c("information")),
                      cp = 0.01, minsplit=30, minbucket=20, maxdepth=3)
rpart.plot(full_model4)

```



```

# Prediksi pada data_tes
predictions <- predict(full_model4, data_test, type = "class")
confusionMatrix(predictions, data_test$Premium_Class)

```

```

## Confusion Matrix and Statistics
##
##      Reference
##      Prediction High   Low
##      High     6847   18
##      Low      1347  188
##
##      Accuracy : 0.8375
##      95% CI : (0.8294, 0.8453)
##      No Information Rate : 0.9755

```

```

##      P-Value [Acc > NIR] : 1
##
##          Kappa : 0.1805
##
##  Mcnemar's Test P-Value : <2e-16
##
##          Sensitivity : 0.8356
##          Specificity : 0.9126
##          Pos Pred Value : 0.9974
##          Neg Pred Value : 0.1225
##          Prevalence : 0.9755
##          Detection Rate : 0.8151
##          Detection Prevalence : 0.8173
##          Balanced Accuracy : 0.8741
##
##          'Positive' Class : High
##

```

Model 2, 3, dan 4 secara berturut-turut mengganti nilai *cp*, *minsplit* bersamaan dengan *minbucket*, dan *maxdepth*. Perubahan nilai metrik-metrik *accuracy*, *sensitivity*, dan *specificity* juga akan dievaluasi pada bagian ini. Saat nilai *cp* diperbesar, maka model menjadi lebih sederhana dan terjadi pengurangan pada seluruh nilai metrik. Saat nilai *minsplit* dan *minbucket* diperbesar, maka model juga menjadi lebih sederhana dan terjadi peningkatan yang kecil pada *accuracy* dan *sensitivity*, sedangkan terjadi penurunan yang signifikan pada *specificity*. Jika nilai dari *maxdepth* diperkecil, maka model juga akan menjadi lebih sederhana, dan terjadi penurunan pada seluruh nilai metrik tersebut.

Fungsi *train control* akan menentukan teknik yang akan digunakan untuk melatih atau memvalidasi model. Dengan menetapkan metodenya sebagai *cv* dan *number = 10*, berarti model akan divalidasi dengan metode *cross validation* di mana data dibagi menjadi 10 bagian. Masing-masing bagian akan digunakan sebagai data uji dan kesembilan bagian lainnya akan digunakan sebagai data latih.

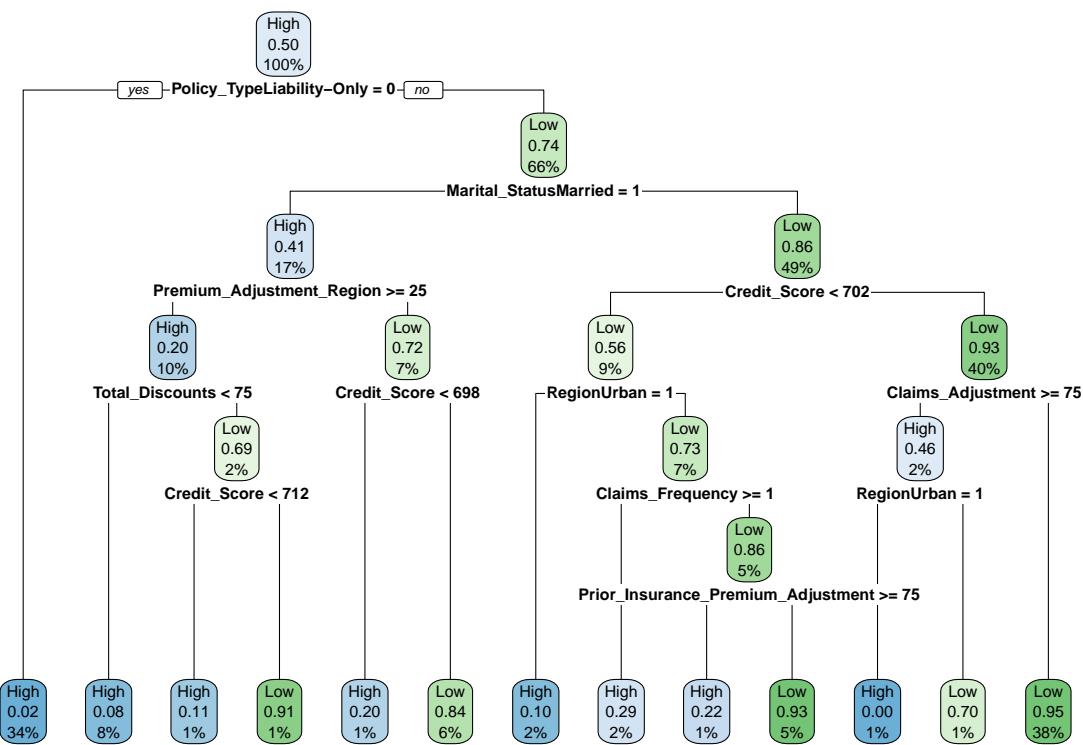
Pada fungsi *train*, ditetapkan nilai dari *tuneLength*-nya adalah 10. Artinya, R akan secara otomatis menentukan 10 nilai *cp* yang bervariasi dan biasanya mengecil seperti 0,1, 0,05, dan 0,01. Kemudian, untuk setiap nilai *cp*, akan diterapkan *cross validation* dan model akan divalidasi. Akan dipilih nilai *cp* pada model dengan performa terbaik. Jika menggunakan *rpart* saja, maka nilai dari *cp* akan ditentukan secara manual yang dapat menyebabkan model menjadi *underfit* maupun *overfit*.

```

ctrl <- trainControl(method = "cv", number = 10)
fit_tree <- train(Premium_Class ~ ., data = data_train[, -which(names(data_train) == "Premium_Amount")])

rpart.plot(fit_tree$finalModel)

```



```

# Prediksi pada data tes
predictions <- predict(fit_tree, newdata = data_test)

# Confusion matrix
confusionMatrix(predictions, data_test$Premium_Class)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction High   Low
##       High 7521   20
##       Low   673   186
##
##                         Accuracy : 0.9175
##                         95% CI : (0.9114, 0.9233)
##      No Information Rate : 0.9755
##      P-Value [Acc > NIR] : 1
##
##                         Kappa : 0.3225
##
##      Mcnemar's Test P-Value : <2e-16
##
##                         Sensitivity : 0.9179
##                         Specificity  : 0.9029
##      Pos Pred Value : 0.9973
##      Neg Pred Value : 0.2165

```

```

##          Prevalence : 0.9755
##          Detection Rate : 0.8954
##  Detection Prevalence : 0.8977
##          Balanced Accuracy : 0.9104
##
##          'Positive' Class : High
##


# Get the variable importance
importance <- varImp(fit_tree, scale = FALSE)
print(importance)

## rpart variable importance
##
##    only 20 most important variables shown (out of 34)
##
##                                     Overall
## Policy_TypeLiability-Only      373.493
## Policy_Adjustment               373.493
## Credit_Score                   243.546
## Married_Premium_Discount       212.318
## Marital_StatusMarried          212.318
## Premium_Adjustment_Credit      128.685
## Claims_Adjustment              116.098
## Premium_Adjustment_Region       76.112
## Prior_Insurance_Premium_Adjustment 71.650
## RegionUrban                    59.193
## Claims_Frequency                55.053
## Total_Discounts                 38.765
## Prior_Insurance>5 years         32.695
## Bundling_Discount               9.305
## Safe_Driver_Discount            6.356
## RegionSuburban                  5.945
## Website_Visits                  3.427
## Time_Since_First_Contact        1.815
## Inquiries                        1.350
## Claims_SeverityLow               0.000

# Make probability predictions (not just the classes)
pred_prob <- predict(fit_tree, newdata = data_test, type = "prob")

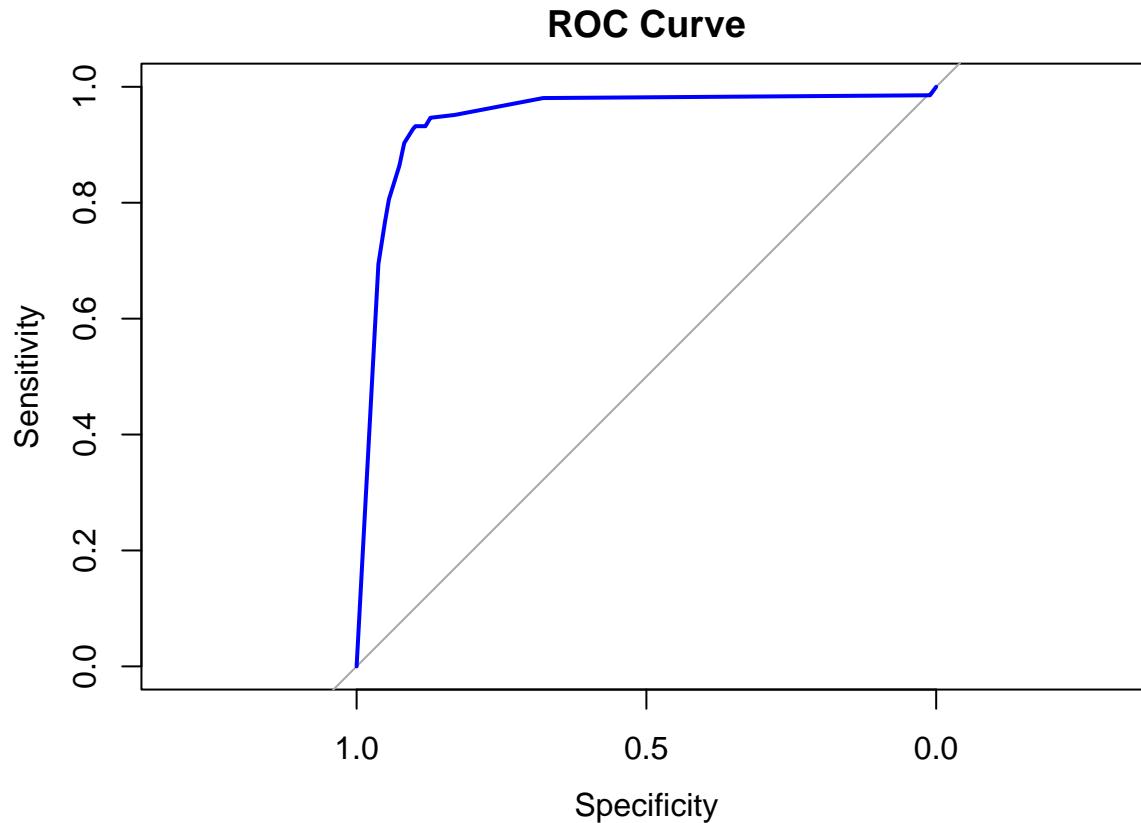
# Assuming your Premium_Class has two levels and the positive class is the first level
roc_curve <- roc(data_test$Premium_Class, pred_prob[, 2]) # Use the probabilities for the positive class

## Setting levels: control = High, case = Low

## Setting direction: controls < cases

# Plot the ROC curve
plot(roc_curve, main = "ROC Curve", col = "blue", lwd = 2)

```



```
auc_value <- auc(roc_curve)
print(paste("AUC:", auc_value))
```

```
## [1] "AUC: 0.94497453737165"
```

Hasil dari *decision tree* yang terbentuk dengan menggunakan *cross validation* ternyata lebih kompleks dibandingkan model pertama dari *decision tree* yang dibentuk (model 1). Hal ini disebabkan karena pada kode tersebut tidak ditetapkan batasan parameter seperti *minsplit* dan *maxbucket*. Hasil prediksi juga menunjukkan peningkatan dalam hal *accuracy* dan *sensitivity* yaitu masing-masing menjadi 91,75% dan 91,79%, tetapi terjadi penurunan dalam *specificity* yaitu menjadi 90,29%.

Selain itu, *variable importance plot* juga ditampilkan dan menunjukkan hasil yang sedikit berbeda dibandingkan *variable importance plot* pada model 1. Sebagai contoh, jenis polis yang awalnya merupakan *full coverage* berubah menjadi *liability only*. Walaupun demikian, terdapat pula variabel-variabel yang masih dianggap sama sebagai variabel yang paling signifikan dalam menentukan *premium class* yaitu *policy adjustment*, *marital status*, dan *married premium discount*.

Terakhir, ditunjukkan plot dari kurva ROC sehingga diperoleh nilai AUC sebesar 0,9449 di mana nilai ini sedikit lebih rendah dibandingkan AUC pada model 1 yaitu sebesar 0,9497. Telah dijelaskan bahwa nilai AUC memberikan informasi mengenai seberapa baik model dapat mengklasifikasi *premium class* menjadi *high* atau *low*. Oleh karena itu, walaupun pada umumnya *decision tree* dengan *cv* mengungguli *decision tree* tanpa *cv*, namun kali ini model *decision tree* tanpa *cv* akan dipilih sebagai model terbaik yang dapat mengatasi masalah klasifikasi *premium class*.

Sekarang, akan diberikan interpretasi dari model 1 yakni model pertama *decision tree* yang dibentuk untuk menentukan faktor-faktor apa yang mempengaruhi seseorang memiliki premi yang besar atau dengan kata

lain termasuk dalam *premium class* yang tinggi (*high*). Berikut adalah kondisi-kondisi yang akan menyebabkan premi seseorang besar:

- Memiliki tipe polis yang *full coverage* karena jaminan risiko yang lebih tinggi akan mengakibatkan nilai premi semakin meningkat pula.
- Memiliki tipe polis selain *full coverage*, telah menikah, masuk dalam wilayah *suburban* atau *urban* dan menerima diskon kurang dari 75, atau tidak masuk ke dua wilayah tersebut namun memiliki *credit score* kurang dari 698.
- Memiliki tipe polis selain *full coverage*, belum menikah, memiliki *credit score* kurang dari 702, masuk dalam wilayah *urban* atau tidak masuk dalam wilayah *urban* tetapi frekuensi klaimnya lebih dari 1.

Sebagai penjelasan ke divisi *marketing*, premi seseorang yang besar cenderung dipengaruhi oleh enam faktor yaitu tipe polis, status menikah, wilayah, besar diskon yang diperoleh, skor kredit, dan frekuensi klaim. Jika seseorang memiliki tipe polis yang *full coverage*, maka hampir dapat dipastikan orang tersebut memiliki premi yang besar. Selanjutnya, jika ia tidak memiliki tipe polis *full coverage*, maka kita akan lanjut untuk melihat status menikah. Namun, melihat seseorang sudah menikah atau belum tidak dapat sepenuhnya memastikan kelas premi dari orang tersebut.

Jika ia belum menikah, maka tinjau wilayah dari orang tersebut. Jika wilayahnya adalah *suburban* atau *urban* dan menerima diskon kurang dari 75, maka ia akan memiliki premi yang besar. Terdapat juga kasus lain yang mengindikasikan seseorang memiliki premi besar yaitu ia tidak tinggal dalam dua wilayah tersebut, tetapi skor kreditnya kurang dari 698.

Sekarang, jika ia sudah menikah, maka tinjau apakah skor kreditnya kurang dari 702. Jika pernyataan tersebut benar, maka lihat apakah ia tinggal dalam wilayah *urban*. Jika ia tinggal dalam wilayah tersebut, maka ia memiliki premi yang tinggi. Akan tetapi, jika ia tidak tinggal dalam wilayah tersebut, maka frekuensi klaim yang melebihi sekali akan membuat orang tersebut memiliki premi yang besar.

### Soal 3 - Random Forest dan Boosting

```
#Penghapusan variabel 'Premium Amount'  
train_data <- data_train %>% select(-Premium_Amount)  
test_data <- data_test %>% select(-Premium_Amount)  
  
# Ubah semua karakter jadi factor  
train_data[] <- lapply(train_data, function(x) {  
  if (is.character(x)) as.factor(x) else x  
})  
  
test_data[] <- lapply(test_data, function(x) {  
  if (is.character(x)) as.factor(x) else x  
})
```

Dengan pembagian data latih dan data uji yang sama seperti pada saat membuat model decision tree, akan dibuat model random forest dan *Gradient Boosting Machine* (GBM) untuk memprediksi kelas premidari pemegang polis.

```
# Model Random Forest  
set.seed(123)  
rf_model <- randomForest(Premium_Class ~ ., data = train_data,
```

```

ntree = 500, mtry = 5, nodesize=10, importance = TRUE)

# Top 5 fitur penting
rf_imp <- importance(rf_model)
rf_top5 <- head(rf_imp[order(rf_imp[, 1], decreasing = TRUE), , drop = FALSE], 5)
print(rf_top5)

##                                     High      Low MeanDecreaseAccuracy
## Policy_Adjustment      27.65014 27.72410          27.97396
## Policy_Type            23.68668 23.97918          24.08971
## Premium_Adjustment_Region 22.95751 30.43377          31.14921
## Married_Premium_Discount 21.80828 24.55664          24.63249
## Marital_Status          21.09731 23.31775          23.60045
##                                     MeanDecreaseGini
## Policy_Adjustment          172.36800
## Policy_Type                148.27242
## Premium_Adjustment_Region  30.59254
## Married_Premium_Discount  49.80428
## Marital_Status              46.01448

```

Lima fitur terpenting dalam model Random Forest adalah *policy adjustment*, *policy type*, *premium adjustment region*, *married premium discount*, dan *marital status*.

Selain itu, dapat dilihat juga nilai *Mean Decrease Accuracy* dan *Mean decrease Gini* dari masing-masing fitur. Nilai *Mean Decrease Accuracy* menunjukkan seberapa besar penurunan akurasi model jika nilai suatu fitur diacak. Sementara itu, *Mean Decrease Gini* mengukur kontribusi fitur dalam memecah node pada pohon-pohon dalam Random Forest. Semakin tinggi kedua nilai, semakin besar peran fitur tersebut dalam membangun model. Perhatikan juga bahwa terdapat nilai “*High*” dan “*Low*” untuk masing-masing fitur. Nilai ini memberikan gambaran mengenai sejauh mana fitur memengaruhi masing-masing kelas target (“*High*” dan “*Low*”).

Sebagai contoh, fitur Policy Adjustment memiliki kontribusi sebesar 172.368 dalam pemecahan node, dan jika nilainya diacak, akurasi model menurun sebesar 27.9%. Fitur ini juga memberikan pengaruh sebesar 27.65014 terhadap prediksi kelas “*High*” dan 27.7241 terhadap kelas “*Low*”.

Selanjutnya, akan diprediksi kelas premium dari data uji yang dimiliki dengan menggunakan model Random Forest yang sudah dibangun.

```

# Prediksi model Random Forest test data
rf_pred2 <- predict(rf_model, newdata = test_data)

# Confusion matrix
confusionMatrix(rf_pred2, test_data$Premium_Class)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction High   Low
##       High    7841    5
##       Low     353   201
##
##               Accuracy : 0.9574
##                     95% CI : (0.9528, 0.9616)
## No Information Rate : 0.9755

```

```

##      P-Value [Acc > NIR] : 1
##
##                  Kappa : 0.5115
##
##  McNemar's Test P-Value : <2e-16
##
##                  Sensitivity : 0.9569
##                  Specificity : 0.9757
##      Pos Pred Value : 0.9994
##      Neg Pred Value : 0.3628
##                  Prevalence : 0.9755
##      Detection Rate : 0.9335
##  Detection Prevalence : 0.9340
##      Balanced Accuracy : 0.9663
##
##      'Positive' Class : High
##

```

Dari confusion matrix di atas, dapat dilihat akurasi prediksi modelnya. Model Random Forest memiliki akurasi sebesar 95.74%. Karena nilainya yang dekat dengan 1, dapat dikatakan bahwa model ini dapat memprediksi kelas premium dengan baik.

```

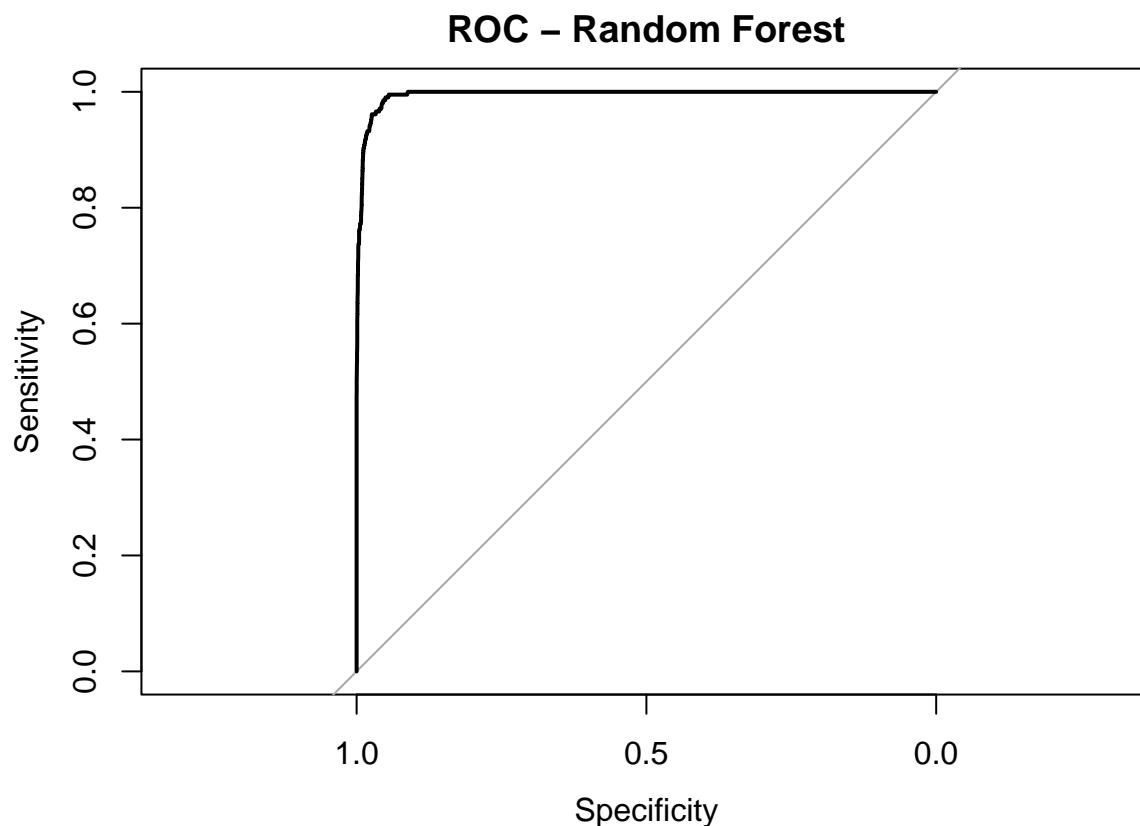
# ROC AUC Test
rf_prob2 <- predict(rf_model, newdata = test_data, type = "prob")[, "High"]
rf_roc2 <- roc(test_data$Premium_Class, rf_prob2)

## Setting levels: control = High, case = Low

## Setting direction: controls > cases

plot(rf_roc2, main = "ROC - Random Forest")

```



```
auc(rf_roc2)
```

```
## Area under the curve: 0.9952
```

Selain dengan melihat akurasi model, dapat pula dilihat performa model dengan menggunakan kurva *Receiver Operating Characteristic* (ROC) *Area Under the Curve* (AUC). Kurva ROC menunjukkan *trade-off* antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR). Pada kurva ROC, nilai FPR merupakan sumbu x dan nilai TPR merupakan sumbu y. True positive rate sendiri merupakan proporsi dari banyaknya orang yang diklasifikasikan sebagai positif dan benar merupakan positif, sedangkan FPR merupakan proporsi banyaknya observasi yang diklasifikasikan sebagai positif, namun sebenarnya adalah negatif. Semakin tinggi nilai y dan semakin rendah nilai x, maka semakin baik model dalam memprediksi. Pada kasus ini, kelas “*High*” merupakan positif dan kelas “*Low*” merupakan negatif. Dari kurva ROC, dapat diperoleh nilai AUC. Nilai ini merupakan luas di bawah kurva yang nilainya diantara 0 sampai 1. Semakin mendekati 1 nilainya, semakin baik model membedakan kedua kelas.

Pada model Random Forest yang dimiliki, dapat dilihat bahwa nilai AUC-nya adalah 0.9952. Nilai ini sangat dekat dengan 1, yang artinya model Random Forest yang dimiliki dapat memprediksi kelas premi dengan sangat baik.

Kemudian, akan dicari nilai parameter “*mtry*” pada model Random Forest agar memiliki nilai AUC tertinggi. Nilai “*mtry*” sendiri merupakan jumlah fitur yang diacak dan dipertimbangkan pada setiap node dalam sebuah pohon keputusan.

```
# Set up trainControl untuk cross-validation
control <- trainControl(
  method = "cv",
```

```

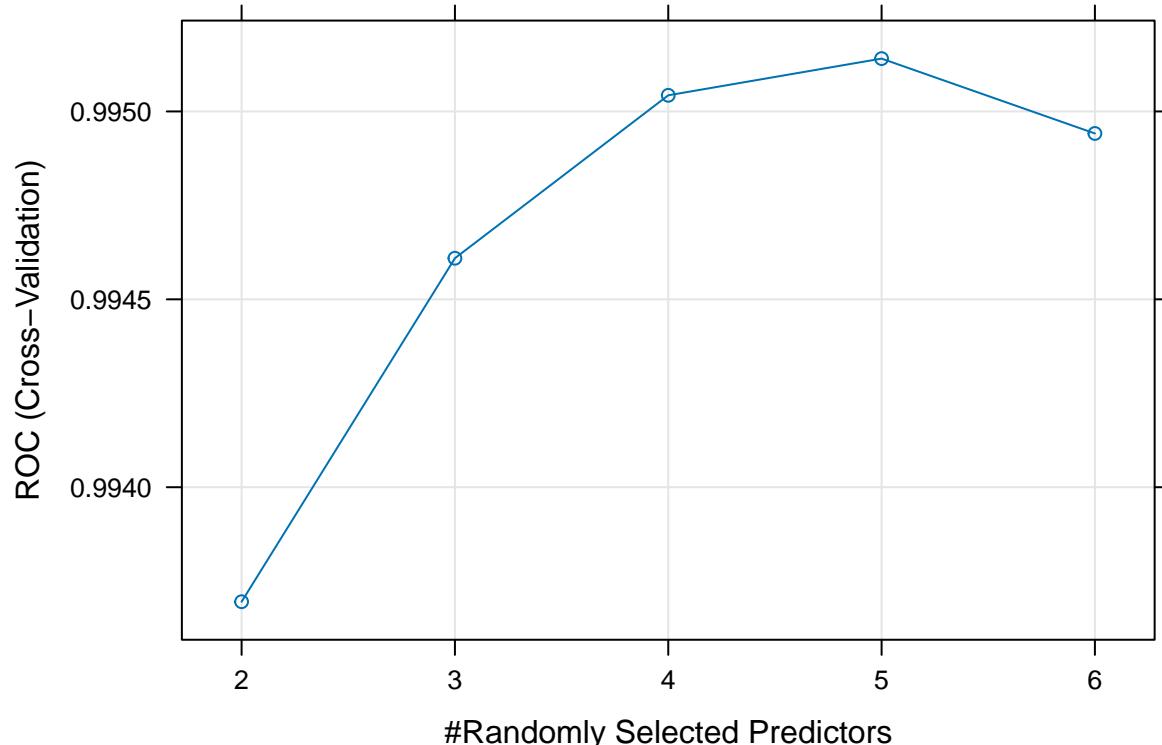
number = 5,
classProbs = TRUE,
summaryFunction = twoClassSummary,
savePredictions = "final"
)

# Tuning model Random Forest dengan parameter mtry
set.seed(123)

tuned_rf <- train(
  Premium_Class ~ .,
  data = train_data,
  method = "rf",
  metric = "ROC",
  tuneGrid = expand.grid(mtry = c(2, 3, 4, 5, 6)),
  trControl = control,
  ntree = 500
)

# Hasil
plot(tuned_rf)

```



```
print(tuned_rf)
```

```
## Random Forest
```

```

## 
## 1600 samples
##   26 predictor
##   2 classes: 'High', 'Low'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 1280, 1280, 1280, 1280, 1280
## Resampling results across tuning parameters:
##
##   mtry   ROC      Sens      Spec
##   2      0.9936953 0.91750  0.99000
##   3      0.9946094 0.93625  0.98750
##   4      0.9950430 0.94375  0.98625
##   5      0.9951406 0.94125  0.98500
##   6      0.9949414 0.95000  0.98500
##
## ROC was used to select the optimal model using the largest value.
## The final value used for the model was mtry = 5.

best_rf_model <- tuned_rf$finalModel
cat("\nBest Random Forest Model (bestTune):\n")

##
## Best Random Forest Model (bestTune):

print(tuned_rf$bestTune)

##   mtry
## 4     5

```

Dengan menggunakan fungsi train dan trainControl, diperoleh bahwa nilai "mtry" yang menghasilkan AUC tertinggi adalah 4 atau 5. Lebih lanjut, AUC untuk 4 *mtry* adalah 0.9950430, sedangkan untuk 5 *mtry* adalah 0.9951406. Dengan selisih yang tipis, nilai untuk 5 *mtry* lebih tinggi dibandingkan dengan 4 *mtry*, sehingga dapat disimpulkan bahwa untuk memperoleh model terbaik guna memprediksi kelas premi dari model Random Forest yang menggunakan 500 pohon adalah dengan mengatur *mtry*-nya menjadi 5.

```

# Mengonversi Premium_Class ke 0 dan 1
train_biner <- train_data
test_biner <- test_data

train_biner$Premium_Class <- ifelse(train_biner$Premium_Class == "High", 1, 0)
test_biner$Premium_Class <- ifelse(test_biner$Premium_Class == "High", 1, 0)

```

Selanjutnya, akan dibangun model GBM untuk memprediksi kelas premi.

```

# Model GBM
set.seed(123)
gbm_model <- gbm(Premium_Class ~ ., data = train_biner,
                  distribution = "bernoulli",
                  n.trees = 5000, interaction.depth = 3, n.minobsinnode=10,
                  shrinkage = 0.05, verbose = FALSE)

```

```
# Top 5 fitur penting
gbm_imp <- summary(gbm_model, plotit = FALSE)
head(gbm_imp, 5)
```

```
##                                     var   rel.inf
## Policy_Type                  Policy_Type 32.716123
## Policy_Adjustment            Policy_Adjustment 15.733535
## Credit_Score                 Credit_Score  8.519626
## Married_Premium_Discount Married_Premium_Discount 7.275101
## Region                      Region    5.423863
```

Pada model GBM, lima fitur terpentingnya adalah *policy type*, *policy adjustment*, *credit score*, *married premium discount*, dan *region*. Dapat dilihat juga nilai *relative influence* dari masing-masing fitur. Nilai ini menunjukkan persentase kontribusi sebuah fitur terhadap keseluruhan prediksi model. Pada model ini, terlihat bahwa fitur *policy type* merupakan fitur yang paling besar memberikan kontribusi dalam memprediksi kelas premi, di mana kontribusinya adalah sebesar 32.72%.

```
#Akurasi Test Data
gbm_prob2 <- predict(gbm_model, newdata = test_data, n.trees = 5000, type = "response")
gbm_pred2 <- ifelse(gbm_prob2 > 0.5, "High", "Low")
gbm_pred2 <- factor(gbm_pred2, levels = c("Low", "High"))

#Confusion matrix
confusionMatrix(gbm_pred2, test_data$Premium_Class)
```

```
## Warning in confusionMatrix.default(gbm_pred2, test_data$Premium_Class): Levels
## are not in the same order for reference and data. Refactoring data to match.
```

```
## Confusion Matrix and Statistics
##
##             Reference
## Prediction High   Low
##       High 7883    1
##       Low   311   205
##
##                   Accuracy : 0.9629
##                           95% CI : (0.9586, 0.9668)
##   No Information Rate : 0.9755
##   P-Value [Acc > NIR] : 1
##
##                   Kappa : 0.5522
##
##   Mcnemar's Test P-Value : <2e-16
##
##                   Sensitivity : 0.9620
##                   Specificity  : 0.9951
##   Pos Pred Value : 0.9999
##   Neg Pred Value : 0.3973
##   Prevalence     : 0.9755
##   Detection Rate : 0.9385
##   Detection Prevalence : 0.9386
##   Balanced Accuracy : 0.9786
```

```
##  
##      'Positive' Class : High  
##
```

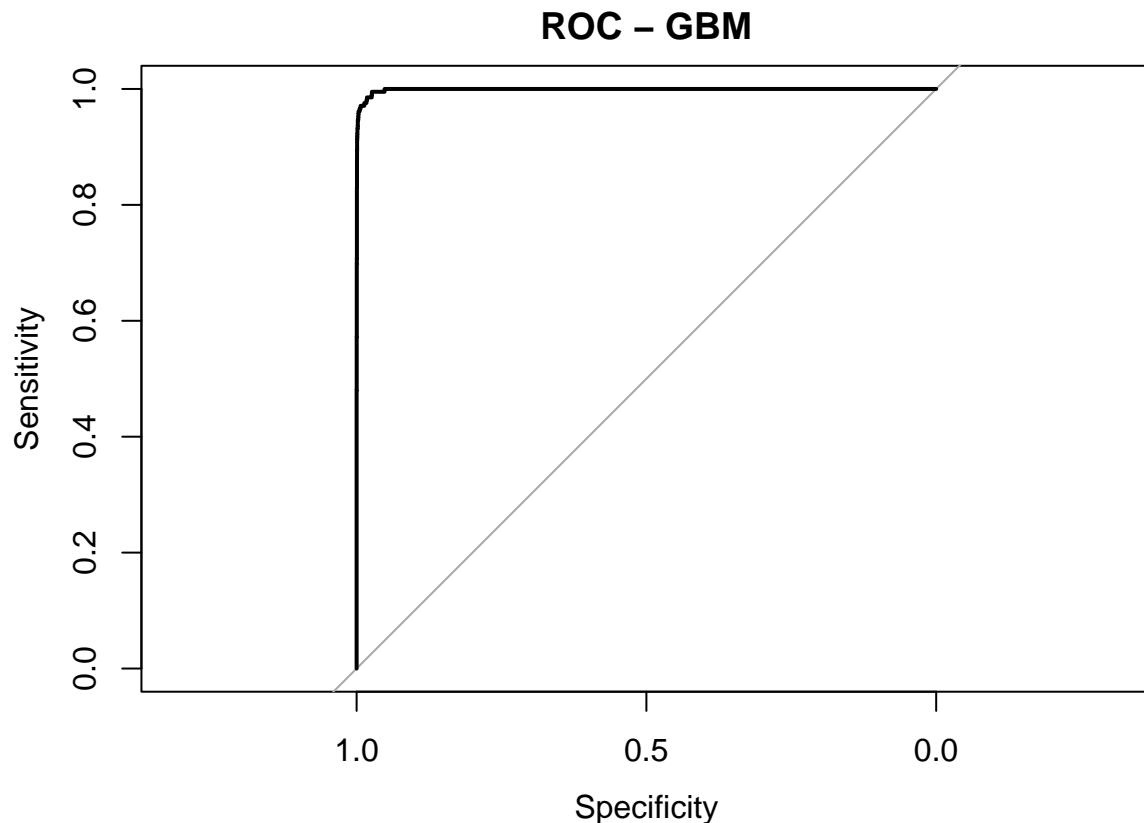
Terlihat dari *confusion matrix* untuk model GBM, akurasi model ini adalah 96.29%. Sebelumnya, diperoleh bahwa unutk model Random Forest nilai akurasinya adalah 95.74%. Artinya, model GBM lebih akurat dalam memprediksi kelas premi dibandingkan dengan model Random Forest.

```
# ROC AUC Test  
gbm_roc2 <- roc(test_data$Premium_Class, gbm_prob2)
```

```
## Setting levels: control = High, case = Low
```

```
## Setting direction: controls > cases
```

```
plot(gbm_roc2, main = "ROC - GBM")
```



```
auc(gbm_roc2)
```

```
## Area under the curve: 0.999
```

Dari kurva ROC di atas, dapat dilihat bahwa nilai TPR sangat tinggi dan nilai FPR relatif rendah. Ini juga tercerminkan dari nilai AUC-nya yang sangat dekat dengan 1, yakni 0.999. Sebelumnya pada model

Random Forest, nilai AUC untuk modelnya adalah 0.9952. Dengan selisih yang tipis, dapat disimpulkan bahwa model GBM lebih baik dalam memprediksi kelas premi.

Kemudian, akan dicari pula nilai-nilai untuk parameter “n.trees”, “interaction.depth”, “shrinkage”, dan “n.minobsinnode”. Parameter “n.trees” menunjukkan jumlah pohon yang digunakan pada model tersebut, di mana semakin banyak pohon yang dibangun, semakin kompleks modelnya. Harapannya model yang lebih kompleks tersebut mampu menangkap pola data dengan lebih baik, meskipun berisiko overfitting jika jumlahnya terlalu besar.

Selanjutnya, parameter “interaction.depth” menentukan seberapa dalam pohon dapat bercabang, atau seberapa rumit interaksi antar fitur yang bisa ditangkap oleh model. Semakin besar nilainya, semakin dalam model dalam memahami hubungan antar variabel.

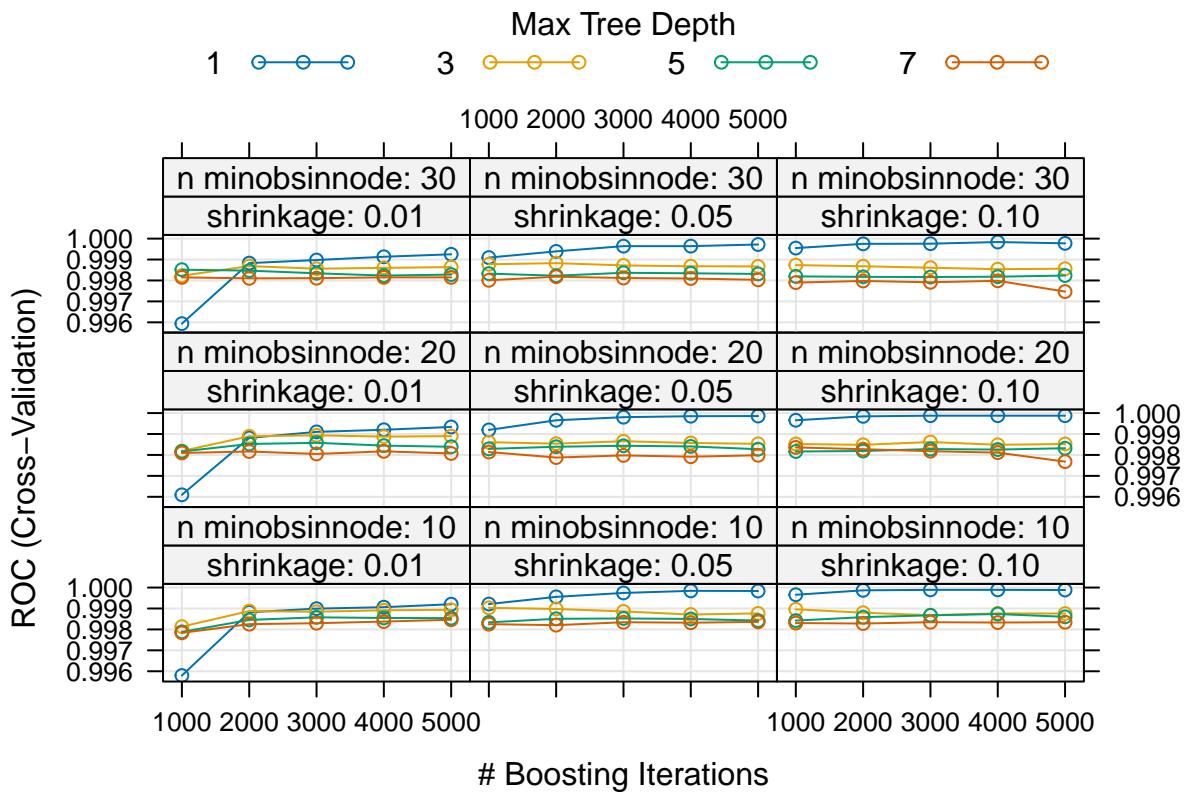
Parameter berikutnya, yaitu “shrinkage”, berfungsi untuk mengatur seberapa besar langkah pembelajaran setiap kali model menambahkan pohon baru. Semakin kecil Nilai yang lebih kecil nilainya, semakin hati-hati pembelajaran model ini, sehingga lebih tahan terhadap overfitting, meskipun memerlukan lebih banyak iterasi.

Terakhir, parameter “n.minobsinnode” digunakan untuk mengatur jumlah minimal data yang harus ada dalam satu node agar pohon dapat bercabang lagi. Hal ini membantu model menghindari pembuatan cabang dengan observasi yang terlalu sedikit, sehingga hasilnya lebih stabil dan tidak terlalu spesifik terhadap data latih.

```
# Tuning model GBM dengan parameter n.trees, interaction.depth, shrinkage, dan n.minobsinnode
set.seed(123)

tuned_gbm <- train(
  Premium_Class ~ .,
  data = train_data,
  method = "gbm",
  metric = "ROC",
  verbose = FALSE,
  tuneGrid = expand.grid(
    n.trees = c(1000, 2000, 3000, 4000, 5000),           # Jumlah pohon
    interaction.depth = c(1, 3, 5, 7),                  # Kedalaman pohon
    shrinkage = c(0.01, 0.05, 0.1),                     # Laju pembelajaran
    n.minobsinnode = c(10, 20, 30)                      # Minimum observasi dalam node
  ),
  trControl = control
)

# Hasil
plot(tuned_gbm)
```



```
best_gbm_model <- tuned_gbm$finalModel
cat("\nBest GBM Model (bestTune):\n")
```

```
##
## Best GBM Model (bestTune):
print(tuned_gbm$bestTune)
```

```
##      n.trees interaction.depth shrinkage n.minobsinnode
## 123      3000                  1       0.1          10
```

Diperoleh bahwa nilai-nilai dari parameter yang menghasilkan nilai AUC tertinggi adalah:

- $n.trees = 2,000$ .
- $interaction.depth = 1$ .
- $shrinkage = 0.1$ .
- $n.minobsinnode = 30$ .

Dengan menggunakan nilai-nilai tersebut, diperoleh nilai AUC-nya adalah 0.9998887. Nilai ini menunjukkan bahwa model GBM dapat sangat baik dalam memprediksi kelas premi, bahkan hampir sempurna. Dengan selisih yang sangat tipis, model GBM memiliki performa yang lebih baik jika dibandingkan dengan model Random Forest. Namun, perlu diperhatikan bahwa model GBM cenderung untuk *overfit*, sehingga ini satu hal yang perlu diwaspadai.

## Rekomendasi untuk Divisi *Marketing*

Berdasarkan hasil evaluasi model, kami merekomendasikan penggunaan Random Forest untuk klasifikasi pemegang polis dengan premi besar dan kecil. Meskipun model GBM menunjukkan performa yang lebih unggul, contohnya nilai AUC yang sangat tinggi, yaitu 0.999, sedangkan Random Forest hanya mencapai 0.9952, perbedaan ini tidak terlalu signifikan, sehingga tidak terlalu berpengaruh. Alasan kami tidak merekomendasikan penggunaan GBM adalah karakteristik dari model ini yang cenderung lebih mudah mengalami *overfitting*, yaitu terlalu menyesuaikan diri dengan data latih sehingga kurang baik saat diterapkan pada data baru. Hal ini dapat membatasi luasnya aplikasi model GBM dalam skenario nyata.

Secara singkat, model GBM sendiri merupakan model yang dibangun secara bertahap dan bertingkat. Dimulai dari model yang sederhana, kemudian secara bertahap menambahkan model-model kecil lainnya guna memperbaiki kesalahan dari model sebelumnya. Setiap model baru difokuskan pada bagian data yang sebelumnya diprediksi dengan kurang tepat. Proses ini terus berlanjut hingga sejumlah langkah tertentu, sehingga model akhir merupakan gabungan dari banyak model kecil yang saling melengkapi. Karena hal ini, GBM dinilai cenderung untuk *overfit* karena model hanya difokuskan dengan data yang dimiliki, di mana bisa saja ada pola-pola yang sebenarnya hanya ada pada data tersebut ditangkap sebagaimana yang umum.

Di sisi lain, model Random Forest dikenal lebih stabil dan umum. Hal ini dikarenakan Random Forest dibangun dari banyak pohon keputusan yang dilatih secara acak, lalu digabungkan melalui pengambilan suara (*voting*). Pendekatan ini membuat Random Forest lebih tahan terhadap risiko *overfitting* dan lebih mampu untuk menghasilkan prediksi dengan baik pada data baru.

Dengan demikian, model Random Forest menjadi pilihan yang lebih dapat dipercaya dan lebih fleksibel untuk mendukung keputusan pemasaran pada kondisi yang beragam.

## Soal 4 - Regresi Linear Berganda, Generalized Linear Model, dan Regularisasi

Pada bagian ini akan dibangun beberapa model untuk memprediksi variabel `Premium_Amount` menggunakan model regresi linear berganda dengan *subset selection*, *stepwise forward*, *stepwise backward*, *Generalized Linear Model* (GLM) serta regularisasi *ridge*, *lasso*, dan *elastic-net*. Kemudian, model-model yang telah dibangun akan dibandingkan dengan berbagai metrik pengukuran *error* untuk menentukan model terbaik untuk memprediksi `Premium_Amount`.

### Memuat Data

```
data = read.csv("data-UTS.csv", sep=";")  
data = na.omit(data)
```

### Menghapus Data Pencilan

Bagian ini data-data `Premium_Amount` yang berupa outlier akan dihapus dari data set. Penghapusan ini bertujuan agar model regresi linear yang dibangun tidak terpengaruh oleh kasus-kasus ekstrim yang ada pada data *outlier*.

### Merubah Data Kategorik menjadi Faktor

Setelah menghilangkan data *outlier*, selanjutnya setiap kolom kategorik dalam data akan diubah menjadi sebuah faktor. Perubahan ini bertujuan agar model di R dapat mengenali dan memperlakukan variabel

tersebut sebagai data kategorik, bukan sebagai data numerik atau karakter biasa. Dengan menjadikan kolom sebagai faktor, R dapat

- Mengelompokkan data berdasarkan kategori secara otomatis,
- Menghitung statistik per kategori dengan benar,

```
## 'data.frame': 9971 obs. of 27 variables:
## $ Age : int 47 37 49 62 36 36 63 51 32 48 ...
## $ Is_Senior : int 0 0 0 1 0 0 1 0 0 0 ...
## $ Marital_Status : Factor w/ 4 levels "Divorced","Married",...: 2 2 2 2 3 2 2 3 2 ...
## $ Married_Premium_Discount : int 86 86 86 86 0 86 86 0 86 0 ...
## $ Prior_Insurance : Factor w/ 3 levels "<1 year",">5 years",...: 3 3 3 2 2 2 3 1 2 ...
## $ Prior_Insurance_Premium_Adjustment: int 50 50 50 0 0 0 50 100 0 0 ...
## $ Claims_Frequency : int 0 0 1 1 2 0 0 0 0 1 ...
## $ Claims_Severity : Factor w/ 3 levels "High","Low","Medium": 2 2 2 2 2 3 2 2 2 1 ...
## $ Claims_Adjustment : int 0 0 50 50 100 0 0 0 0 200 ...
## $ Policy_Type : Factor w/ 2 levels "Full Coverage",...: 1 1 1 1 1 2 1 1 2 1 ...
## $ Policy_Adjustment : int 0 0 0 0 -200 0 0 -200 0 ...
## $ Premium_Amount : int 2286 2336 2386 2336 2350 1936 2286 2300 1936 2350 ...
## $ Safe_Driver_Discount : int 0 0 0 0 0 0 1 0 0 1 ...
## $ Multi_Policy_Discount : int 0 0 0 0 0 0 0 1 1 0 ...
## $ Bundling_Discount : int 0 0 0 0 0 1 0 0 0 0 ...
## $ Total_Discounts : int 0 0 0 0 0 50 50 50 50 50 ...
## $ Source_of_Lead : Factor w/ 3 levels "Agent","Online",...: 1 2 2 2 1 2 2 2 2 2 ...
## $ Time_Since_First_Contact : int 10 22 28 4 14 13 2 1 16 27 ...
## $ Conversion_Status : int 0 0 0 1 1 1 0 1 0 ...
## $ Website_Visits : int 5 5 4 6 8 4 5 3 5 5 ...
## $ Inquiries : int 1 1 4 2 4 1 1 0 1 3 ...
## $ Quotes_Requested : int 2 2 1 2 2 1 2 2 3 2 ...
## $ Time_to_Conversion : int 99 99 99 2 10 7 1 99 3 99 ...
## $ Credit_Score : int 704 726 772 809 662 729 795 639 724 710 ...
## $ Premium_Adjustment_Credit : int -50 -50 -50 -50 -50 -50 -50 -50 -50 ...
## $ Region : Factor w/ 3 levels "Rural","Suburban",...: 2 3 3 3 2 1 3 2 1 3 ...
## $ Premium_Adjustment_Region : int 50 100 100 100 50 0 100 50 0 100 ...

##      Age     Is_Senior   Marital_Status Married_Premium_Discount
## Min.   :18   Min.   :0.0000 Divorced: 917   Min.   : 0.0
## 1st Qu.:29  1st Qu.:0.0000 Married :4881  1st Qu.: 0.0
## Median :39  Median :0.0000 Single  :3252  Median : 0.0
## Mean   :40  Mean   :0.1594 Widowed : 921   Mean   :42.1
## 3rd Qu.:50  3rd Qu.:0.0000                  3rd Qu.:86.0
## Max.   :90  Max.   :1.0000                  Max.   :86.0
## 
##      Prior_Insurance Prior_Insurance_Premium_Adjustment Claims_Frequency
## <1 year    :2121      Min.   : 0.00          Min.   :0.000
## >5 years   :2607      1st Qu.: 0.00          1st Qu.:0.000
## 1-5 years  :5243      Median : 50.00          Median :0.000
##                   Mean   : 47.56          Mean   :0.491
##                   3rd Qu.: 50.00          3rd Qu.:1.000
##                   Max.   :100.00          Max.   :5.000
## 
##      Claims_Severity Claims_Adjustment Policy_Type Policy_Adjustment
## High   : 933       Min.   : 0.00 Full Coverage :5982   Min.   :-200.00
## Low    :7003        1st Qu.: 0.00 Liability-Only:3989  1st Qu.:-200.00
## Medium:2035       Median : 0.00                  Median : 0.00
```

```

##          Mean   : 35.46           Mean   : -80.01
##          3rd Qu.: 50.00           3rd Qu.:  0.00
##          Max.   :600.00           Max.   :  0.00
## Premium_Amount Safe_Driver_Discount Multi_Policy_Discount Bundling_Discount
## Min.   :1800    Min.   :0.0          Min.   :0.0000    Min.   :0.00000
## 1st Qu.:2100    1st Qu.:0.0         1st Qu.:0.0000    1st Qu.:0.00000
## Median :2236    Median :0.0         Median :0.0000    Median :0.00000
## Mean   :2218    Mean   :0.2         Mean   :0.3053    Mean   :0.09728
## 3rd Qu.:2336    3rd Qu.:0.0         3rd Qu.:1.0000    3rd Qu.:0.00000
## Max.   :2686    Max.   :1.0         Max.   :1.0000    Max.   :1.00000
## Total_Discounts Source_of_Lead Time_Since_First_Contact Conversion_Status
## Min.   : 0.00    Agent   :2996    Min.   : 1.00     Min.   :0.0000
## 1st Qu.: 0.00    Online   :6015    1st Qu.: 8.00     1st Qu.:0.0000
## Median : 50.00    Referral: 960    Median :15.00     Median :1.0000
## Mean   : 30.13                Mean   :15.47     Mean   :0.5762
## 3rd Qu.: 50.00                3rd Qu.:23.00    3rd Qu.:1.0000
## Max.   :150.00                Max.   :30.00     Max.   :1.0000
## Website_Visits   Inquiries   Quotes_Requested Time_to_Conversion
## Min.   : 0.000    Min.   :0.000    Min.   :1.000    Min.   : 1.00
## 1st Qu.: 3.000    1st Qu.:1.000    1st Qu.:1.000    1st Qu.: 6.00
## Median : 5.000    Median :2.000    Median :2.000    Median :12.00
## Mean   : 5.021    Mean   :1.997    Mean   :1.998    Mean   :46.12
## 3rd Qu.: 6.000    3rd Qu.:3.000    3rd Qu.:3.000    3rd Qu.:99.00
## Max.   :16.000    Max.   :9.000    Max.   :3.000    Max.   :99.00
## Credit_Score     Premium_Adjustment_Credit Region
## Min.   :530.0    Min.   :-50.00    Rural   :2053
## 1st Qu.:681.0    1st Qu.:-50.00    Suburban:3016
## Median :715.0    Median :-50.00    Urban   :4902
## Mean   :714.3    Mean   :-11.37
## 3rd Qu.:748.0    3rd Qu.: 50.00
## Max.   :850.0    Max.   : 50.00
## Premium_Adjustment_Region
## Min.   : 0.00
## 1st Qu.: 50.00
## Median : 50.00
## Mean   : 64.29
## 3rd Qu.:100.00
## Max.   :100.00

```

## Data Latih dan Data Uji

Pada bagian ini, data akan dibagi menjadi data latih dan data uji. Pembagian data akan mengikuti komposisi 80% untuk data latih dan 20% untuk data uji. Hal ini dilakukan agar model regresi memiliki cukup banyak data untuk mempelajari pola-pola yang terdapat dalam data. Dengan proporsi 80% untuk pelatihan, model memiliki informasi yang memadai untuk mengenali hubungan antara variabel prediktor dan variabel target. Sementara itu, 20% sisanya disisihkan sebagai data uji agar model dapat dievaluasi secara objektif terhadap data yang belum pernah dilihat sebelumnya. Hal ini penting untuk menilai kemampuan generalisasi model dalam memprediksi data baru, serta menghindari overfitting.

```

set.seed(888)
train_index = sample(1:nrow(data_clean), 0.8*nrow(data_clean))
train = data_clean[train_index,]
test = data_clean[-train_index,]

```

## Subset Selection

Pada tahap ini, akan digunakan fungsi `subset_selection` dari pustaka `leaps` untuk membangun model yang memprediksi variabel `Premium_Amount` berdasarkan variabel-variabel prediktor yang tersedia dalam data. Jumlah maksimum variabel bebas yang dapat digunakan dalam satu model adalah 27, yang dalam konteks ini berarti model yang mencakup seluruh variabel prediktor dalam data. Setelah berbagai model dibentuk, akan dipilih satu model terbaik berdasarkan kinerjanya terhadap data uji. Model terbaik tersebut kemudian akan dievaluasi lebih lanjut untuk memastikan apakah ia memenuhi asumsi-asumsi dasar dari regresi linear.

```
regfit_full = regsubsets(Premium_Amount ~ ., data=train, nvmax=27)

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 5 linear dependencies found

## Reordering variables and trying again:

set.seed(888)
reg_summary_subset = summary(regfit_full)
print(reg_summary_subset)

## Subset selection object
## Call: regsubsets.formula(Premium_Amount ~ ., data = train, nvmax = 27)
## 32 Variables  (and intercept)
##                                         Forced in    Forced out
## Age                               FALSE      FALSE
## Is_Senior                          FALSE      FALSE
## Marital_StatusMarried              FALSE      FALSE
## Marital_StatusSingle               FALSE      FALSE
## Marital_StatusWidowed              FALSE      FALSE
## Prior_Insurance>5 years            FALSE      FALSE
## Prior_Insurance1-5 years           FALSE      FALSE
## Claims_Frequency                  FALSE      FALSE
## Claims_SeverityLow                FALSE      FALSE
## Claims_SeverityMedium              FALSE      FALSE
## Claims_Adjustment                 FALSE      FALSE
## Policy_TypeLiability-Only         FALSE      FALSE
## Safe_Driver_Discount              FALSE      FALSE
## Multi_Policy_Discount             FALSE      FALSE
## Bundling_Discount                 FALSE      FALSE
## Source_of_LeadOnline               FALSE      FALSE
## Source_of_LeadReferral             FALSE      FALSE
## Time_Since_First_Contact          FALSE      FALSE
## Conversion_Status                 FALSE      FALSE
## Website_Visits                   FALSE      FALSE
## Inquiries                          FALSE      FALSE
## Quotes_Requested                  FALSE      FALSE
## Time_to_Conversion                FALSE      FALSE
## Credit_Score                       FALSE      FALSE
## Premium_Adjustment_Credit          FALSE      FALSE
## RegionSuburban                    FALSE      FALSE
## RegionUrban                        FALSE      FALSE
```

```

## Married_Premium_Discount           FALSE  FALSE
## Prior_Insurance_Premium_Adjustment FALSE  FALSE
## Policy_Adjustment                 FALSE  FALSE
## Total_Discounts                  FALSE  FALSE
## Premium_Adjustment_Region        FALSE  FALSE
## 1 subsets of each size up to 27
## Selection Algorithm: exhaustive
##          Age Is_Senior Marital_StatusMarried Marital_StatusSingle
## 1  ( 1 ) " " " "      " "          " "
## 2  ( 1 ) " " " "      " "          " "
## 3  ( 1 ) " " " "      " "          " "
## 4  ( 1 ) " " " "      "*"         " "
## 5  ( 1 ) " " " "      "*"         " "
## 6  ( 1 ) " " " "      "*"         " "
## 7  ( 1 ) " " " "      "*"         " "
## 8  ( 1 ) "*" " "      "*"         " "
## 9  ( 1 ) "*" "*"      "*"         " "
## 10 ( 1 ) "*" " "      "*"         " "
## 11 ( 1 ) "*" "*"      "*"         " "
## 12 ( 1 ) "*" "*"      "*"         "*"
## 13 ( 1 ) "*" "*"      "*"         "*"
## 14 ( 1 ) "*" "*"      "*"         "*"
## 15 ( 1 ) "*" "*"      "*"         "*"
## 16 ( 1 ) "*" "*"      "*"         "*"
## 17 ( 1 ) "*" "*"      "*"         "*"
## 18 ( 1 ) "*" "*"      "*"         "*"
## 19 ( 1 ) "*" "*"      "*"         "*"
## 20 ( 1 ) "*" "*"      " "        " "
## 21 ( 1 ) "*" "*"      " "        " "
## 22 ( 1 ) "*" "*"      "*"         "*"
## 23 ( 1 ) "*" "*"      "*"         "*"
## 24 ( 1 ) "*" "*"      " "        " "
## 25 ( 1 ) "*" "*"      "*"         "*"
## 26 ( 1 ) "*" "*"      "*"         "*"
## 27 ( 1 ) "*" "*"      "*"         "*"
##          Marital_StatusWidowed Married_Premium_Discount
## 1  ( 1 ) " "          " "        "
## 2  ( 1 ) " "          " "        "
## 3  ( 1 ) " "          " "        "
## 4  ( 1 ) " "          " "        "
## 5  ( 1 ) " "          " "        "
## 6  ( 1 ) " "          " "        "
## 7  ( 1 ) " "          " "        "
## 8  ( 1 ) " "          " "        "
## 9  ( 1 ) " "          " "        "
## 10 ( 1 ) " "          " "        "
## 11 ( 1 ) " "          " "        "
## 12 ( 1 ) "*"          " "        "
## 13 ( 1 ) "*"          " "        "
## 14 ( 1 ) "*"          " "        "
## 15 ( 1 ) "*"          " "        "
## 16 ( 1 ) "*"          " "        "
## 17 ( 1 ) "*"          " "        "
## 18 ( 1 ) "*"          " "        "

```

```

## 19 ( 1 ) "*"      ""
## 20 ( 1 ) "*"      "*"
## 21 ( 1 ) "*"      "*"
## 22 ( 1 ) "*"      ""
## 23 ( 1 ) "*"      ""
## 24 ( 1 ) "*"      "*"
## 25 ( 1 ) "*"      "*"
## 26 ( 1 ) "*"      ""
## 27 ( 1 ) "*"      ""

##          Prior_Insurance>5 years Prior_Insurance1-5 years

## 1 ( 1 ) ""      ""
## 2 ( 1 ) ""      ""
## 3 ( 1 ) ""      ""
## 4 ( 1 ) ""      ""
## 5 ( 1 ) ""      ""
## 6 ( 1 ) ""      ""
## 7 ( 1 ) ""      ""
## 8 ( 1 ) ""      ""
## 9 ( 1 ) ""      ""
## 10 ( 1 ) ""     ""
## 11 ( 1 ) ""     ""
## 12 ( 1 ) "*"     "*"
## 13 ( 1 ) "*"     "*"
## 14 ( 1 ) "*"     "*"
## 15 ( 1 ) "*"     ""
## 16 ( 1 ) "*"     ""
## 17 ( 1 ) "*"     ""
## 18 ( 1 ) "*"     ""
## 19 ( 1 ) ""     "*"
## 20 ( 1 ) "*"     ""
## 21 ( 1 ) "*"     "*"
## 22 ( 1 ) "*"     ""
## 23 ( 1 ) "*"     ""
## 24 ( 1 ) "*"     "*"
## 25 ( 1 ) ""     "*"
## 26 ( 1 ) ""     "*"
## 27 ( 1 ) "*"     ""

##          Prior_Insurance_Premium_Adjustment Claims_Frequency

## 1 ( 1 ) ""      ""
## 2 ( 1 ) ""      ""
## 3 ( 1 ) ""      ""
## 4 ( 1 ) ""      ""
## 5 ( 1 ) ""      ""
## 6 ( 1 ) "*"     ""
## 7 ( 1 ) "*"     ""
## 8 ( 1 ) "*"     ""
## 9 ( 1 ) "*"     ""
## 10 ( 1 ) "*"    "*"
## 11 ( 1 ) "*"    "*"
## 12 ( 1 ) ""      ""
## 13 ( 1 ) ""      "*"
## 14 ( 1 ) ""      "*"
## 15 ( 1 ) "*"     "*"
## 16 ( 1 ) "*"     "*"

```

```

## 17 ( 1 ) "*"      "*"
## 18 ( 1 ) "*"      "*"
## 19 ( 1 ) "*"      "*"
## 20 ( 1 ) "*"      "*"
## 21 ( 1 ) " "     "*"
## 22 ( 1 ) "*"      "*"
## 23 ( 1 ) "*"      "*"
## 24 ( 1 ) " "     "*"
## 25 ( 1 ) "*"      "*"
## 26 ( 1 ) "*"      "*"
## 27 ( 1 ) " "     "*"

##          Claims_SeverityLow Claims_SeverityMedium Claims_Adjustment
## 1 ( 1 ) " "           " "           " "
## 2 ( 1 ) " "           " "           "*"
## 3 ( 1 ) " "           " "           "*"
## 4 ( 1 ) " "           " "           "*"
## 5 ( 1 ) " "           " "           "*"
## 6 ( 1 ) " "           " "           "*"
## 7 ( 1 ) " "           " "           "*"
## 8 ( 1 ) " "           " "           "*"
## 9 ( 1 ) " "           " "           "*"
## 10 ( 1 ) " "          " "           "*"
## 11 ( 1 ) " "          " "           "*"
## 12 ( 1 ) " "          " "           "*"
## 13 ( 1 ) " "          " "           "*"
## 14 ( 1 ) "*"          " "           "*"
## 15 ( 1 ) "*"          " "           "*"
## 16 ( 1 ) "*"          " "           "*"
## 17 ( 1 ) "*"          " "           "*"
## 18 ( 1 ) "*"          " "           "*"
## 19 ( 1 ) "*"          " "           "*"
## 20 ( 1 ) "*"          " "           "*"
## 21 ( 1 ) "*"          " "           "*"
## 22 ( 1 ) "*"          " "           "*"
## 23 ( 1 ) "*"          " "           "*"
## 24 ( 1 ) "*"          " "           "*"
## 25 ( 1 ) "*"          " "           "*"
## 26 ( 1 ) "*"          " "           "*"
## 27 ( 1 ) "*"          " "           "*"

##          Policy_TypeLiability-Only Policy_Adjustment Safe_Driver_Discount
## 1 ( 1 ) "*"          " "           " "
## 2 ( 1 ) "*"          " "           " "
## 3 ( 1 ) "*"          " "           " "
## 4 ( 1 ) "*"          " "           " "
## 5 ( 1 ) " "          "*"          " "
## 6 ( 1 ) " "          "*"          " "
## 7 ( 1 ) "*"          " "           " "
## 8 ( 1 ) " "          "*"          " "
## 9 ( 1 ) " "          "*"          " "
## 10 ( 1 ) " "         "*"          " "
## 11 ( 1 ) " "         "*"          " "
## 12 ( 1 ) " "         "*"          " "
## 13 ( 1 ) "*"          " "           " "
## 14 ( 1 ) "*"          " "           " "

```

```

## 15 ( 1 ) " "      "*"      " "
## 16 ( 1 ) " "      "*"      " "
## 17 ( 1 ) " "      "*"      " "
## 18 ( 1 ) " "      "*"      " "
## 19 ( 1 ) " "      "*"      " "
## 20 ( 1 ) " "      "*"      " "
## 21 ( 1 ) "*"      " "      "*"
## 22 ( 1 ) "*"      " "      "*"
## 23 ( 1 ) " "      "*"      " "
## 24 ( 1 ) "*"      " "      "*"
## 25 ( 1 ) "*"      " "      " "
## 26 ( 1 ) " "      "*"      "*"
## 27 ( 1 ) "*"      " "      "*"
##          Multi_Policy_Discount Bundling_Discount Total_Discounts
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
## 4 ( 1 ) " "      " "      " "
## 5 ( 1 ) " "      " "      " "
## 6 ( 1 ) " "      " "      " "
## 7 ( 1 ) " "      " "      "*"
## 8 ( 1 ) " "      " "      "*"
## 9 ( 1 ) " "      " "      "*"
## 10 ( 1 ) " "     " "      "*"
## 11 ( 1 ) " "     " "      "*"
## 12 ( 1 ) " "     " "      "*"
## 13 ( 1 ) " "     " "      "*"
## 14 ( 1 ) " "     " "      "*"
## 15 ( 1 ) " "     " "      "*"
## 16 ( 1 ) " "     " "      "*"
## 17 ( 1 ) " "     " "      "*"
## 18 ( 1 ) " "     " "      "*"
## 19 ( 1 ) "*"     " "      "*"
## 20 ( 1 ) "*"     " "      "*"
## 21 ( 1 ) "*"     " "      "*"
## 22 ( 1 ) " "     " "      "*"
## 23 ( 1 ) " "     " "      "*"
## 24 ( 1 ) "*"     " "      " "
## 25 ( 1 ) "*"     " "      "*"
## 26 ( 1 ) "*"     " "      " "
## 27 ( 1 ) "*"     " "      " "
##          Source_of_LeadOnline Source_of_LeadReferral Time_Since_First_Contact
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
## 4 ( 1 ) " "      " "      " "
## 5 ( 1 ) " "      " "      " "
## 6 ( 1 ) " "      " "      " "
## 7 ( 1 ) " "      " "      " "
## 8 ( 1 ) " "      " "      " "
## 9 ( 1 ) " "      " "      " "
## 10 ( 1 ) " "     " "      " "
## 11 ( 1 ) " "     " "      " "
## 12 ( 1 ) " "     " "      " "

```

```

## 13 ( 1 ) " " " "
## 14 ( 1 ) " " " "
## 15 ( 1 ) " " " "
## 16 ( 1 ) " " " *"
## 17 ( 1 ) " " " *"
## 18 ( 1 ) " " " *"
## 19 ( 1 ) " " " *"
## 20 ( 1 ) " " " *"
## 21 ( 1 ) " " " *"
## 22 ( 1 ) "*" " *"
## 23 ( 1 ) "*" " *"
## 24 ( 1 ) "*" " *"
## 25 ( 1 ) "*" " *"
## 26 ( 1 ) "*" " *"
## 27 ( 1 ) "*" " *"
## Conversion_Status Website_Visits Inquiries Quotes_Requested
## 1 ( 1 ) " " " " "
## 2 ( 1 ) " " " " "
## 3 ( 1 ) " " " " "
## 4 ( 1 ) " " " " "
## 5 ( 1 ) " " " " "
## 6 ( 1 ) " " " " "
## 7 ( 1 ) " " " " "
## 8 ( 1 ) " " " " "
## 9 ( 1 ) " " " " "
## 10 ( 1 ) " " " " "
## 11 ( 1 ) " " " " "
## 12 ( 1 ) " " " " "
## 13 ( 1 ) " " " " "
## 14 ( 1 ) " " " " "
## 15 ( 1 ) " " " " "
## 16 ( 1 ) " " " " "
## 17 ( 1 ) " " " *"
## 18 ( 1 ) "*" " "
## 19 ( 1 ) "*" " "
## 20 ( 1 ) "*" " "
## 21 ( 1 ) "*" " "
## 22 ( 1 ) "*" " *"
## 23 ( 1 ) "*" " *"
## 24 ( 1 ) "*" " *"
## 25 ( 1 ) "*" " *"
## 26 ( 1 ) "*" " *"
## 27 ( 1 ) "*" " *"
## Time_to_Conversion Credit_Score Premium_Adjustment_Credit
## 1 ( 1 ) " " " "
## 2 ( 1 ) " " " "
## 3 ( 1 ) " " " *"
## 4 ( 1 ) " " " *"
## 5 ( 1 ) " " " *"
## 6 ( 1 ) " " " *"
## 7 ( 1 ) " " " *"
## 8 ( 1 ) " " " *"
## 9 ( 1 ) " " " *"
## 10 ( 1 ) " " " *"

```

```

## 11 ( 1 ) " "      "*"      "*"
## 12 ( 1 ) " "      " "      "*"
## 13 ( 1 ) " "      " "      "*"
## 14 ( 1 ) " "      " "      "*"
## 15 ( 1 ) " "      "*"      "*"
## 16 ( 1 ) " "      "*"      "*"
## 17 ( 1 ) " "      "*"      "*"
## 18 ( 1 ) "*"      "*"      "*"
## 19 ( 1 ) "*"      "*"      "*"
## 20 ( 1 ) "*"      "*"      "*"
## 21 ( 1 ) "*"      "*"      "*"
## 22 ( 1 ) "*"      "*"      "*"
## 23 ( 1 ) "*"      "*"      "*"
## 24 ( 1 ) "*"      "*"      "*"
## 25 ( 1 ) "*"      "*"      "*"
## 26 ( 1 ) "*"      "*"      "*"
## 27 ( 1 ) "*"      "*"      "*"

##          RegionSuburban RegionUrban Premium_Adjustment_Region
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
## 4 ( 1 ) " "      " "      " "
## 5 ( 1 ) " "      " "      "*"
## 6 ( 1 ) " "      " "      "*"
## 7 ( 1 ) " "      " "      "*"
## 8 ( 1 ) " "      " "      "*"
## 9 ( 1 ) " "      " "      "*"
## 10 ( 1 ) " "      " "      "*"
## 11 ( 1 ) " "      " "      "*"
## 12 ( 1 ) " "      " "      "*"
## 13 ( 1 ) " "      " "      "*"
## 14 ( 1 ) " "      " "      "*"
## 15 ( 1 ) " "      " "      "*"
## 16 ( 1 ) " "      " "      "*"
## 17 ( 1 ) " "      " "      "*"
## 18 ( 1 ) " "      " "      "*"
## 19 ( 1 ) " "      " "      "*"
## 20 ( 1 ) "*"      "*"      " "
## 21 ( 1 ) "*"      " "      "*"
## 22 ( 1 ) "*"      " "      "*"
## 23 ( 1 ) " "      "*"      "*"
## 24 ( 1 ) " "      "*"      "*"
## 25 ( 1 ) " "      " "      "*"
## 26 ( 1 ) "*"      "*"      " "
## 27 ( 1 ) "*"      "*"      " "

```

Tabel di atas menunjukkan hasil dari proses *subset selection* menggunakan fungsi `regsubsets` untuk memodelkan variabel respons `Premium_Amount` berdasarkan seluruh variabel bebas yang tersedia dalam data. Prosedur ini dilakukan secara ekshhaustif, yaitu dengan mengevaluasi seluruh kemungkinan kombinasi variabel hingga maksimum 27 prediktor.

Setiap baris pada tabel merepresentasikan sebuah model dengan jumlah variabel bebas tertentu, mulai dari model dengan satu variabel hingga model dengan 27 variabel. Simbol "\*" menandakan bahwa variabel tersebut disertakan dalam model pada baris tersebut. Selanjutnya akan ditampilkan metrik-metrik untuk mengukur performa dari semua model yang telah dibentuk oleh proses *subset-selection*.

```

for (metric in names(reg_summary_subset)) {
  if (metric != "which" && metric != "outmat" && metric != "obj") {
    cat(metric, ":\n")
    print(reg_summary_subset[[metric]])
    cat("\n")
  }
}

## rsq :
## [1] 0.4506927 0.6195438 0.7303598 0.8153171 0.8899720 0.9463980 1.0000000
## [8] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [15] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [22] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##
## rss :
## [1] 9.247868e+07 6.405173e+07 4.539530e+07 3.109230e+07 1.852377e+07
## [6] 9.024170e+06 1.345852e-14 1.827893e-18 1.140510e-18 9.082140e-19
## [11] 7.452662e-19 7.046565e-19 6.954540e-19 6.926713e-19 6.919641e-19
## [16] 6.914491e-19 6.914026e-19 6.909995e-19 6.909370e-19 6.908438e-19
## [21] 6.907743e-19 6.907238e-19 6.907078e-19 6.907005e-19 6.906957e-19
## [26] 6.906953e-19 0.000000e+00
##
## adjr2 :
## [1] 0.4506238 0.6194484 0.7302583 0.8152245 0.8899030 0.9463576 1.0000000
## [8] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [15] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [22] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##
## cp :
## [1] 1.063505e+30 7.365953e+29 5.220463e+29 3.575616e+29 2.130235e+29
## [6] 1.037780e+29 1.547651e+08 1.306279e+04 5.159873e+03 2.490467e+03
## [11] 6.185659e+02 1.535546e+02 4.972551e+01 1.972479e+01 1.359207e+01
## [16] 9.669245e+00 1.113431e+01 8.498298e+00 9.779589e+00 1.070810e+01
## [21] 1.190923e+01 1.332825e+01 1.514427e+01 1.706014e+01 1.900532e+01
## [26] 2.100013e+01 -7.920000e+03
##
## bic :
## [1] -4760.431 -7680.929 -10417.941 -13427.462 -17549.291 -23276.233
## [7] -405752.988 -476763.930 -480517.135 -482324.691 -483892.870 -484330.788
## [13] -484426.654 -484449.647 -484448.810 -484445.765 -484437.317 -484432.985
## [19] -484424.722 -484416.814 -484408.632 -484400.231 -484391.431 -484382.532
## [25] -484373.602 -484364.623 -Inf

```

Data di atas menunjukkan nilai-nilai metrik performa dari 27 model yang dihasilkan melalui proses *subset selection*, yaitu metode pemilihan subset terbaik dari sekumpulan variabel prediktor. Setiap model memiliki jumlah prediktor yang berbeda, mulai dari satu hingga semua variabel yang tersedia. Lima metrik evaluasi digunakan untuk menilai performa model, yaitu: *R-squared* (**rsq**), *Residual Sum of Squares* (**rss**), *Adjusted R-squared* (**adjr2**), *Mallow's Cp* (**cp**), dan *Bayesian Information Criterion* (**bic**). Masing-masing metrik terdiri dari 27 angka yang mencerminkan performa model ke-1 hingga ke-27.

- **R-squared** (**rsq**) mengukur proporsi variasi dalam data respons yang dapat dijelaskan oleh prediktor dalam model. Nilainya meningkat seiring penambahan prediktor dan mencapai 1 pada model ke-7 dan seterusnya, yang mengindikasikan bahwa model mampu menjelaskan seluruh variasi data.

- **Residual Sum of Squares (rss)** menunjukkan total kesalahan prediksi model. Nilainya menurun drastis dari model pertama hingga ketujuh, yang artinya penambahan prediktor mengurangi error. Setelah model ke-7, nilai RSS mendekati nol, bahkan menjadi nol pada model ke-27.
- **Adjusted R-squared (adjr2)** merupakan pengembangan dari  $R^2$  yang memberikan penalti untuk penambahan variabel yang tidak memberi kontribusi berarti terhadap model. Nilai adjr2 meningkat hingga model ke-6 atau ke-7, lalu menetap di angka 1, yang menandakan bahwa model setelah titik ini tidak memberikan peningkatan dalam kualitas prediksi, meskipun model bertambah kompleks.
- **Mallow's Cp (cp)** digunakan untuk menilai keseimbangan antara bias dan varians dalam model. Ide-alnya, nilai Cp mendekati jumlah parameter dalam model ditambah 1. Dalam data ini, Cp mengalami penurunan ekstrem dari nilai sangat besar ke nilai yang lebih stabil mulai sekitar model ke-7. Nilai Cp yang paling mendekati jumlah prediktor ditemukan pada model ke-16.
- **Bayesian Information Criterion (bic)** adalah metrik penalti kompleksitas model, di mana nilai yang lebih kecil (lebih negatif) menunjukkan model yang lebih baik. Nilai BIC turun secara konsisten hingga model ke-26, lalu menjadi negatif tak terhingga ( $-\infty$ ) pada model ke-27. Hal ini kemungkinan besar terjadi karena RSS bernilai nol yang dapat berarti bahwa model *overfit*.

Selanjutnya akan ditampilkan model-model terbaik berdasarkan metrik performa *Adjusted R<sup>2</sup>*, *Mallow's Cp*, dan BIC. Alasan kami memilih metrik performa ini adalah metrik performa tersebut memberikan penalti kepada model yang terlalu kompleks dan mengoptimalkan error dan jumlah variabel dalam model.

```
set.seed(888)
cp <- reg_summary_subset$cp
bic <- reg_summary_subset$bic
adjr2 <- reg_summary_subset$adjr2

# Hapus nilai -Inf di BIC (hanya untuk BIC)
bic_valid <- bic[is.finite(bic)] # Hapus nilai Inf

num_predictors <- 1:length(cp)

par(mfrow = c(1, 3), oma = c(0, 0, 2, 0), mar = c(4, 4, 2, 1))

# Plot Cp
cp_diff <- abs(cp - (num_predictors + 1))
best_cp <- which.min(cp_diff)
plot(num_predictors, cp, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = expression(C[p]),
     main = "Cp Plot", cex.axis = 0.9, cex.lab = 1.1)
points(best_cp, cp[best_cp], col = "red", cex = 2, pch = 4)
text(best_cp, cp[best_cp], labels = paste("Best:", best_cp), pos = 4, col = "red", cex = 0.9)

# Plot BIC
num_predictors_bic <- 1:length(bic_valid)
plot(num_predictors_bic, bic_valid, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = "BIC", main = "BIC Plot",
     cex.axis = 0.9, cex.lab = 1.1)
best_bic <- which.min(bic_valid)
points(best_bic, bic_valid[best_bic], col = "red", cex = 2, pch = 4)
text(best_bic, bic_valid[best_bic], labels = paste("Best:", best_bic), pos = 4, col = "red", cex = 0.9)

# Plot Adjusted R2
```

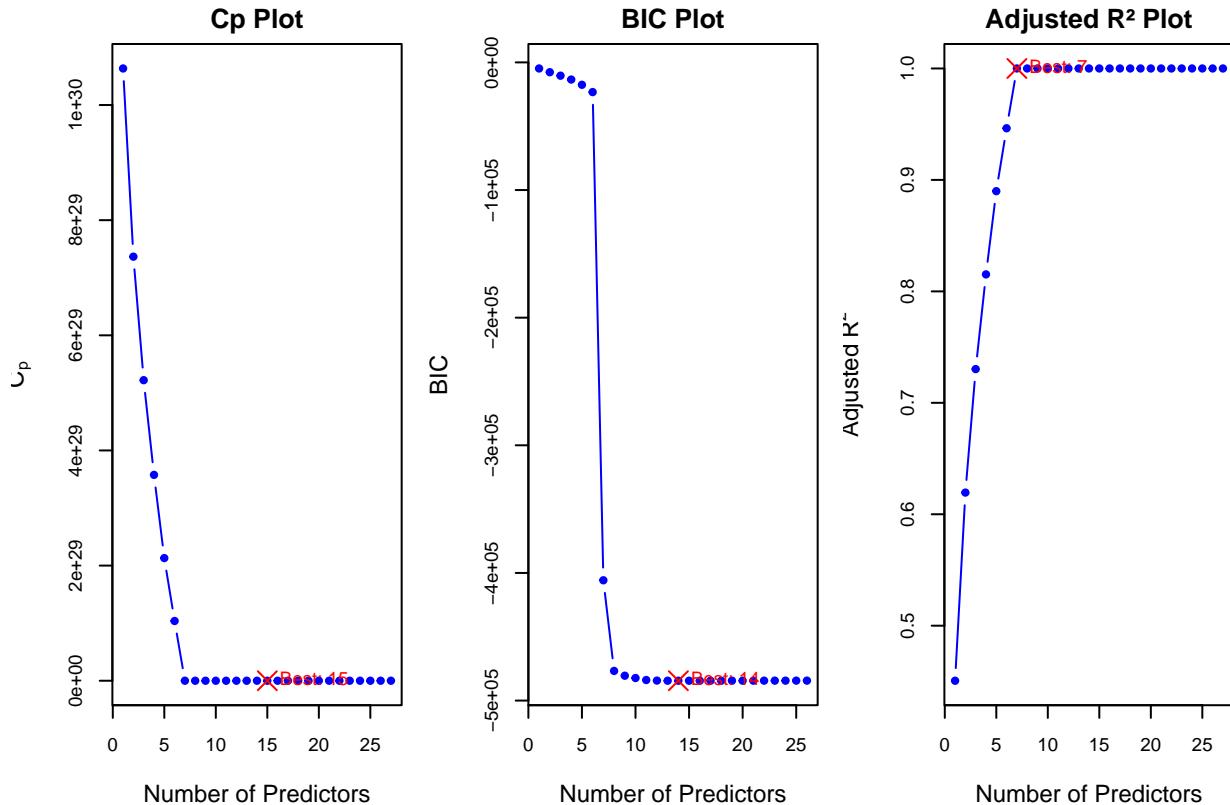
```

plot(num_predictors, adjr2, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = expression(Adjusted~R^2), main = "Adjusted R2 Plot",
     cex.axis = 0.9, cex.lab = 1.1)
best_adjr2 <- which.max(adjr2)
points(best_adjr2, adjr2[best_adjr2], col = "red", cex = 2, pch = 4)
text(best_adjr2, adjr2[best_adjr2], labels = paste("Best:", best_adjr2), pos = 4, col = "red", cex = 0.9)

# Add main title
mtext("Best Subset Selection Model", side = 3, line = 0, cex = 1.5, outer = TRUE)

```

## Best Subset Selection Model



Gambar di atas menunjukkan bahwa model terbaik dengan menggunakan *Best Subset Selection* berdasarkan parameter *Mallow's Cp* adalah model dengan 15 variabel, sedangkan untuk metrik BIC model terbaik diberikan oleh model dengan 14 variabel, dan dengan metrik *Adjusted R<sup>2</sup>* diberikan oleh model dengan 7 variabel. Dari hasil-hasil di atas dapat disimpulkan bahwa model terbaik dengan metode *Best Subset Selection* adalah model dengan 15 variabel. Hal ini disebabkan bahwa model dengan 15 variabel berdasarkan memberikan performa terbaik pada *Mallow's Cp*, performa yang setara dengan model terbaik pada metrik *Adjusted R<sup>2</sup>*, serta memiliki nilai BIC yang sangat dekat dengan model dengan BIC terbaik.

### Koefesien Model Terbaik *Subset Selection*

```
coef(regfit_full, id=15)
```

##	(Intercept)	Age
----	-------------	-----

```

##                               2046.53740787          0.03720339
##                               Is_Senior           Marital_StatusMarried
##                               1.77649509          87.63615845
##                               Marital_StatusSingle   Marital_StatusWidowed
##                               2.29943418          3.96457267
##                               Prior_Insurance1-5 years   Claims_SeverityLow
##                               -6.01748162          9.28443159
##                               Claims_SeverityMedium   Claims_Adjustment
##                               10.73314554          0.97307261
##                               Safe_Driver_Discount   Bundling_Discount
##                               -49.52834079          -53.67111820
##                               Conversion_Status   Married_Premium_Discount
##                               -18.08334519          0.00000000
## Prior_Insurance_Premium_Adjustment Premium_Adjustment_Region
##                               1.00268680          0.98432599

```

## Stepwise Forward Selection

Pada tahap ini, proses pemodelan dilakukan menggunakan fungsi `subset_selection` dengan metode *forward selection* dari pustaka `leaps`. Tujuannya adalah untuk membangun model regresi yang memprediksi variabel `Premium_Amount` berdasarkan sejumlah variabel prediktor yang tersedia dalam data. Karena alur penggerjaan pada bagian ini serupa dengan langkah-langkah yang telah dijelaskan pada bagian sebelumnya, penjelasan teknis yang sama tidak akan diulang kembali.

```

set.seed(888)
regfit_fwd <- regsubsets(Premium_Amount ~ ., data = train,
nvmax = 27, method = "forward")

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 5 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

summary_fwd = summary(regfit_fwd)

print(summary_fwd)

## Subset selection object
## Call: regsubsets.formula(Premium_Amount ~ ., data = train, nvmax = 27,
##   method = "forward")
## 32 Variables  (and intercept)
##                                         Forced in    Forced out
## Age                               FALSE      FALSE
## Is_Senior                          FALSE      FALSE
## Marital_StatusMarried              FALSE      FALSE
## Marital_StatusSingle               FALSE      FALSE
## Marital_StatusWidowed              FALSE      FALSE
## Prior_Insurance>5 years            FALSE      FALSE

```

```

## Prior_Insurance1-5 years          FALSE    FALSE
## Claims_Frequency                 FALSE    FALSE
## Claims_SeverityLow               FALSE    FALSE
## Claims_SeverityMedium            FALSE    FALSE
## Claims_Adjustment                FALSE    FALSE
## Policy_TypeLiability-Only       FALSE    FALSE
## Safe_Driver_Discount             FALSE    FALSE
## Multi_Policy_Discount            FALSE    FALSE
## Bundling_Discount                FALSE    FALSE
## Source_of_LeadOnline              FALSE    FALSE
## Source_of_LeadReferral             FALSE    FALSE
## Time_Since_First_Contact         FALSE    FALSE
## Conversion_Status                FALSE    FALSE
## Website_Visits                  FALSE    FALSE
## Inquiries                         FALSE    FALSE
## Quotes_Requested                 FALSE    FALSE
## Time_to_Conversion                FALSE    FALSE
## Credit_Score                      FALSE    FALSE
## Premium_Adjustment_Credit        FALSE    FALSE
## RegionSuburban                   FALSE    FALSE
## RegionUrban                       FALSE    FALSE
## Married_Premium_Discount          FALSE    FALSE
## Prior_Insurance_Premium_Adjustment FALSE    FALSE
## Policy_Adjustment                 FALSE    FALSE
## Total_Discounts                  FALSE    FALSE
## Premium_Adjustment_Region         FALSE    FALSE
## 1 subsets of each size up to 27
## Selection Algorithm: forward
##           Age Is_Senior Marital_StatusMarried Marital_StatusSingle
## 1 ( 1 ) " " " " " "
## 2 ( 1 ) " " " " " "
## 3 ( 1 ) " " " " " "
## 4 ( 1 ) " " " " " "
## 5 ( 1 ) " " " " " "
## 6 ( 1 ) " " " " " "
## 7 ( 1 ) " " " " " "
## 8 ( 1 ) " " " " " "
## 9 ( 1 ) " " " " " "
## 10 ( 1 ) " " " " " "
## 11 ( 1 ) " " " " " "
## 12 ( 1 ) "*" " " " "
## 13 ( 1 ) "*" " " " "
## 14 ( 1 ) "*" " " " "
## 15 ( 1 ) "*" " " " "
## 16 ( 1 ) "*" " " " "
## 17 ( 1 ) "*" " " " "
## 18 ( 1 ) "*" " " " "
## 19 ( 1 ) "*" " " " "
## 20 ( 1 ) "*" " " " "
## 21 ( 1 ) "*" " " " "
## 22 ( 1 ) "*" " " " "
## 23 ( 1 ) "*" " " " "
## 24 ( 1 ) "*" " " " "
## 25 ( 1 ) "*" "*" " "

```

```

## 26  ( 1 ) "*" "*"      " "          "*"
## 27  ( 1 ) "*" "*"      " "          "*"
##           Marital_StatusWidowed Married_Premium_Discount
## 1  ( 1 ) " "           " "          ""
## 2  ( 1 ) " "           " "          ""
## 3  ( 1 ) " "           " "          ""
## 4  ( 1 ) " "           " "          "*"
## 5  ( 1 ) " "           " "          "*"
## 6  ( 1 ) " "           " "          "*"
## 7  ( 1 ) " "           " "          "*"
## 8  ( 1 ) " "           " "          "*"
## 9  ( 1 ) " "           " "          "*"
## 10 ( 1 ) " "           " "          "*"
## 11 ( 1 ) " "           " "          "*"
## 12 ( 1 ) " "           " "          "*"
## 13 ( 1 ) " "           " "          "*"
## 14 ( 1 ) " "           " "          "*"
## 15 ( 1 ) " "           " "          "*"
## 16 ( 1 ) " "           " "          "*"
## 17 ( 1 ) " "           " "          "*"
## 18 ( 1 ) " "           " "          "*"
## 19 ( 1 ) " "           " "          "*"
## 20 ( 1 ) " "           " "          "*"
## 21 ( 1 ) " "           " "          "*"
## 22 ( 1 ) " "           " "          "*"
## 23 ( 1 ) " "           " "          "*"
## 24 ( 1 ) " "           " "          "*"
## 25 ( 1 ) " "           " "          "*"
## 26 ( 1 ) " "           " "          "*"
## 27 ( 1 ) "*"          "*"          "*"

##           Prior_Insurance>5 years Prior_Insurance1-5 years
## 1  ( 1 ) " "           " "          ""
## 2  ( 1 ) " "           " "          ""
## 3  ( 1 ) " "           " "          ""
## 4  ( 1 ) " "           " "          ""
## 5  ( 1 ) " "           " "          ""
## 6  ( 1 ) " "           " "          ""
## 7  ( 1 ) " "           " "          ""
## 8  ( 1 ) " "           " "          ""
## 9  ( 1 ) " "           " "          ""
## 10 ( 1 ) "*"          " "          ""
## 11 ( 1 ) "*"          " "          ""
## 12 ( 1 ) "*"          " "          ""
## 13 ( 1 ) "*"          " "          ""
## 14 ( 1 ) "*"          " "          ""
## 15 ( 1 ) "*"          " "          ""
## 16 ( 1 ) "*"          " "          ""
## 17 ( 1 ) "*"          " "          ""
## 18 ( 1 ) "*"          " "          ""
## 19 ( 1 ) "*"          " "          ""
## 20 ( 1 ) "*"          " "          ""
## 21 ( 1 ) "*"          " "          ""
## 22 ( 1 ) "*"          " "          ""
## 23 ( 1 ) "*"          " "          ""

```

```

## 24  ( 1 ) "*"          " "          " "
## 25  ( 1 ) "*"          " "          " "
## 26  ( 1 ) "*"          " "          " "
## 27  ( 1 ) "*"          " "          " "
##                  Prior_Insurance_Premium_Adjustment Claims_Frequency
## 1  ( 1 ) " "          " "          " "
## 2  ( 1 ) " "          " "          " "
## 3  ( 1 ) " "          " "          " "
## 4  ( 1 ) " "          " "          " "
## 5  ( 1 ) " "          " "          " "
## 6  ( 1 ) "*"          " "          " "
## 7  ( 1 ) "*"          " "          " "
## 8  ( 1 ) "*"          " "          "*"
## 9  ( 1 ) "*"          " "          "*"
## 10 ( 1 ) "*"          " "          "*"
## 11 ( 1 ) "*"          " "          "*"
## 12 ( 1 ) "*"          " "          "*"
## 13 ( 1 ) "*"          " "          "*"
## 14 ( 1 ) "*"          " "          "*"
## 15 ( 1 ) "*"          " "          "*"
## 16 ( 1 ) "*"          " "          "*"
## 17 ( 1 ) "*"          " "          "*"
## 18 ( 1 ) "*"          " "          "*"
## 19 ( 1 ) "*"          " "          "*"
## 20 ( 1 ) "*"          " "          "*"
## 21 ( 1 ) "*"          " "          "*"
## 22 ( 1 ) "*"          " "          "*"
## 23 ( 1 ) "*"          " "          "*"
## 24 ( 1 ) "*"          " "          "*"
## 25 ( 1 ) "*"          " "          "*"
## 26 ( 1 ) "*"          " "          "*"
## 27 ( 1 ) "*"          " "          "*"
##                  Claims_SeverityLow Claims_SeverityMedium Claims_Adjustment
## 1  ( 1 ) " "          " "          " "
## 2  ( 1 ) " "          " "          "*"
## 3  ( 1 ) " "          " "          "*"
## 4  ( 1 ) " "          " "          "*"
## 5  ( 1 ) " "          " "          "*"
## 6  ( 1 ) " "          " "          "*"
## 7  ( 1 ) " "          " "          "*"
## 8  ( 1 ) " "          " "          "*"
## 9  ( 1 ) " "          " "          "*"
## 10 ( 1 ) " "          " "          "*"
## 11 ( 1 ) " "          " "          "*"
## 12 ( 1 ) " "          " "          "*"
## 13 ( 1 ) " "          " "          "*"
## 14 ( 1 ) " "          " "          "*"
## 15 ( 1 ) " "          " "          "*"
## 16 ( 1 ) " "          " "          "*"
## 17 ( 1 ) " "          " "          "*"
## 18 ( 1 ) " "          " "          "*"
## 19 ( 1 ) " "          " "          "*"
## 20 ( 1 ) " "          " "          "*"
## 21 ( 1 ) " "          " "          "*"

```

```

## 22 ( 1 ) " "      "*"      "*"
## 23 ( 1 ) "*"      "*"      "*"
## 24 ( 1 ) "*"      "*"      "*"
## 25 ( 1 ) "*"      "*"      "*"
## 26 ( 1 ) "*"      "*"      "*"
## 27 ( 1 ) "*"      "*"      "*"
##          Policy_TypeLiability-Only Policy_Adjustment Safe_Driver_Discount
## 1 ( 1 ) " "      "*"      " "
## 2 ( 1 ) " "      "*"      " "
## 3 ( 1 ) " "      "*"      " "
## 4 ( 1 ) " "      "*"      " "
## 5 ( 1 ) " "      "*"      " "
## 6 ( 1 ) " "      "*"      " "
## 7 ( 1 ) " "      "*"      " "
## 8 ( 1 ) " "      "*"      " "
## 9 ( 1 ) " "      "*"      " "
## 10 ( 1 ) " "     "*"      " "
## 11 ( 1 ) " "     "*"      " "
## 12 ( 1 ) " "     "*"      " "
## 13 ( 1 ) " "     "*"      " "
## 14 ( 1 ) " "     "*"      " "
## 15 ( 1 ) " "     "*"      " "
## 16 ( 1 ) " "     "*"      " "
## 17 ( 1 ) " "     "*"      " "
## 18 ( 1 ) " "     "*"      " "
## 19 ( 1 ) " "     "*"      " "
## 20 ( 1 ) " "     "*"      " "
## 21 ( 1 ) " "     "*"      " "
## 22 ( 1 ) " "     "*"      " "
## 23 ( 1 ) " "     "*"      " "
## 24 ( 1 ) " "     "*"      "*"
## 25 ( 1 ) " "     "*"      "*"
## 26 ( 1 ) " "     "*"      "*"
## 27 ( 1 ) " "     "*"      "*"
##          Multi_Policy_Discount Bundling_Discount Total_Discounts
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
## 4 ( 1 ) " "      " "      " "
## 5 ( 1 ) " "      " "      " "
## 6 ( 1 ) " "      " "      " "
## 7 ( 1 ) " "      " "      "*"
## 8 ( 1 ) " "      " "      "*"
## 9 ( 1 ) " "      " "      "*"
## 10 ( 1 ) " "     " "      "*"
## 11 ( 1 ) " "     " "      "*"
## 12 ( 1 ) " "     " "      "*"
## 13 ( 1 ) " "     " "      "*"
## 14 ( 1 ) " "     " "      "*"
## 15 ( 1 ) " "     " "      "*"
## 16 ( 1 ) " "     " "      "*"
## 17 ( 1 ) " "     "*"      "*"
## 18 ( 1 ) " "     "*"      "*"
## 19 ( 1 ) " "     "*"      "*"

```

```

## 20  ( 1 ) " "      "*"      "*"
## 21  ( 1 ) " "      "*"      "*"
## 22  ( 1 ) " "      "*"      "*"
## 23  ( 1 ) " "      "*"      "*"
## 24  ( 1 ) " "      "*"      "*"
## 25  ( 1 ) " "      "*"      "*"
## 26  ( 1 ) " "      "*"      "*"
## 27  ( 1 ) " "      "*"      "*"
##          Source_of_LeadOnline Source_of_LeadReferral Time_Since_First_Contact
## 1  ( 1 ) " "      " "      " "
## 2  ( 1 ) " "      " "      " "
## 3  ( 1 ) " "      " "      " "
## 4  ( 1 ) " "      " "      " "
## 5  ( 1 ) " "      " "      " "
## 6  ( 1 ) " "      " "      " "
## 7  ( 1 ) " "      " "      " "
## 8  ( 1 ) " "      " "      " "
## 9  ( 1 ) " "      " "      " "
## 10 ( 1 ) " "      " "      " "
## 11 ( 1 ) " "      " "      "*"
## 12 ( 1 ) " "      " "      "*"
## 13 ( 1 ) "*"      " "      "*"
## 14 ( 1 ) "*"      "*"      "*"
## 15 ( 1 ) "*"      "*"      "*"
## 16 ( 1 ) "*"      "*"      "*"
## 17 ( 1 ) "*"      "*"      "*"
## 18 ( 1 ) "*"      "*"      "*"
## 19 ( 1 ) "*"      "*"      "*"
## 20 ( 1 ) "*"      "*"      "*"
## 21 ( 1 ) "*"      "*"      "*"
## 22 ( 1 ) "*"      "*"      "*"
## 23 ( 1 ) "*"      "*"      "*"
## 24 ( 1 ) "*"      "*"      "*"
## 25 ( 1 ) "*"      "*"      "*"
## 26 ( 1 ) "*"      "*"      "*"
## 27 ( 1 ) "*"      "*"      "*"
##          Conversion_Status Website_Visits Inquiries Quotes_Requested
## 1  ( 1 ) " "      " "      " "      " "
## 2  ( 1 ) " "      " "      " "      " "
## 3  ( 1 ) " "      " "      " "      " "
## 4  ( 1 ) " "      " "      " "      " "
## 5  ( 1 ) " "      " "      " "      " "
## 6  ( 1 ) " "      " "      " "      " "
## 7  ( 1 ) " "      " "      " "      " "
## 8  ( 1 ) " "      " "      " "      " "
## 9  ( 1 ) " "      " "      " "      " "
## 10 ( 1 ) " "      " "      " "      " "
## 11 ( 1 ) " "      " "      " "      " "
## 12 ( 1 ) " "      " "      " "      " "
## 13 ( 1 ) " "      " "      " "      " "
## 14 ( 1 ) " "      " "      " "      " "
## 15 ( 1 ) " "      " "      " "      " "
## 16 ( 1 ) " "      "*"      " "      " "
## 17 ( 1 ) " "      "*"      " "      " "

```

```

## 18 ( 1 ) " "      "*"      " "      " "
## 19 ( 1 ) "*"      "*"      " "      " "
## 20 ( 1 ) "*"      "*"      " "      "*" 
## 21 ( 1 ) "*"      "*"      "*"      "*" 
## 22 ( 1 ) "*"      "*"      "*"      "*" 
## 23 ( 1 ) "*"      "*"      "*"      "*" 
## 24 ( 1 ) "*"      "*"      "*"      "*" 
## 25 ( 1 ) "*"      "*"      "*"      "*" 
## 26 ( 1 ) "*"      "*"      "*"      "*" 
## 27 ( 1 ) "*"      "*"      "*"      "*" 

##          Time_to_Conversion Credit_Score Premium_Adjustment_Credit
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      "*" 
## 4 ( 1 ) " "      " "      "*" 
## 5 ( 1 ) " "      " "      "*" 
## 6 ( 1 ) " "      " "      "*" 
## 7 ( 1 ) " "      " "      "*" 
## 8 ( 1 ) " "      " "      "*" 
## 9 ( 1 ) " "      " "      "*" 
## 10 ( 1 ) " "     " "      "*" 
## 11 ( 1 ) " "     " "      "*" 
## 12 ( 1 ) " "     " "      "*" 
## 13 ( 1 ) " "     " "      "*" 
## 14 ( 1 ) " "     " "      "*" 
## 15 ( 1 ) " "     " "      "*" 
## 16 ( 1 ) " "     " "      "*" 
## 17 ( 1 ) " "     " "      "*" 
## 18 ( 1 ) "*"     " "      "*" 
## 19 ( 1 ) "*"     " "      "*" 
## 20 ( 1 ) "*"     " "      "*" 
## 21 ( 1 ) "*"     " "      "*" 
## 22 ( 1 ) "*"     " "      "*" 
## 23 ( 1 ) "*"     " "      "*" 
## 24 ( 1 ) "*"     " "      "*" 
## 25 ( 1 ) "*"     " "      "*" 
## 26 ( 1 ) "*"     "*"      "*" 
## 27 ( 1 ) "*"     "*"      "*" 

##          RegionSuburban RegionUrban Premium_Adjustment_Region
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      " "
## 4 ( 1 ) " "      " "      " "
## 5 ( 1 ) " "      " "      "*" 
## 6 ( 1 ) " "      " "      "*" 
## 7 ( 1 ) " "      " "      "*" 
## 8 ( 1 ) " "      " "      "*" 
## 9 ( 1 ) " "      " "      "*" 
## 10 ( 1 ) " "     " "      "*" 
## 11 ( 1 ) " "     " "      "*" 
## 12 ( 1 ) " "     " "      "*" 
## 13 ( 1 ) " "     " "      "*" 
## 14 ( 1 ) " "     " "      "*" 
## 15 ( 1 ) "*"     " "      "*"

```

```

## 16  ( 1 ) "*"      " "      "*"
## 17  ( 1 ) "*"      " "      "*"
## 18  ( 1 ) "*"      " "      "*"
## 19  ( 1 ) "*"      " "      "*"
## 20  ( 1 ) "*"      " "      "*"
## 21  ( 1 ) "*"      " "      "*"
## 22  ( 1 ) "*"      " "      "*"
## 23  ( 1 ) "*"      " "      "*"
## 24  ( 1 ) "*"      " "      "*"
## 25  ( 1 ) "*"      " "      "*"
## 26  ( 1 ) "*"      " "      "*"
## 27  ( 1 ) "*"      " "      "*"

for (metric in names(summary_fwd)) {
  if (metric != "which" && metric != "outmat" && metric != "obj") {
    cat(metric, ":\\n")
    print(summary_fwd[[metric]])
    cat("\\n")
  }
}

## rsq :
## [1] 0.4506927 0.6195438 0.7303598 0.8153171 0.8899720 0.9463980 1.0000000
## [8] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [15] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [22] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##
## rss :
## [1] 9.247868e+07 6.405173e+07 4.539530e+07 3.109230e+07 1.852377e+07
## [6] 9.024170e+06 6.937800e-19 6.931723e-19 6.927140e-19 6.924290e-19
## [11] 6.921786e-19 6.919949e-19 6.918299e-19 6.916524e-19 6.915328e-19
## [16] 6.914427e-19 6.913604e-19 6.913146e-19 6.907766e-19 6.907524e-19
## [21] 6.907363e-19 6.907252e-19 6.907051e-19 6.906961e-19 6.906953e-19
## [26] 6.906953e-19 6.906953e-19
##
## adjr2 :
## [1] 0.4506238 0.6194484 0.7302583 0.8152245 0.8899030 0.9463576 1.0000000
## [8] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [15] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [22] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##
## cp :
## [1] 1.063505e+30 7.365953e+29 5.220463e+29 3.575616e+29 2.130235e+29
## [6] 1.037780e+29 1.847493e+01 1.348582e+01 1.021545e+01 8.938483e+00
## [11] 8.058788e+00 7.945972e+00 8.048675e+00 8.006987e+00 8.631179e+00
## [16] 9.595115e+00 1.064952e+01 1.212191e+01 7.935658e+00 9.657708e+00
## [21] 1.147190e+01 1.334453e+01 1.511267e+01 1.700962e+01 1.900053e+01
## [26] 2.100002e+01 2.300000e+01
##
## bic :
## [1] -4760.431 -7680.929 -10417.941 -13427.462 -17549.291 -23276.233
## [7] -484499.780 -484497.786 -484494.077 -484488.374 -484482.275 -484475.408
## [13] -484468.326 -484461.389 -484453.784 -484445.839 -484437.804 -484429.349
## [19] -484426.573 -484417.868 -484409.071 -484400.214 -484391.463 -484382.582

```

```

## [25] -484373.607 -484364.624 -484355.639

set.seed(888)
cp <- summary_fwd$cp
bic <- summary_fwd$bic
adjr2 <- summary_fwd$adjr2

bic_valid <- bic[is.finite(bic)]

num_predictors <- 1:length(cp)

par(mfrow = c(1, 3), oma = c(0, 0, 2, 0), mar = c(4, 4, 2, 1)) # Outer margin and plot margins

# Plot Cp
cp_diff <- abs(cp - (num_predictors + 1))
best_cp <- which.min(cp_diff)
plot(num_predictors, cp, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = expression(C[p]),
     main = "Cp Plot", cex.axis = 0.9, cex.lab = 1.1)
points(best_cp, cp[best_cp], col = "red", cex = 2, pch = 4)
text(best_cp, cp[best_cp], labels = paste("Best:", best_cp), pos = 4, col = "red", cex = 0.9)

# Plot BIC

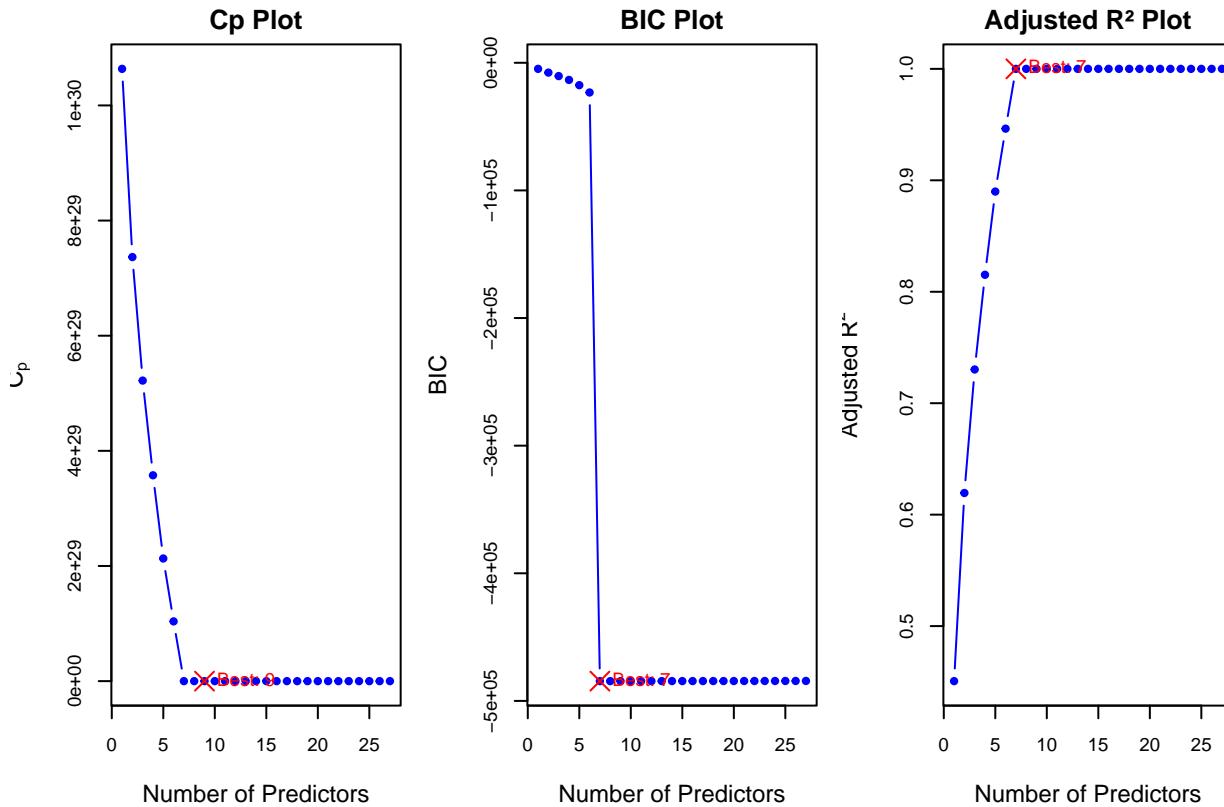
num_predictors_bic <- 1:length(bic_valid)
plot(num_predictors_bic, bic_valid, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = "BIC", main = "BIC Plot",
     cex.axis = 0.9, cex.lab = 1.1)
best_bic <- which.min(bic_valid)
points(best_bic, bic_valid[best_bic], col = "red", cex = 2, pch = 4)
text(best_bic, bic_valid[best_bic], labels = paste("Best:", best_bic), pos = 4, col = "red", cex = 0.9)

# Plot Adjusted R^2
plot(num_predictors, adjr2, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = expression(Adjusted~R^2), main = "Adjusted R^2 Plot",
     cex.axis = 0.9, cex.lab = 1.1)
best_adjr2 <- which.max(adjr2)
points(best_adjr2, adjr2[best_adjr2], col = "red", cex = 2, pch = 4)
text(best_adjr2, adjr2[best_adjr2], labels = paste("Best:", best_adjr2), pos = 4, col = "red", cex = 0.9)

# Add main title
mtext("Forward Selection Model", side = 3, line = 0, cex = 1.5, outer = TRUE)

```

## Forward Selection Model



Perhatikan gambar di atas, gambar di atas menunjukkan bahwa model terbaik berdasarkan metrik *Mallow's Cp* adalah model dengan 9 variabel, berdasarkan metrik BIC adalah model dengan 7 variabel bebas, dan berdasarkan metrik *Adjusted R<sup>2</sup>* adalah model dengan 7 variabel. Maka dapat disimpulkan bahwa dengan metode *forward selection* model terbaik diberikan oleh model dengan 7 variabel karena memberikan performa terbaik berdasarkan metrik BIC dan *Adjusted R<sup>2</sup>*, serta memberikan performa yang masih cukup baik pada metrik *Mallow's Cp*.

### Koefesien Model Terbaik *Forward Selection*

```
coef(regfit_fwd, id=7)
```

```
##                               (Intercept)          Prior_Insurance>5 years
##                         2148.0951131           17.0193005
##             Claims_SeverityLow          Safe_Driver_Discount
##                      -33.7243083           -46.7255651
##             Bundling_Discount          Conversion_Status
##                      -55.1031038           -22.6923810
## Prior_Insurance_Premium_Adjustment Premium_Adjustment_Region
##                           1.1372815            0.9703715
```

### Stepwise Backward Selection

Pada tahap ini, proses pemodelan dilakukan menggunakan fungsi `subset_selection` dengan metode *backward selection* dari pustaka `leaps`. Tujuannya adalah untuk membangun model regresi yang memprediksi

variabel Premium\_Amount berdasarkan sejumlah variabel prediktor yang tersedia dalam data. Karena alur pengerjaan pada bagian ini serupa dengan langkah-langkah yang telah dijelaskan pada bagian sebelumnya, penjelasan teknis yang sama tidak akan diulang kembali.

```

set.seed(888)
regfit_bwd <- regsubsets(Premium_Amount ~ ., data = train,
nvmax = 27, method = "backward")

## Warning in leaps.setup(x, y, wt = wt, nbest = nbest, nvmax = nvmax, force.in =
## force.in, : 5 linear dependencies found

## Reordering variables and trying again:

## Warning in rval$lopt[] <- rval$vorder[rval$lopt]: number of items to replace is
## not a multiple of replacement length

summary_bwd <- summary(regfit_bwd)

print(summary_bwd)

## Subset selection object
## Call: regsubsets.formula(Premium_Amount ~ ., data = train, nvmax = 27,
##   method = "backward")
## 32 Variables  (and intercept)
##                                         Forced in    Forced out
## Age                               FALSE      FALSE
## Is_Senior                          FALSE      FALSE
## Marital_StatusMarried              FALSE      FALSE
## Marital_StatusSingle               FALSE      FALSE
## Marital_StatusWidowed             FALSE      FALSE
## Prior_Insurance>5 years            FALSE      FALSE
## Prior_Insurance1-5 years           FALSE      FALSE
## Claims_Frequency                  FALSE      FALSE
## Claims_SeverityLow                FALSE      FALSE
## Claims_SeverityMedium              FALSE      FALSE
## Claims_Adjustment                 FALSE      FALSE
## Policy_TypeLiability-Only         FALSE      FALSE
## Safe_Driver_Discount              FALSE      FALSE
## Multi_Policy_Discount             FALSE      FALSE
## Bundling_Discount                 FALSE      FALSE
## Source_of_LeadOnline               FALSE      FALSE
## Source_of_LeadReferral             FALSE      FALSE
## Time_Since_First_Contact          FALSE      FALSE
## Conversion_Status                 FALSE      FALSE
## Website_Visits                   FALSE      FALSE
## Inquiries                          FALSE      FALSE
## Quotes_Requested                  FALSE      FALSE
## Time_to_Conversion                FALSE      FALSE
## Credit_Score                       FALSE      FALSE
## Premium_Adjustment_Credit          FALSE      FALSE
## RegionSuburban                    FALSE      FALSE
## RegionUrban                        FALSE      FALSE

```

```

## Married_Premium_Discount      FALSE  FALSE
## Prior_Insurance_Premium_Adjustment FALSE  FALSE
## Policy_Adjustment             FALSE  FALSE
## Total_Discounts               FALSE  FALSE
## Premium_Adjustment_Region    FALSE  FALSE
## 1 subsets of each size up to 27
## Selection Algorithm: backward
##          Age Is_Senior Marital_StatusMarried Marital_StatusSingle
## 1  ( 1 ) " " " "      " "           " "
## 2  ( 1 ) " " " "      " "           " "
## 3  ( 1 ) " " " "      " "           " "
## 4  ( 1 ) " " " "      "*"          " "
## 5  ( 1 ) " " " "      "*"          " "
## 6  ( 1 ) " " " "      "*"          " "
## 7  ( 1 ) " " " "      "*"          " "
## 8  ( 1 ) " " " "      "*"          " "
## 9  ( 1 ) " " " "      "*"          " "
## 10 ( 1 ) " " " "      "*"          " "
## 11 ( 1 ) " " " "      "*"          " "
## 12 ( 1 ) " " " "      "*"          " "
## 13 ( 1 ) " " " "      "*"          " "
## 14 ( 1 ) " " " "      "*"          " "
## 15 ( 1 ) " " " "      "*"          "*"
## 16 ( 1 ) " " " "      "*"          "*"
## 17 ( 1 ) " " " "      "*"          "*"
## 18 ( 1 ) " " " "      "*"          "*"
## 19 ( 1 ) " " " "      "*"          "*"
## 20 ( 1 ) "*" " "      "*"          "*"
## 21 ( 1 ) "*" " "      "*"          "*"
## 22 ( 1 ) "*" " "      "*"          "*"
## 23 ( 1 ) "*" " "      "*"          "*"
## 24 ( 1 ) "*" " "      "*"          "*"
## 25 ( 1 ) "*" "*"      "*"          "*"
## 26 ( 1 ) "*" "*"      "*"          "*"
## 27 ( 1 ) "*" "*"      "*"          "*"
##          Marital_StatusWidowed Married_Premium_Discount
## 1  ( 1 ) " "           " "
## 2  ( 1 ) " "           " "
## 3  ( 1 ) " "           " "
## 4  ( 1 ) " "           " "
## 5  ( 1 ) " "           " "
## 6  ( 1 ) " "           " "
## 7  ( 1 ) " "           " "
## 8  ( 1 ) " "           " "
## 9  ( 1 ) " "           " "
## 10 ( 1 ) " "           " "
## 11 ( 1 ) " "           " "
## 12 ( 1 ) " "           " "
## 13 ( 1 ) " "           " "
## 14 ( 1 ) " "           " "
## 15 ( 1 ) " "           " "
## 16 ( 1 ) " "           " "
## 17 ( 1 ) " "           " "
## 18 ( 1 ) " "           " "

```

```

## 19 ( 1 ) " "          " "
## 20 ( 1 ) " "          " "
## 21 ( 1 ) " "          " "
## 22 ( 1 ) " "          " "
## 23 ( 1 ) " "          " "
## 24 ( 1 ) " "          " "
## 25 ( 1 ) " "          " "
## 26 ( 1 ) " "          " "
## 27 ( 1 ) "*"          " "
##
##           Prior_Insurance>5 years Prior_Insurance1-5 years
## 1 ( 1 ) " "          " "
## 2 ( 1 ) " "          " "
## 3 ( 1 ) " "          " "
## 4 ( 1 ) " "          " "
## 5 ( 1 ) " "          " "
## 6 ( 1 ) "*"          " "
## 7 ( 1 ) "*"          " "
## 8 ( 1 ) "*"          " "
## 9 ( 1 ) "*"          "*"
## 10 ( 1 ) "*"         "*"
## 11 ( 1 ) "*"         "*"
## 12 ( 1 ) "*"         "*"
## 13 ( 1 ) "*"         "*"
## 14 ( 1 ) "*"         "*"
## 15 ( 1 ) "*"         "*"
## 16 ( 1 ) "*"         "*"
## 17 ( 1 ) "*"         "*"
## 18 ( 1 ) "*"         "*"
## 19 ( 1 ) "*"         "*"
## 20 ( 1 ) "*"         "*"
## 21 ( 1 ) "*"         "*"
## 22 ( 1 ) "*"         "*"
## 23 ( 1 ) "*"         "*"
## 24 ( 1 ) "*"         "*"
## 25 ( 1 ) "*"         "*"
## 26 ( 1 ) "*"         "*"
## 27 ( 1 ) "*"         "*"
##
##           Prior_Insurance_Premium_Adjustment Claims_Frequency
## 1 ( 1 ) " "          " "
## 2 ( 1 ) " "          " "
## 3 ( 1 ) " "          " "
## 4 ( 1 ) " "          " "
## 5 ( 1 ) " "          " "
## 6 ( 1 ) " "          " "
## 7 ( 1 ) " "          " "
## 8 ( 1 ) " "          " "
## 9 ( 1 ) " "          " "
## 10 ( 1 ) " "         " "
## 11 ( 1 ) " "         " "
## 12 ( 1 ) " "         "*"
## 13 ( 1 ) " "         "*"
## 14 ( 1 ) " "         "*"
## 15 ( 1 ) " "         "*"
## 16 ( 1 ) " "         "*"

```

```

## 17 ( 1 ) " "          "*"
## 18 ( 1 ) " "          "*"
## 19 ( 1 ) " "          "*"
## 20 ( 1 ) " "          "*"
## 21 ( 1 ) " "          "*"
## 22 ( 1 ) " "          "*"
## 23 ( 1 ) " "          "*"
## 24 ( 1 ) " "          "*"
## 25 ( 1 ) " "          "*"
## 26 ( 1 ) " "          "*"
## 27 ( 1 ) " "          "*"

##          Claims_SeverityLow Claims_SeverityMedium Claims_Adjustment
## 1 ( 1 ) " "           " "           " "
## 2 ( 1 ) " "           " "           "*"
## 3 ( 1 ) " "           " "           "*"
## 4 ( 1 ) " "           " "           "*"
## 5 ( 1 ) " "           " "           "*"
## 6 ( 1 ) " "           " "           "*"
## 7 ( 1 ) " "           " "           "*"
## 8 ( 1 ) " "           " "           "*"
## 9 ( 1 ) " "           " "           "*"
## 10 ( 1 ) " "          " "           "*"
## 11 ( 1 ) " "          " "           "*"
## 12 ( 1 ) " "          " "           "*"
## 13 ( 1 ) " "          " "           "*"
## 14 ( 1 ) " "          " "           "*"
## 15 ( 1 ) " "          " "           "*"
## 16 ( 1 ) " "          " "           "*"
## 17 ( 1 ) " "          " "           "*"
## 18 ( 1 ) " "          " "           "*"
## 19 ( 1 ) " "          " "           "*"
## 20 ( 1 ) " "          " "           "*"
## 21 ( 1 ) " "          " "           "*"
## 22 ( 1 ) " "          " "           "*"
## 23 ( 1 ) " "          "*"          "*"
## 24 ( 1 ) "*"          "*"          "*"
## 25 ( 1 ) "*"          "*"          "*"
## 26 ( 1 ) "*"          "*"          "*"
## 27 ( 1 ) "*"          "*"          "*"

##          Policy_TypeLiability-Only Policy_Adjustment Safe_Driver_Discount
## 1 ( 1 ) "*"          " "           " "
## 2 ( 1 ) "*"          " "           " "
## 3 ( 1 ) "*"          " "           " "
## 4 ( 1 ) "*"          " "           " "
## 5 ( 1 ) "*"          " "           " "
## 6 ( 1 ) "*"          " "           " "
## 7 ( 1 ) "*"          " "           " "
## 8 ( 1 ) "*"          " "           "*"
## 9 ( 1 ) "*"          " "           "*"
## 10 ( 1 ) "*"         " "           "*"
## 11 ( 1 ) "*"         " "           "*"
## 12 ( 1 ) "*"         " "           "*"
## 13 ( 1 ) "*"         " "           "*"
## 14 ( 1 ) "*"         " "           "*"

```

```

## 15 ( 1 ) "*"      " "      "*"
## 16 ( 1 ) "*"      " "      "*"
## 17 ( 1 ) "*"      " "      "*"
## 18 ( 1 ) "*"      " "      "*"
## 19 ( 1 ) "*"      " "      "*"
## 20 ( 1 ) "*"      " "      "*"
## 21 ( 1 ) "*"      " "      "*"
## 22 ( 1 ) "*"      " "      "*"
## 23 ( 1 ) "*"      " "      "*"
## 24 ( 1 ) "*"      " "      "*"
## 25 ( 1 ) "*"      " "      "*"
## 26 ( 1 ) "*"      " "      "*"
## 27 ( 1 ) "*"      " "      "*"
##          Multi_Policy_Discount Bundling_Discount Total_Discounts
## 1 ( 1 ) " "        " "        " "
## 2 ( 1 ) " "        " "        " "
## 3 ( 1 ) " "        " "        " "
## 4 ( 1 ) " "        " "        " "
## 5 ( 1 ) " "        " "        " "
## 6 ( 1 ) " "        " "        " "
## 7 ( 1 ) "*"       " "        " "
## 8 ( 1 ) "*"       " "        " "
## 9 ( 1 ) "*"       " "        " "
## 10 ( 1 ) "*"      " "        " "
## 11 ( 1 ) "*"      "*"       " "
## 12 ( 1 ) "*"      "*"       " "
## 13 ( 1 ) "*"      "*"       " "
## 14 ( 1 ) "*"      "*"       " "
## 15 ( 1 ) "*"      "*"       " "
## 16 ( 1 ) "*"      "*"       " "
## 17 ( 1 ) "*"      "*"       " "
## 18 ( 1 ) "*"      "*"       " "
## 19 ( 1 ) "*"      "*"       " "
## 20 ( 1 ) "*"      "*"       " "
## 21 ( 1 ) "*"      "*"       " "
## 22 ( 1 ) "*"      "*"       " "
## 23 ( 1 ) "*"      "*"       " "
## 24 ( 1 ) "*"      "*"       " "
## 25 ( 1 ) "*"      "*"       " "
## 26 ( 1 ) "*"      "*"       " "
## 27 ( 1 ) "*"      "*"       " "
##          Source_of_LeadOnline Source_of_LeadReferral Time_Since_First_Contact
## 1 ( 1 ) " "        " "        " "
## 2 ( 1 ) " "        " "        " "
## 3 ( 1 ) " "        " "        " "
## 4 ( 1 ) " "        " "        " "
## 5 ( 1 ) " "        " "        " "
## 6 ( 1 ) " "        " "        " "
## 7 ( 1 ) " "        " "        " "
## 8 ( 1 ) " "        " "        " "
## 9 ( 1 ) " "        " "        " "
## 10 ( 1 ) " "       " "        " "
## 11 ( 1 ) " "       " "        " "
## 12 ( 1 ) " "       " "        " "

```

```

## 13 ( 1 ) " "      " "      " "
## 14 ( 1 ) " "      " "      " "
## 15 ( 1 ) " "      " "      " "
## 16 ( 1 ) " "      " "      "*"
## 17 ( 1 ) "*"     " "      "*"
## 18 ( 1 ) "*"     " *"    "*"
## 19 ( 1 ) "*"     " *"    "*"
## 20 ( 1 ) "*"     " *"    "*"
## 21 ( 1 ) "*"     " *"    "*"
## 22 ( 1 ) "*"     " *"    "*"
## 23 ( 1 ) "*"     " *"    "*"
## 24 ( 1 ) "*"     " *"    "*"
## 25 ( 1 ) "*"     " *"    "*"
## 26 ( 1 ) "*"     " *"    "*"
## 27 ( 1 ) "*"     " *"    "*"
##               Conversion_Status Website_Visits Inquiries Quotes_Requested
## 1 ( 1 ) " "      " "      " "      " "
## 2 ( 1 ) " "      " "      " "      " "
## 3 ( 1 ) " "      " "      " "      " "
## 4 ( 1 ) " "      " "      " "      " "
## 5 ( 1 ) " "      " "      " "      " "
## 6 ( 1 ) " "      " "      " "      " "
## 7 ( 1 ) " "      " "      " "      " "
## 8 ( 1 ) " "      " "      " "      " "
## 9 ( 1 ) " "      " "      " "      " "
## 10 ( 1 ) " "     " "      " "      " "
## 11 ( 1 ) " "     " "      " "      " "
## 12 ( 1 ) " "     " "      " "      " "
## 13 ( 1 ) " "     " "      " "      " "
## 14 ( 1 ) "*"     " "      " "      " "
## 15 ( 1 ) "*"     " "      " "      " "
## 16 ( 1 ) "*"     " "      " "      " "
## 17 ( 1 ) "*"     " "      " "      " "
## 18 ( 1 ) "*"     " "      " "      " "
## 19 ( 1 ) "*"     " *"    " "      " "
## 20 ( 1 ) "*"     " *"    " "      " "
## 21 ( 1 ) "*"     " *"    " *"    "*"
## 22 ( 1 ) "*"     " *"    " *"    "*"
## 23 ( 1 ) "*"     " *"    " *"    "*"
## 24 ( 1 ) "*"     " *"    " *"    "*"
## 25 ( 1 ) "*"     " *"    " *"    "*"
## 26 ( 1 ) "*"     " *"    " *"    "*"
## 27 ( 1 ) "*"     " *"    " *"    "*"
##               Time_to_Conversion Credit_Score Premium_Adjustment_Credit
## 1 ( 1 ) " "      " "      " "
## 2 ( 1 ) " "      " "      " "
## 3 ( 1 ) " "      " "      "*"
## 4 ( 1 ) " "      " "      "*"
## 5 ( 1 ) " "      " "      "*"
## 6 ( 1 ) " "      " "      "*"
## 7 ( 1 ) " "      " "      "*"
## 8 ( 1 ) " "      " "      "*"
## 9 ( 1 ) " "      " "      "*"
## 10 ( 1 ) " "     " "      "*"

```

```

## 11  ( 1 ) " "      " "      "*"
## 12  ( 1 ) " "      " "      "*"
## 13  ( 1 ) "*"     " "      "*"
## 14  ( 1 ) "*"     " "      "*"
## 15  ( 1 ) "*"     " "      "*"
## 16  ( 1 ) "*"     " "      "*"
## 17  ( 1 ) "*"     " "      "*"
## 18  ( 1 ) "*"     " "      "*"
## 19  ( 1 ) "*"     " "      "*"
## 20  ( 1 ) "*"     " "      "*"
## 21  ( 1 ) "*"     " "      "*"
## 22  ( 1 ) "*"     " "      "*"
## 23  ( 1 ) "*"     " "      "*"
## 24  ( 1 ) "*"     " "      "*"
## 25  ( 1 ) "*"     " "      "*"
## 26  ( 1 ) "*"     "*"     "*"
## 27  ( 1 ) "*"     "*"     "*"

##          RegionSuburban RegionUrban Premium_Adjustment_Region
## 1  ( 1 ) " "      " "      " "
## 2  ( 1 ) " "      " "      " "
## 3  ( 1 ) " "      " "      " "
## 4  ( 1 ) " "      " "      " "
## 5  ( 1 ) " "      "*"     " "
## 6  ( 1 ) " "      "*"     " "
## 7  ( 1 ) " "      "*"     " "
## 8  ( 1 ) " "      "*"     " "
## 9  ( 1 ) " "      "*"     " "
## 10 ( 1 ) "*"     "*"     " "
## 11 ( 1 ) "*"     "*"     " "
## 12 ( 1 ) "*"     "*"     " "
## 13 ( 1 ) "*"     "*"     " "
## 14 ( 1 ) "*"     "*"     " "
## 15 ( 1 ) "*"     "*"     " "
## 16 ( 1 ) "*"     "*"     " "
## 17 ( 1 ) "*"     "*"     " "
## 18 ( 1 ) "*"     "*"     " "
## 19 ( 1 ) "*"     "*"     " "
## 20 ( 1 ) "*"     "*"     " "
## 21 ( 1 ) "*"     "*"     " "
## 22 ( 1 ) "*"     "*"     " "
## 23 ( 1 ) "*"     "*"     " "
## 24 ( 1 ) "*"     "*"     " "
## 25 ( 1 ) "*"     "*"     " "
## 26 ( 1 ) "*"     "*"     " "
## 27 ( 1 ) "*"     "*"     " "

for (metric in names(summary_fwd)) {
  if (metric != "which" && metric != "outmat" && metric != "obj") {
    cat(metric, ":\\n")
    print(summary_fwd[[metric]])
    cat("\\n")
  }
}

```

```

## rsq :
## [1] 0.4506927 0.6195438 0.7303598 0.8153171 0.8899720 0.9463980 1.0000000
## [8] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [15] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [22] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##
## rss :
## [1] 9.247868e+07 6.405173e+07 4.539530e+07 3.109230e+07 1.852377e+07
## [6] 9.024170e+06 6.937800e-19 6.931723e-19 6.927140e-19 6.924290e-19
## [11] 6.921786e-19 6.919949e-19 6.918299e-19 6.916524e-19 6.915328e-19
## [16] 6.914427e-19 6.913604e-19 6.913146e-19 6.907766e-19 6.907524e-19
## [21] 6.907363e-19 6.907252e-19 6.907051e-19 6.906961e-19 6.906953e-19
## [26] 6.906953e-19 6.906953e-19
##
## adjr2 :
## [1] 0.4506238 0.6194484 0.7302583 0.8152245 0.8899030 0.9463576 1.0000000
## [8] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [15] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
## [22] 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000 1.0000000
##
## cp :
## [1] 1.063505e+30 7.365953e+29 5.220463e+29 3.575616e+29 2.130235e+29
## [6] 1.037780e+29 1.847493e+01 1.348582e+01 1.021545e+01 8.938483e+00
## [11] 8.058788e+00 7.945972e+00 8.048675e+00 8.006987e+00 8.631179e+00
## [16] 9.595115e+00 1.064952e+01 1.212191e+01 7.935658e+00 9.657708e+00
## [21] 1.147190e+01 1.334453e+01 1.511267e+01 1.700962e+01 1.900053e+01
## [26] 2.100002e+01 2.300000e+01
##
## bic :
## [1] -4760.431 -7680.929 -10417.941 -13427.462 -17549.291 -23276.233
## [7] -484499.780 -484497.786 -484494.077 -484488.374 -484482.275 -484475.408
## [13] -484468.326 -484461.389 -484453.784 -484445.839 -484437.804 -484429.349
## [19] -484426.573 -484417.868 -484409.071 -484400.214 -484391.463 -484382.582
## [25] -484373.607 -484364.624 -484355.639

set.seed(888)
cp <- summary_bwd$cp
bic <- summary_bwd$bic
adjr2 <- summary_bwd$adjr2

# Hapus nilai -Inf di BIC
bic_valid <- bic[is.finite(bic)]

num_predictors <- 1:length(cp)

par(mfrow = c(1, 3), oma = c(0, 0, 2, 0), mar = c(4, 4, 2, 1)) # Outer margin and

# Plot Cp
cp_diff <- abs(cp - (num_predictors + 1))
best_cp <- which.min(cp_diff)
plot(num_predictors, cp, type = "b", pch = 20, col = "blue",
     xlab = "Number of Predictors", ylab = expression(C[p]),
     main = "Cp Plot", cex.axis = 0.9, cex.lab = 1.1)

```

```

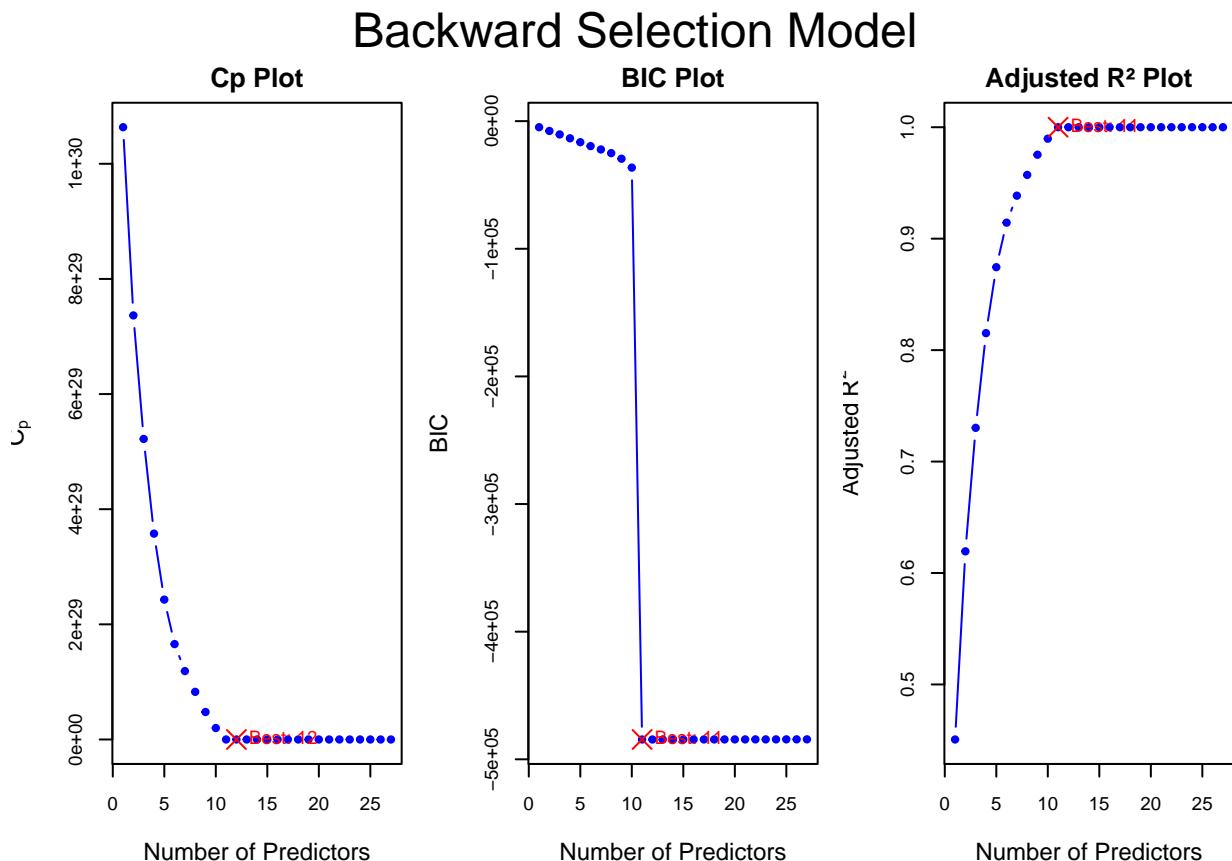
points(best_cp, cp[best_cp], col = "red", cex = 2, pch = 4)
text(best_cp, cp[best_cp], labels = paste("Best:", best_cp), pos = 4, col = "red", cex = 0.9)

# Plot BIC
num_predictors_bic <- 1:length(bic_valid)
plot(num_predictors_bic, bic_valid, type = "b", pch = 20, col = "blue",
      xlab = "Number of Predictors", ylab = "BIC", main = "BIC Plot",
      cex.axis = 0.9, cex.lab = 1.1)
best_bic <- which.min(bic_valid)
points(best_bic, bic_valid[best_bic], col = "red", cex = 2, pch = 4)
text(best_bic, bic_valid[best_bic], labels = paste("Best:", best_bic), pos = 4, col = "red", cex = 0.9)

# Plot Adjusted R^2
plot(num_predictors, adjr2, type = "b", pch = 20, col = "blue",
      xlab = "Number of Predictors", ylab = expression(Adjusted~R^2), main = "Adjusted R^2 Plot",
      cex.axis = 0.9, cex.lab = 1.1)
best_adjr2 <- which.max(adjr2)
points(best_adjr2, adjr2[best_adjr2], col = "red", cex = 2, pch = 4)
text(best_adjr2, adjr2[best_adjr2], labels = paste("Best:", best_adjr2), pos = 4, col = "red", cex = 0.9)

# Add main title
mtext("Backward Selection Model", side = 3, line = 0, cex = 1.5, outer = TRUE)

```



Perhatikan gambar di atas, gambar di atas menunjukkan bahwa model terbaik berdasarkan metrik *Mallow's Cp* adalah model dengan 12 variabel, berdasarkan metrik BIC adalah model dengan 11 variabel bebas, dan berdasarkan metrik Adjusted  $R^2$  adalah model dengan 11 variabel. Maka dapat disimpulkan bahwa dengan

metode *backward selection* model terbaik diberikan oleh model dengan 11 variabel karena memberikan performa terbaik berdasarkan metrik Cp dan *Adjusted R<sup>2</sup>*, serta memberikan performa yang masih cukup baik pada metrik BIC.

### Koefesien Model Terbaik Backward Selection

```
coef(regfit_bwd, id=12)

##                               (Intercept)          Marital_StatusMarried
##                         2203.6897494           86.0914547
##             Prior_Insurance1-5 years      Claims_Frequency
##                           -0.8394895           69.0361071
##             Claims_SeverityMedium      Safe_Driver_Discount
##                           16.7144192           3.0981636
##             Multi_Policy_Discount     Source_of_LeadOnline
##                           3.1063770           1.5925549
##             Source_of_LeadReferral    Time_Since_First_Contact
##                           -0.2456061           -0.2402518
## Prior_Insurance_Premium_Adjustment Policy_Adjustment
##                           0.9854177           0.9973882
##             Total_Discounts
##                           -1.0397244
```

### Overall Model Terbaik (Soal 4 - a)

Pada bagian ini akan ditentukan model terbaik berdasarkan *best subset selection*, *forward selection*, dan *backward selection*. Hasil yang diperoleh di atas menunjukkan bahwa model yang dibangun dengan metode best subset selection memperoleh hasil yang terbaik di 15 prediktor, sedangkan untuk stepwise forward di 7 prediktor, dan *stepwise backward* di 11 prediktor. Sehingga ketiga model ini akan dibangun kembali dengan variabel yang sama lalu akan diperiksa pemenuhan asumsi regresi linearnya menggunakan fungsi *check\_model* pada pustaka *see* dan *performance*.

```
best_subset=names(coef(regfit_full, id =15))
print(best_subset)

## [1] "(Intercept)"                  "Age"
## [3] "Is_Senior"                   "Marital_StatusMarried"
## [5] "Marital_StatusSingle"        "Marital_StatusWidowed"
## [7] "Prior_Insurance1-5 years"    "Claims_SeverityLow"
## [9] "Claims_SeverityMedium"       "Claims_Adjustment"
## [11] "Safe_Driver_Discount"        "Bundling_Discount"
## [13] "Conversion_Status"          "Married_Premium_Discount"
## [15] "Prior_Insurance_Premium_Adjustment" "Premium_Adjustment_Region"

best_fwd=names(coef(regfit_fwd, id=7))
print(best_fwd)

## [1] "(Intercept)"                  "Prior_Insurance>5 years"
## [3] "Claims_SeverityLow"           "Safe_Driver_Discount"
## [5] "Bundling_Discount"            "Conversion_Status"
## [7] "Prior_Insurance_Premium_Adjustment" "Premium_Adjustment_Region"
```

```

best_bwd=names(coef(regfit_bwd, id=11))
print(best_bwd)

## [1] "(Intercept)"                               "Marital_StatusMarried"
## [3] "Prior_Insurance1-5 years"                 "Claims_Frequency"
## [5] "Safe_Driver_Discount"                     "Multi_Policy_Discount"
## [7] "Source_of_LeadOnline"                      "Source_of_LeadReferral"
## [9] "Time_Since_First_Contact"                  "Prior_Insurance_Premium_Adjustment"
## [11] "Policy_Adjustment"                         "Total_Discounts"

model_best_subset=lm(Premium_Amount~Is_Senior+Marital_Status+Prior_Insurance+Claims_Severity+Claims_Adjustment+Safe_Driver_Discount+Bundling_Discount+Conversion_Status+Married_Premium_Discount+Prior_Insurance_Premium_Adjustment+Premium_Adjustment_Region, data = train)

model_best_fwd=lm(Premium_Amount~Prior_Insurance+Claims_Severity+Safe_Driver_Discount+Bundling_Discount+Conversion_Status+Marital_Status+Prior_Insurance+Claims_Frequency+Safe_Driver_Discount+Married_Premium_Discount+Prior_Insurance_Premium_Adjustment+Premium_Adjustment_Region, data = train)

model_best_bwd=lm(Premium_Amount~Marital_Status+Prior_Insurance+Claims_Frequency+Safe_Driver_Discount+Married_Premium_Discount+Prior_Insurance_Premium_Adjustment+Premium_Adjustment_Region, data = train)

summary(model_best_subset)

##
## Call:
## lm(formula = Premium_Amount ~ Is_Senior + Marital_Status + Prior_Insurance +
##     Claims_Severity + Claims_Adjustment + Safe_Driver_Discount +
##     Bundling_Discount + Conversion_Status + Married_Premium_Discount +
##     Prior_Insurance_Premium_Adjustment + Premium_Adjustment_Region,
##     data = train)
##
## Residuals:
##      Min        1Q    Median        3Q       Max 
## -215.48 -109.65   17.44   69.29  185.10 
##
## Coefficients: (2 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)    
## (Intercept) 2148.16797   6.92490 310.209 < 2e-16 ***
## Is_Senior    2.76398   3.42553  0.807  0.420    
## Marital_StatusMarried 87.55036   4.48816 19.507 < 2e-16 ***
## Marital_StatusSingle  2.18022   4.65699  0.468  0.640    
## Marital_StatusWidowed  3.95879   5.81358  0.681  0.496    
## Prior_Insurance>5 years -100.15705  3.65353 -27.414 < 2e-16 ***
## Prior_Insurance1-5 years -56.12921  3.20465 -17.515 < 2e-16 ***
## Claims_SeverityLow     9.28030   4.54080  2.044  0.041 *  
## Claims_SeverityMedium  10.74421  5.00990  2.145  0.032 *  
## Claims_Adjustment       0.97308   0.02157 45.106 < 2e-16 ***
## Safe_Driver_Discount  -49.54005  3.13942 -15.780 < 2e-16 ***
## Bundling_Discount      -53.66969  4.24180 -12.653 < 2e-16 ***
## Conversion_Status     -18.08255  2.53283 -7.139 1.02e-12 ***
## Married_Premium_Discount NA          NA          NA      
## Prior_Insurance_Premium_Adjustment NA          NA          NA      
## Premium_Adjustment_Region 0.98446   0.03179 30.966 < 2e-16 ***
## ---                        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 111.6 on 7962 degrees of freedom

```

```

## Multiple R-squared:  0.4112, Adjusted R-squared:  0.4103
## F-statistic: 427.8 on 13 and 7962 DF,  p-value: < 2.2e-16

summary(model_best_bwd)

##
## Call:
## lm(formula = Premium_Amount ~ Marital_Status + Prior_Insurance +
##     Claims_Frequency + Safe_Driver_Discount + Multi_Policy_Discount +
##     Source_of_Lead + Time_Since_First_Contact + Prior_Insurance_Premium_Adjustment +
##     Policy_Adjustment + Total_Discounts, data = train)
##
## Residuals:
##    Min      1Q  Median      3Q      Max 
## -165.09  -52.80   -4.60   47.78  392.71 
##
## Coefficients: (1 not defined because of singularities)
##                               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)                 2.306e+03  3.764e+00 612.836 < 2e-16 ***
## Marital_StatusMarried       8.549e+01  2.861e+00 29.884 < 2e-16 ***
## Marital_StatusSingle        -8.473e-01 2.968e+00 -0.285 0.77528  
## Marital_StatusWidowed      -2.286e+00 3.705e+00 -0.617 0.53722  
## Prior_Insurance>5 years    -9.865e+01 2.325e+00 -42.426 < 2e-16 ***
## Prior_Insurance1-5 years    -5.029e+01 2.043e+00 -24.611 < 2e-16 *** 
## Claims_Frequency            6.903e+01  1.114e+00 61.943 < 2e-16 *** 
## Safe_Driver_Discount        3.485e+00  3.353e+00  1.039 0.29867  
## Multi_Policy_Discount       3.268e+00  3.199e+00  1.021 0.30708  
## Source_of_LeadOnline         1.804e+00  1.777e+00  1.015 0.31002  
## Source_of_LeadReferral       -1.793e-01 2.934e+00 -0.061 0.95128  
## Time_Since_First_Contact    -2.426e-01 9.162e-02 -2.648 0.00811 ** 
## Prior_Insurance_Premium_Adjustment NA          NA          NA          NA      
## Policy_Adjustment            9.981e-01  8.135e-03 122.691 < 2e-16 *** 
## Total_Discounts              -1.041e+00 5.406e-02 -19.260 < 2e-16 *** 
## ---                        
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 71.12 on 7962 degrees of freedom
## Multiple R-squared:  0.7608, Adjusted R-squared:  0.7604
## F-statistic: 1948 on 13 and 7962 DF,  p-value: < 2.2e-16

summary(model_best_fwd)

##
## Call:
## lm(formula = Premium_Amount ~ Prior_Insurance + Claims_Severity +
##     Safe_Driver_Discount + Bundling_Discount + Conversion_Status +
##     Prior_Insurance_Premium_Adjustment + Premium_Adjustment_Region,
##     data = train)
##
## Residuals:
##    Min      1Q  Median      3Q      Max 
## -325.06  -98.64   4.40   90.40  452.83 
##

```

```

## Coefficients: (1 not defined because of singularities)
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                2274.93966   6.46278 352.006 < 2e-16 ***
## Prior_Insurance>5 years   -96.84576   4.32355 -22.400 < 2e-16 ***
## Prior_Insurance1-5 years  -56.93227   3.79918 -14.985 < 2e-16 ***
## Claims_SeverityLow        -46.80186   5.19137 -9.015 < 2e-16 ***
## Claims_SeverityMedium     -19.00982   5.88537 -3.230 0.00124 **
## Safe_Driver_Discount      -46.75111   3.72222 -12.560 < 2e-16 ***
## Bundling_Discount         -54.99499   5.02690 -10.940 < 2e-16 ***
## Conversion_Status          -22.56692   3.00150 -7.519 6.14e-14 ***
## Prior_Insurance_Premium_Adjustment NA          NA          NA          NA
## Premium_Adjustment_Region  0.96966    0.03769  25.725 < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 132.3 on 7967 degrees of freedom
## Multiple R-squared:  0.1715, Adjusted R-squared:  0.1707
## F-statistic: 206.2 on 8 and 7967 DF,  p-value: < 2.2e-16

calculate_metrics_test <- function(model, test_data) {

  predictions <- predict(model, newdata = test_data)

  # MAPE
  mape <- mean(abs((test_data$Premium_Amount - predictions) / test_data$Premium_Amount)) * 100

  # Adjusted R-squared
  n <- nrow(test_data) # Number of observations in the test data
  p <- length(coef(model)) - 1 # Number of predictors (excluding intercept)
  sse <- sum((test_data$Premium_Amount - predictions)^2)
  sst <- sum((test_data$Premium_Amount - mean(test_data$Premium_Amount))^2)
  adj_r_squared <- 1 - ((1 - (sse / sst)) * (n - 1) / (n - p - 1))

  # Mallows' Cp
  mse <- sse / (n - p)
  cp <- (sse / mse) - (n - 2 * p)

  # BIC (Bayesian Information Criterion)
  bic <- BIC(model)

  return(list(MAPE = mape, Adjusted_R_Squared = adj_r_squared, Mallows_Cp = cp, BIC = bic))
}

metrics_best_subset_test <- calculate_metrics_test(model_best_subset, test)
metrics_best_fwd_test <- calculate_metrics_test(model_best_fwd, test)
metrics_best_bwd_test <- calculate_metrics_test(model_best_bwd, test)

# Print
print("Metrics for Best Subset Model on Test Data:")

## [1] "Metrics for Best Subset Model on Test Data:"

```

```

print(metrics_best_subset_test)

## $MAPE
## [1] 4.405128
##
## $Adjusted_R_Squared
## [1] 0.5847184
##
## $Mallows_Cp
## [1] 15
##
## $BIC
## [1] 97964.73

print("Metrics for Best Forward Model on Test Data:")

## [1] "Metrics for Best Forward Model on Test Data:"

print(metrics_best_fwd_test)

## $MAPE
## [1] 4.893118
##
## $Adjusted_R_Squared
## [1] 0.7994499
##
## $Mallows_Cp
## [1] 9
##
## $BIC
## [1] 100644

print("Metrics for Best Backward Model on Test Data:")

## [1] "Metrics for Best Backward Model on Test Data:"

print(metrics_best_bwd_test)

## $MAPE
## [1] 2.445511
##
## $Adjusted_R_Squared
## [1] 0.2329139
##
## $Mallows_Cp
## [1] 14
##
## $BIC
## [1] 90781.2

```

Berdasarkan hasil pengujian model terhadap data *testing*, dapat disimpulkan bahwa model terbaik diperoleh dari metode *Stepwise Forward Selection*. Model ini menunjukkan kinerja yang unggul dengan nilai *adjusted R<sup>2</sup>* yang paling tinggi, nilai MAPE yang cukup kecil (sekitar 4%), nilai *Mallow's Cp* yang mendekati jumlah variabel ditambah satu (yaitu 8), serta nilai BIC yang relatif rendah, yang menunjukkan keseimbangan yang baik antara kompleksitas model dan kualitas prediksi.

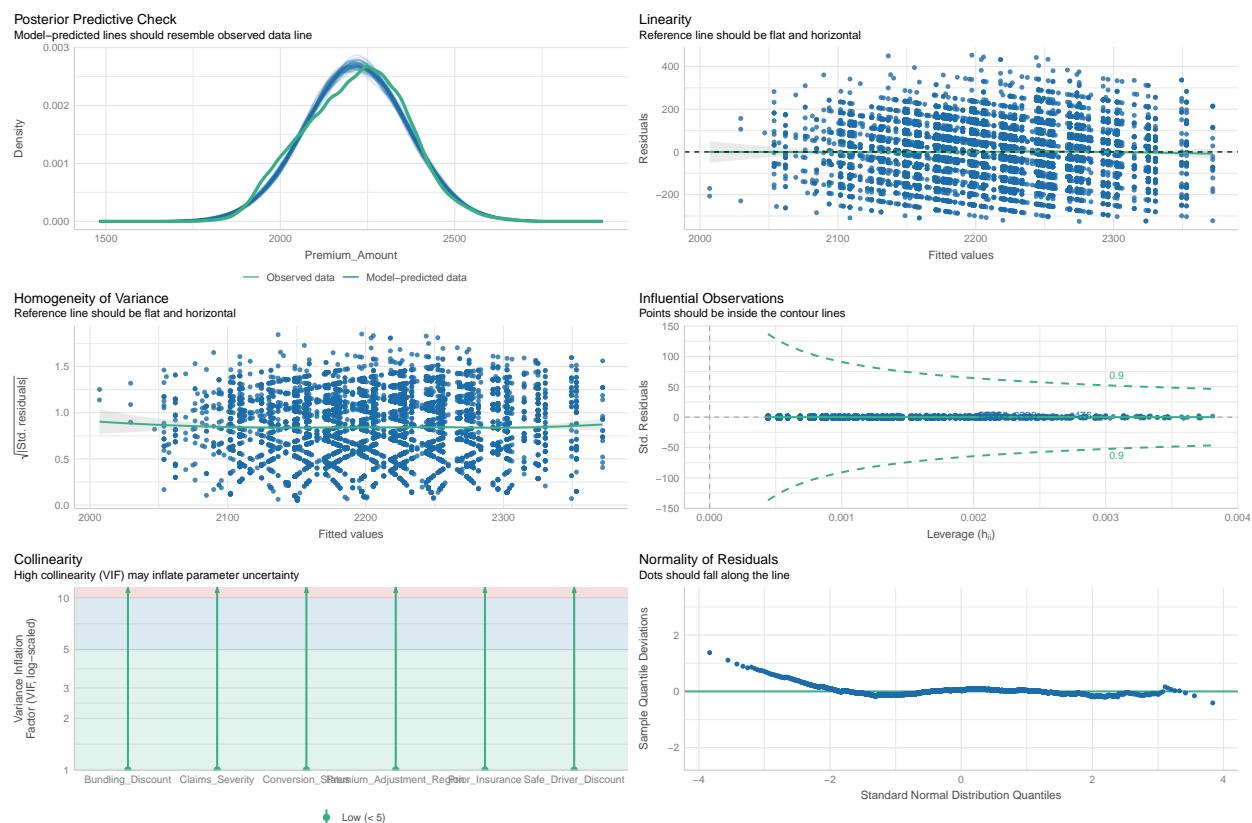
Selanjutnya, akan dilakukan evaluasi terhadap pemenuhan asumsi-asumsi dasar regresi linear pada model *forward selection* dengan 7 variabel prediktor. Pemeriksaan ini akan dilakukan menggunakan fungsi `check_model()` dari pustaka `see` dan `performance`.

Sebagai pengingat, asumsi-asumsi dasar dalam regresi linear meliputi:

1. Linearitas: Hubungan antara variabel prediktor dan respon bersifat linear.
2. Normalitas residual: Error model (residual) terdistribusi normal.
3. Homoskedastisitas: Varians error konstan pada seluruh nilai prediktor.
4. Multikolinearitas: Variabel prediktor tidak memiliki hubungan linear yang kuat satu sama lain.
5. Independensi observasi: Setiap observasi bersifat independen.

Perhatikan gambar di bawah,

```
check_model(model_best_fwd)
```



Perhatikan bahwa, pertama, asumsi linearitas terlihat pada gambar berjudul *Linearity*. Garis referensi di grafik tersebut seharusnya datar jika hubungan antara variabel prediktor dan Premium\_Amount benar-benar linear. Meskipun garisnya sedikit melengkung, secara umum pola titik-titik masih menyebar cukup merata, sehingga hubungan linear masih dapat diterima.

Kedua, asumsi normalitas residual dapat dilihat dari gambar *Normality of Residuals*. Pada gambar ini, titik-titik seharusnya mengikuti garis lurus jika sisa kesalahan model (residual) mengikuti distribusi normal. Namun, tampak bahwa titik-titik menyimpang dari garis, terutama di bagian ujung (ekor). Ini menunjukkan bahwa residual tidak sepenuhnya normal, sehingga asumsi ini kurang terpenuhi.

Ketiga, asumsi homoskedastisitas ditunjukkan oleh gambar *Homogeneity of Variance*. Dalam grafik ini, jika varians residual konstan, maka sebaran titik akan merata dan tidak membentuk pola tertentu. Namun pada gambar terlihat adanya pola melebar seiring naiknya fitted values, meskipun demikian pola titik-titik masih menyebar secara merata, sehingga asumsi ini bisa dianggap dipenuhi.

Keempat, asumsi tidak adanya multikolinearitas ditunjukkan oleh gambar *Collinearity*. Di sini, semua variabel memiliki nilai VIF di bawah 5, artinya tidak ada hubungan yang terlalu kuat antar variabel prediktor. Maka, asumsi ini dapat dikatakan sudah terpenuhi.

Kelima, asumsi independensi antar observasi tidak ditunjukkan secara langsung dalam gambar, tetapi karena data yang digunakan tidak berasal dari data berurutan seperti deret waktu atau data panel, maka asumsi ini diasumsikan terpenuhi.

Sehingga secara keseluruhan, model memenuhi sebagian besar asumsi dasar regresi linear, meskipun perlu perhatian lebih terhadap normalitas dan kesamaan varians residual.

## Generalized Linear Model

Pada bagian ini, akan dibuat *Generalized Linear Model* (GLM) untuk memprediksi `Premium_Amount`. Di bagian ini juga akan dibandinkan performa dari model GLM tipe Poisson dan Gamma dalam memprediksi `Premium_Amount`

### Regresi Poisson

Pertama, akan dibentuk sebuah model regresi Poisson dengan link functionnya yaitu ‘log’. Pemilihan variabel bebas untuk regresi Poisson di bawah akan menggunakan metode stepwise dengan parameter *direction* adalah *both* yang berarti bahwa model bisa ditambahkan ataupun dikurangi variabel bebas pada setiap langkahnya.

```
pois_model=glm(Premium_Amount~.,data=train ,family=poisson(link='log'))
```

```
summary(step_poisson)
```

```
##  
## Call:  
## glm(formula = Premium_Amount ~ Marital_Status + Prior_Insurance +  
##       Claims_Frequency + Claims_Adjustment + Policy_Type + Safe_Driver_Discount +  
##       Multi_Policy_Discount + Bundling_Discount + Premium_Adjustment_Credit +  
##       Region, family = poisson(link = "log"), data = train)  
##  
## Coefficients:  
##                               Estimate Std. Error z value Pr(>|z|)  
## (Intercept)                7.717e+00  1.081e-03 7138.268  <2e-16 ***  
## Marital_StatusMarried     3.898e-02  8.605e-04   45.303  <2e-16 ***  
## Marital_StatusSingle      1.070e-04  8.953e-04    0.119   0.905  
## Marital_StatusWidowed     1.386e-04  1.118e-03    0.124   0.901  
## Prior_Insurance>5 years -4.500e-02  6.928e-04  -64.954  <2e-16 ***  
## Prior_Insurance1-5 years -2.227e-02  6.048e-04  -36.816  <2e-16 ***  
## Claims_Frequency          8.415e-04  5.665e-04    1.485   0.137
```

```

## Claims_Adjustment      4.308e-04  6.581e-06   65.459 <2e-16 ***
## Policy_TypeLiability-Only -9.120e-02  4.908e-04  -185.816 <2e-16 ***
## Safe_Driver_Discount    -2.271e-02  6.013e-04  -37.771 <2e-16 ***
## Multi_Policy_Discount   -2.274e-02  5.178e-04  -43.910 <2e-16 ***
## Bundling_Discount       -2.286e-02  8.157e-04  -28.031 <2e-16 ***
## Premium_Adjustment_Credit 4.498e-04  4.866e-06   92.433 <2e-16 ***
## RegionSuburban          2.298e-02  6.839e-04   33.605 <2e-16 ***
## RegionUrban              4.547e-02  6.269e-04   72.529 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
## Null deviance: 76102.79  on 7975  degrees of freedom
## Residual deviance: 117.98  on 7961  degrees of freedom
## AIC: 76238
##
## Number of Fisher Scoring iterations: 3

predictions_step_test = predict(step_poisson, newdata = test, type = "response")
residuals_step_test = test$Premium_Amount - predictions_step_test
rmse_step_test = sqrt(mean(residuals_step_test^2))
print(rmse_step_test)

```

```
## [1] 5.727907
```

Nilai RMSE untuk data testing untuk model regresi Poisson adalah 5.7279

## Regresi Gamma

Selanjutnya akan dibangun sebuah model regresi Gamma dengan *link function* yaitu *inverse*. Pemilihan variabel bebas juga akan menggunakan stepwise dengan parameter *direction both*.

```
gamma_model=glm(Premium_Amount~.,data=train ,family=Gamma(link='inverse'))
```

```
summary(step_gamma)
```

```

##
## Call:
## glm(formula = Premium_Amount ~ Is_Senior + Marital_Status + Prior_Insurance +
##     Claims_Frequency + Claims_Adjustment + Policy_Type + Safe_Driver_Discount +
##     Multi_Policy_Discount + Bundling_Discount + Conversion_Status +
##     Inquiries + Time_to_Conversion + Premium_Adjustment_Credit +
##     Region, family = Gamma(link = "inverse"), data = train)
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)           4.446e-04  8.594e-07 517.384 <2e-16 ***
## Is_Senior             1.095e-07  7.141e-08   1.533  0.1254
## Marital_StatusMarried -1.760e-05  9.481e-08  -185.669 <2e-16 ***
## Marital_StatusSingle -9.795e-08  9.892e-08   -0.990  0.3221

```

```

## Marital_StatusWidowed      -1.184e-07  1.235e-07  -0.958  0.3379
## Prior_Insurance>5 years   2.016e-05  7.573e-08  266.250 <2e-16 ***
## Prior_Insurance1-5 years   9.868e-06  6.558e-08  150.483 <2e-16 ***
## Claims_Frequency          -7.140e-07  6.043e-08  -11.816 <2e-16 ***
## Claims_Adjustment          -1.852e-07  6.897e-10  -268.484 <2e-16 ***
## Policy_TypeLiability-Only 4.151e-05  5.450e-08  761.570 <2e-16 ***
## Safe_Driver_Discount       1.027e-05  6.618e-08  155.221 <2e-16 ***
## Multi_Policy_Discount      1.029e-05  5.686e-08  181.060 <2e-16 ***
## Bundling_Discount          1.040e-05  9.019e-08  115.351 <2e-16 ***
## Conversion_Status          1.387e-06  7.919e-07   1.752  0.0799 .
## Inquiries                  2.914e-08  1.829e-08   1.593  0.1111
## Time_to_Conversion         1.353e-08  8.601e-09   1.573  0.1157
## Premium_Adjustment_Credit -2.015e-07  5.299e-10  -380.318 <2e-16 ***
## RegionSuburban              -1.052e-05  7.553e-08  -139.259 <2e-16 ***
## RegionUrban                 -2.059e-05  6.909e-08  -298.063 <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for Gamma family taken to be 2.661829e-05)
##
## Null deviance: 34.46702  on 7975  degrees of freedom
## Residual deviance:  0.21304  on 7957  degrees of freedom
## AIC: 61546
##
## Number of Fisher Scoring iterations: 3

predictions_step_test = predict(step_gamma, newdata = test, type = "response")
residuals_step_test = test$Premium_Amount - predictions_step_test
rmse_step_test = sqrt(mean(residuals_step_test^2))
print(rmse_step_test)

```

```
## [1] 11.53539
```

Nilai RMSE untuk model regresi Gamma adalah sebesar 11.5353. Berdasarkan hasil ini, model regresi Poisson memberikan kinerja yang lebih baik dalam memprediksi `Premium_Amount`. Salah satu alasan yang mungkin adalah karena `Premium_Amount` pada data asli berbentuk bilangan bulat, sehingga lebih sesuai dimodelkan sebagai data cacah menggunakan regresi Poisson.

## Cross Validation (Soal 4 - b)

Pada bagian ini, digunakan fungsi `cv.glmnet` dari pustaka `glmnet` untuk melakukan *cross-validation* pada model regresi linear berganda dengan teknik regularisasi *ridge*, *lasso*, dan *elastic net*. Tujuan dari penerapan *cross-validation* dan regularisasi ini adalah untuk menghindari terjadinya *overfitting* pada data pelatihan, sehingga model yang dihasilkan dapat melakukan generalisasi dengan lebih baik saat diterapkan pada data baru.

### Ridge Cross Validation

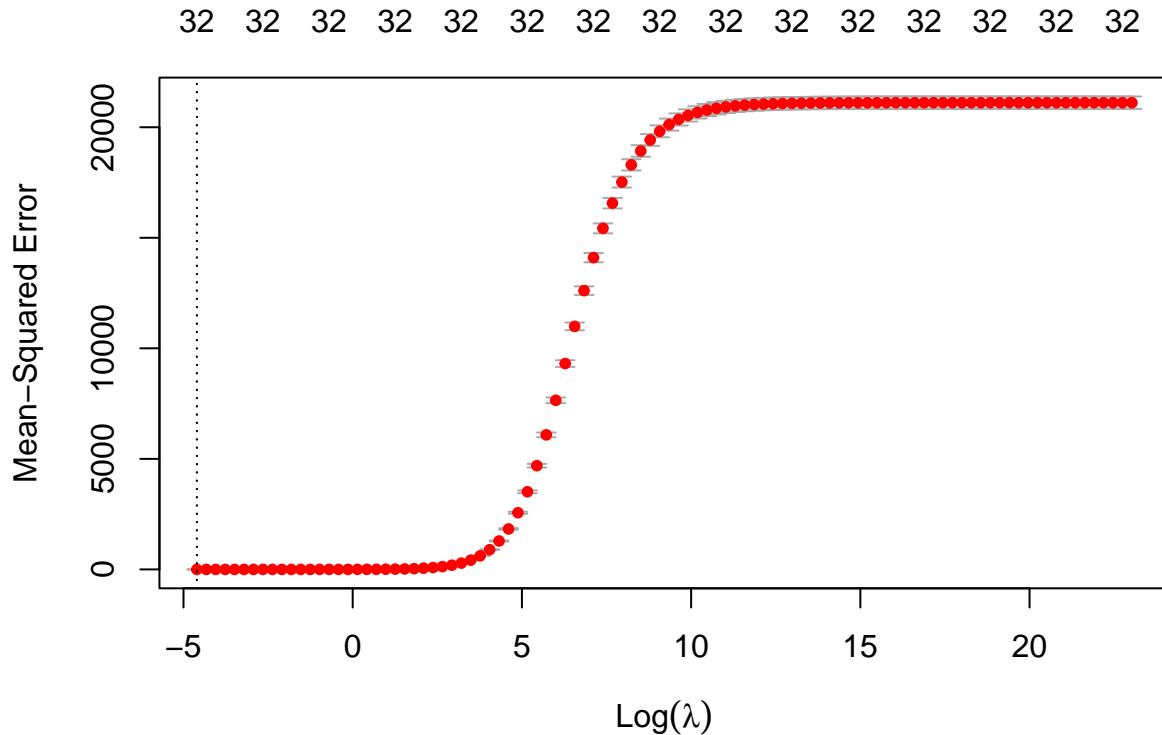
Pada bagian ini akan digunakan teknik regularisasi *ridge* pada model regresi linear berganda. Teknik regularisasi *ridge* akan memberikan penalti sebesar  $\lambda$  terhadap sebuah parameter jika nilai parameter tersebut terlalu besar, akibatnya nilai dari parameter yang tidak signifikan akan menjadi sangat kecil hingga mendekati

0. Untuk menerapkan teknik regularisasi *ridge*, pertama data akan dibentuk sebagai sebuah matriks karena fungsi `cv.glmnet` hanya bisa menerima parameter berupa matriks. Setelah itu akan dilakukan latih dengan jumlah parameter *foldnya* adalah 5.

```
x_train = model.matrix(Premium_Amount~., data=train) [,-1]
y_train = train$Premium_Amount
x_test = model.matrix(Premium_Amount~., data=test) [,-1]
y_test = test$Premium_Amount
grid = 10^seq(-2,10, length=100)

cv_ridge = cv.glmnet(x_train,y_train, alpha=0, nfolds=5, lambda=grid, type.measure='mse')

plot(cv_ridge)
```



Gambar di atas menunjukkan nilai MSE dari masing-masing model regresi linear berganda dengan teknik regularisasi *ridge* pada  $\lambda$  yang berbeda-beda.

```
lambda_ridge = cv_ridge$lambda.min
rmse_ridge = min(sqrt(cv_ridge$cvm))
print(paste('Nilai Lambda Terbaik:', lambda_ridge))

## [1] "Nilai Lambda Terbaik: 0.01"
```

```
print(paste('Nilai RMSE Terbaik:', rmse_ridge))
```

```
## [1] "Nilai RMSE Terbaik: 0.0229847058139809"
```

Berdasarkan hasil di atas nilai lambda terbaik berada  $\lambda = 0.01$  dengan nilai RMSEnya pada data latih adalah 0.02298 .

```
ridge = glmnet(x_train, y_train, alpha=0, lambda=grid)
ridge_pred = predict(ridge, s=lambda_ridge, newx = x_test)
rmse_pred = sqrt(mean(ridge_pred - y_test)^2)
print(paste('Nilai RMSE data testing',rmse_pred))
```

```
## [1] "Nilai RMSE data testing 0.00100835562350215"
```

Nilai RMSE pada data uji dari model regresi linear berganda dengan teknik regularisasi *ridge* adalah 0.0010 yang lebih kecil dari nilai RMSE dari data testing. Hal ini tentu masuk akal karena tujuan dari teknik regularisasi adalah memastikan bahwa model bisa menyesuaikan dengan lebih baik pada data uji.

```
best_coef = coef(ridge, s = lambda_ridge)
print(best_coef)
```

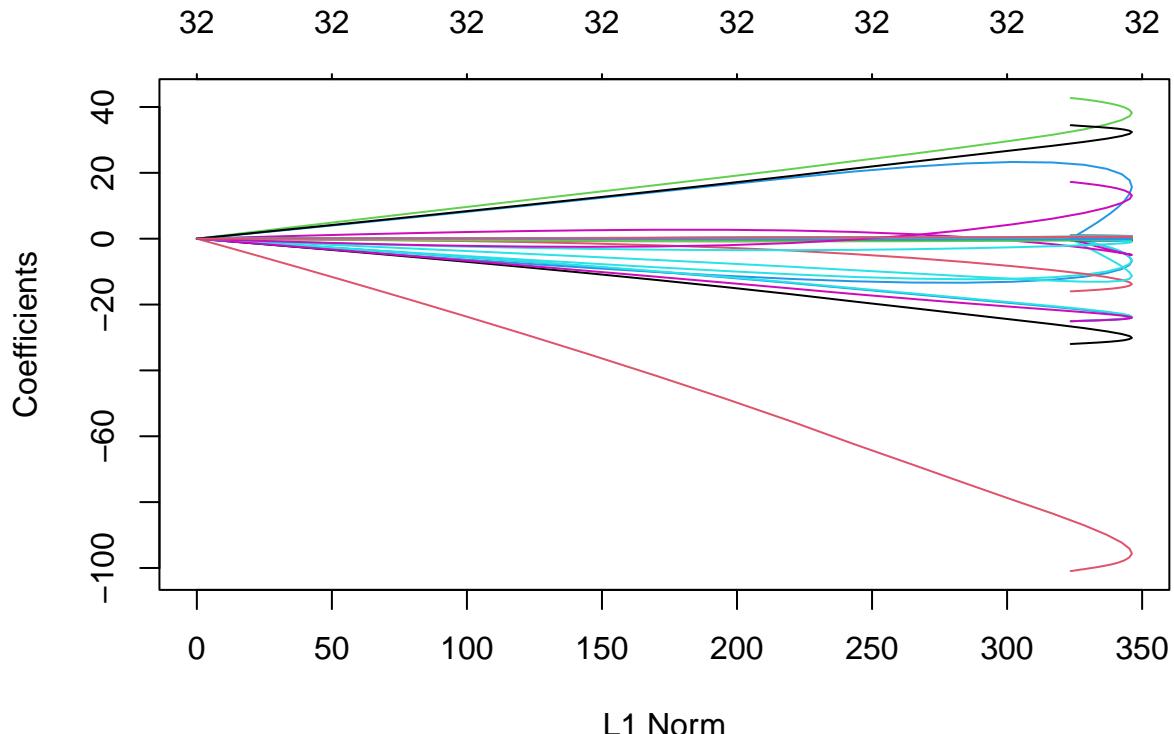
```
## 33 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)          2.182302e+03
## Age                  -8.950218e-05
## Is_Senior            1.571922e-03
## Marital_StatusMarried 4.270101e+01
## Marital_StatusSingle -8.961549e-03
## Marital_StatusWidowed -8.039121e-03
## Married_Premium_Discount 5.033476e-01
## Prior_Insurance>5 years -3.196806e+01
## Prior_Insurance1-5 years -1.598307e+01
## Prior_Insurance_Premium_Adjustment 6.802697e-01
## Claims_Frequency      5.407844e-02
## Claims_SeverityLow    -5.736948e-02
## Claims_SeverityMedium -3.689824e-02
## Claims_Adjustment     9.993223e-01
## Policy_TypeLiability-Only -1.009136e+02
## Policy_Adjustment     4.953968e-01
## Safe_Driver_Discount -2.505670e+01
## Multi_Policy_Discount -2.505614e+01
## Bundling_Discount     -2.505554e+01
## Total_Discounts       -4.988471e-01
## Source_of_LeadOnline   6.435192e-04
## Source_of_LeadReferral -6.798269e-04
## Time_Since_First_Contact -3.698258e-05
## Conversion_Status     -1.214237e-01
## Website_Visits        -4.114414e-05
## Inquiries              -2.175109e-04
## Quotes_Requested       2.816285e-04
## Time_to_Conversion    -1.301945e-03
```

```

## Credit_Score           -2.042627e-04
## Premium_Adjustment_Credit 9.997598e-01
## RegionSuburban        1.722858e+01
## RegionUrban            3.446351e+01
## Premium_Adjustment_Region 6.553084e-01

```

```
plot(ridge)
```



## Lasso Regularization

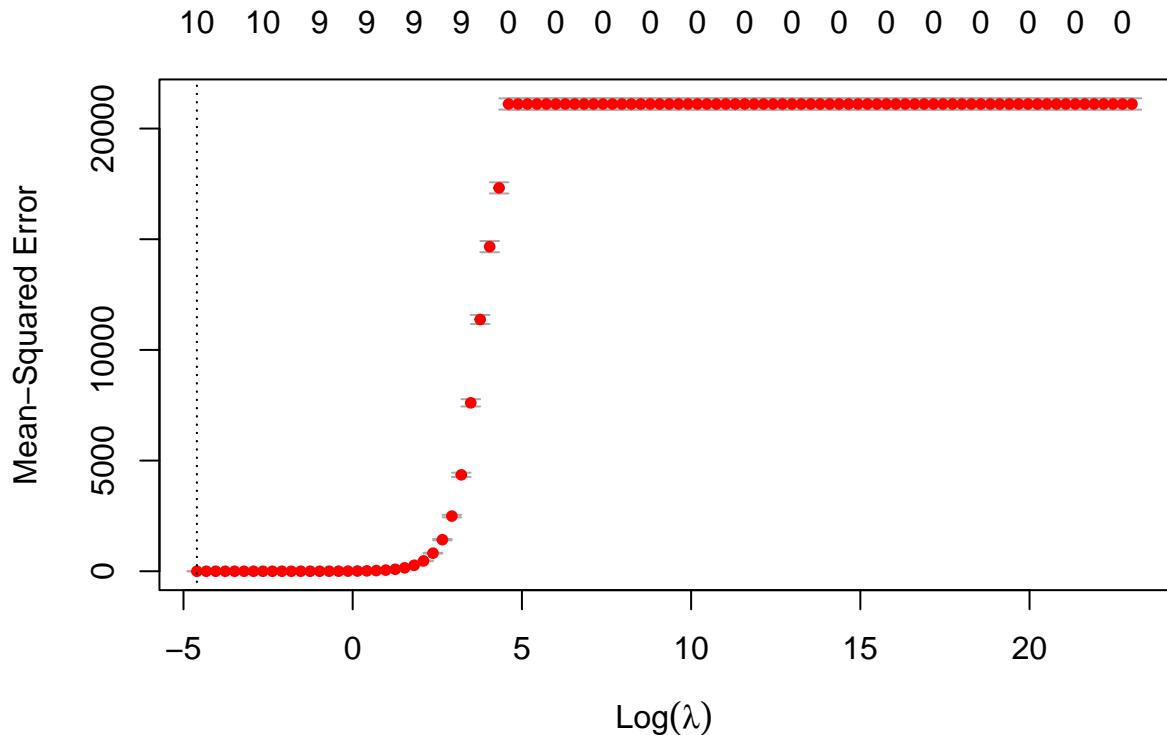
Pada bagian ini akan diterapkan teknik regularisasi *lasso* pada model regresi linear berganda. Sama seperti Ridge, tujuan utama dari *lasso* adalah untuk mencegah *overfitting* dan meningkatkan kemampuan generalisasi model. Namun, keunggulan *lasso* terletak pada kemampuannya untuk menyusutkan koefisien variabel yang kurang penting hingga menjadi nol, sehingga secara otomatis melakukan seleksi variabel dalam proses pemodelan.

```

cv_lasso = cv.glmnet(x_train,y_train,alpha=1,
,nfolds=5,lambda = grid,type.measure='mse')

```

```
plot(cv_lasso)
```



```
lambda_lasso = cv_lasso$lambda.min
rmse_lasso = min(sqrt(cv_lasso$cvm))
print(paste('Nilai Lambda Terbaik:', lambda_lasso))
```

```
## [1] "Nilai Lambda Terbaik: 0.01"
```

```
print(paste('Nilai RMSE Terbaik:', rmse_lasso))
```

```
## [1] "Nilai RMSE Terbaik: 0.0267151060802311"
```

Nilai lambda terbaik untuk teknik regularisasi *lasso* adalah 0.01 dengan nilai RMSEnya pada data latih berupa 0.02672.

```
lasso = glmnet(x_train, y_train, alpha=1, lambda=grid)
lasso_pred = predict(lasso , s=lambda_lasso, newx = x_test)
rmse_pred = sqrt(mean(lasso_pred - y_test)^2)
print(rmse_pred)
```

```
## [1] 0.000490744
```

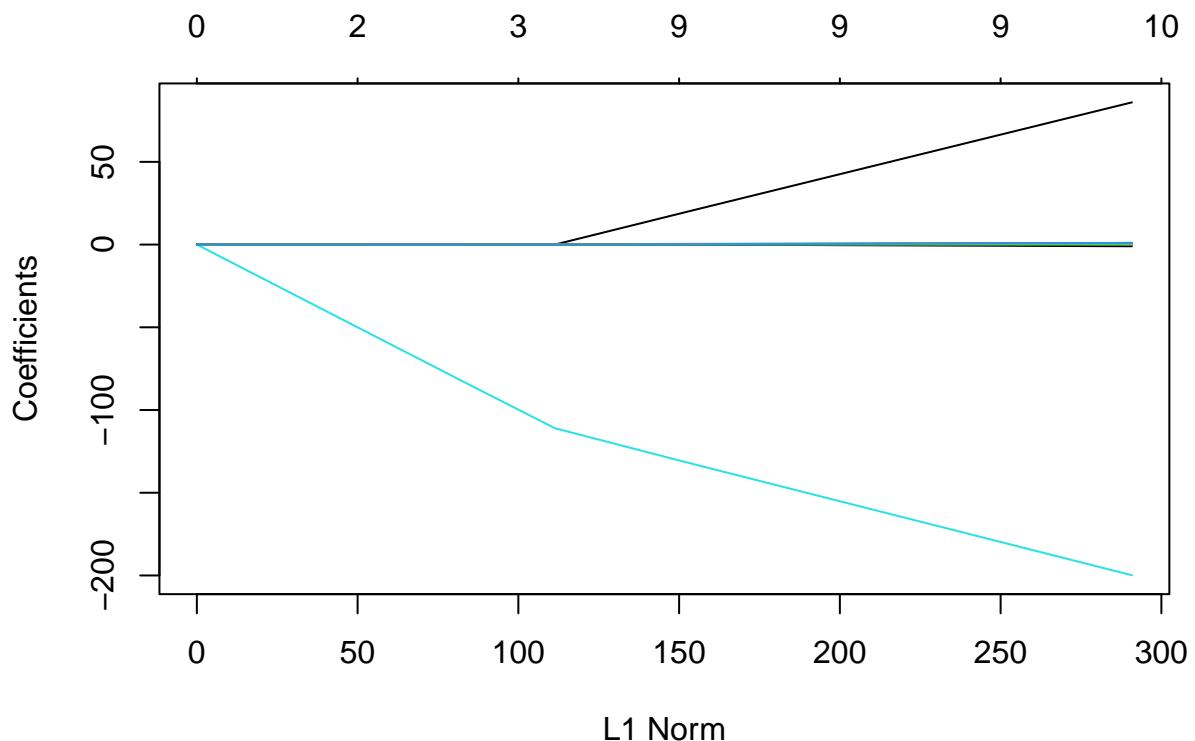
Nilai RMSE dari model regresi linear berganda dengan teknik regularisasi *lasso* adalah 0.0004907.

```
best_coef = coef(lasso, s = lambda_lasso)
lasso.coef = predict(lasso, type='coefficients', s= lambda_lasso)
print(best_coef)
```

```
## 33 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)          2.150028e+03
## Age                  .
## Is_Senior             .
## Marital_StatusMarried 8.598004e+01
## Marital_StatusSingle   .
## Marital_StatusWidowed   .
## Married_Premium_Discount 2.916038e-12
## Prior_Insurance>5 years   .
## Prior_Insurance1-5 years   .
## Prior_Insurance_Premium_Adjustment 9.997058e-01
## Claims_Frequency       .
## Claims_SeverityLow      .
## Claims_SeverityMedium    .
## Claims_Adjustment        9.998288e-01
## Policy_TypeLiability-Only -1.998624e+02
## Policy_Adjustment        5.848025e-04
## Safe_Driver_Discount     .
## Multi_Policy_Discount     .
## Bundling_Discount        .
## Total_Discounts          -9.996988e-01
## Source_of_LeadOnline      .
## Source_of_LeadReferral     .
## Time_Since_First_Contact    .
## Conversion_Status         .
## Website_Visits            .
## Inquiries                 .
## Quotes_Requested           .
## Time_to_Conversion         .
## Credit_Score                .
## Premium_Adjustment_Credit 9.997893e-01
## RegionSuburban              .
## RegionUrban                 7.717404e-03
## Premium_Adjustment_Region 9.996577e-01
```

```
plot(lasso)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



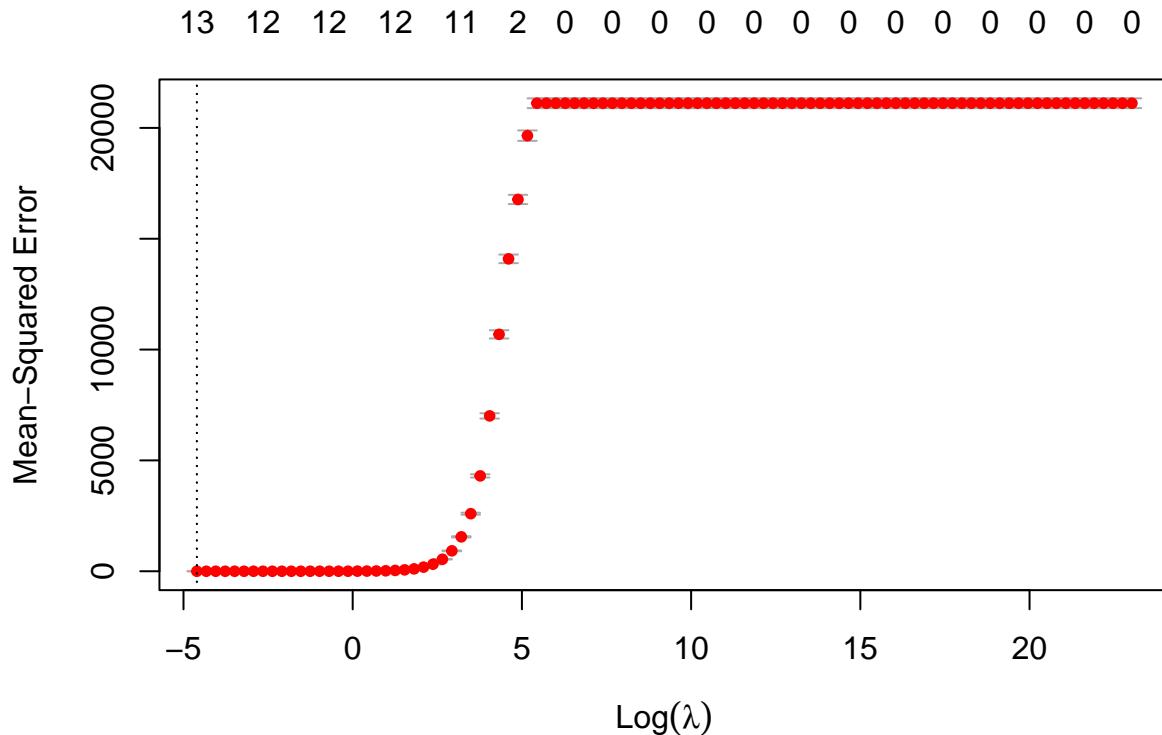
Perhatikan bahwa hasil di atas menunjukkan dari 27 variabel yang ada untuk memprediksi Premium\_Amount hanya ada sekitar 13 variabel yang signifikan dengan variabel lainnya dianggap tidak signifikan karena nilai dari parameternya adalah 0.

### Elastic Net Regularization

Pada bagian ini akan diterapkan teknik regularisasi *elastic net* pada model regresi linear berganda. *Elastic net* merupakan kasus gabungan dan juga perumuman dari teknik regularisasi *ridge* dan *lasso*. Teknik regularisasi ini menggabungkan *ridge* dan *lasso* dengan harapan bahwa variabel yang tidak signifikan masih akan diberi nilai 0.

```
cv_el = cv.glmnet(x_train,y_train,alpha=0.5,nfolds=5,lambda = grid, type.measure='mse')
```

```
plot(cv_el)
```



```
lambda_el = cv_el$lambda.min
rmse_el = min(sqrt(cv_el$cvm))
print(paste('Nilai Lambda Terbaik:', lambda_el))
```

```
## [1] "Nilai Lambda Terbaik: 0.01"
```

```
print(paste('Nilai RMSE Terbaik:', rmse_el))
```

```
## [1] "Nilai RMSE Terbaik: 0.0195245534127135"
```

Nilai lambda terbaik untuk model regresi linear berganda dengan teknik regularisasi *elastic net* adalah 0.01 dengan nilai RMSEnya untuk data latih adalah 0.0195.

```
el = glmnet(x_train, y_train, alpha=0.5, lambda=grid)
el_pred = predict(el, s=lambda_el, newx = x_test)
rmse_pred = sqrt(mean(el_pred - y_test)^2)
print(rmse_pred)
```

```
## [1] 0.0005863242
```

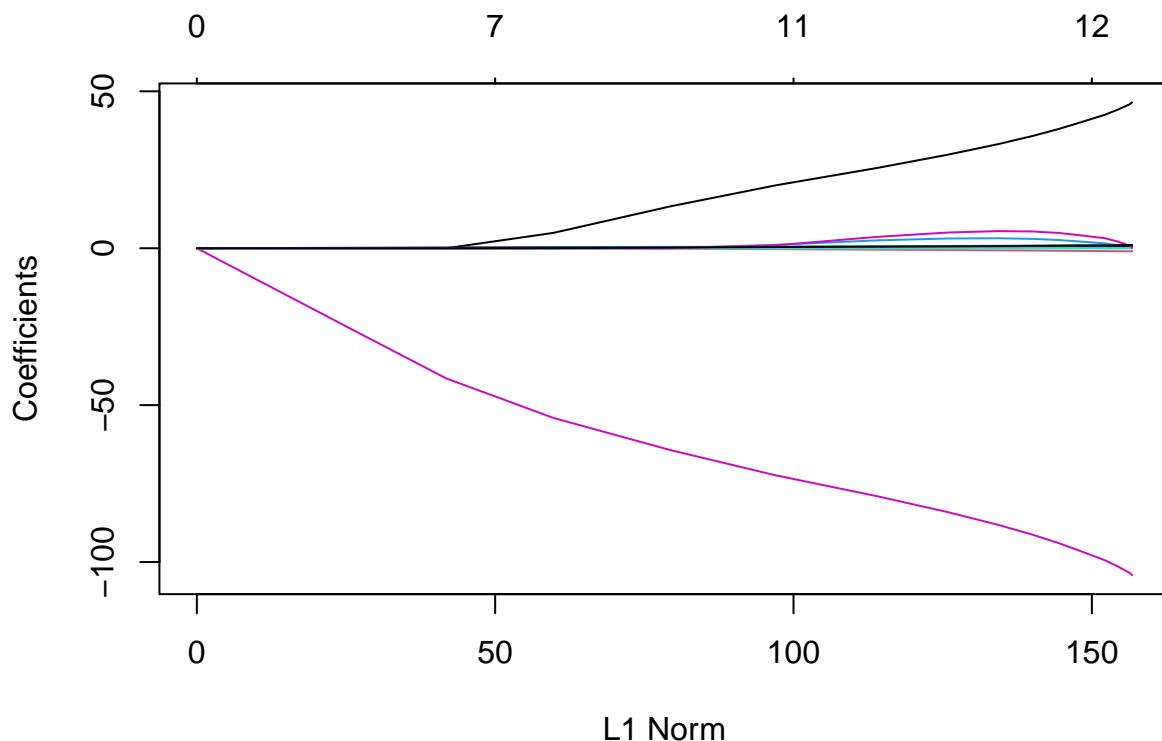
Nilai RMSE dari model regresi linear berganda dengan teknik regularisasi *elastic net* pada data uji adalah 0.00058

```
best_coef = coef(el, s = lambda_el)
print(best_coef)
```

```
## 33 x 1 sparse Matrix of class "dgCMatrix"
##                                     s1
## (Intercept)          2.150039e+03
## Age                  .
## Is_Senior             .
## Marital_StatusMarried 4.648079e+01
## Marital_StatusSingle   .
## Marital_StatusWidowed   .
## Married_Premium_Discount 4.593918e-01
## Prior_Insurance>5 years   .
## Prior_Insurance1-5 years   .
## Prior_Insurance_Premium_Adjustment 9.998214e-01
## Claims_Frequency        1.065703e-02
## Claims_SeverityLow      .
## Claims_SeverityMedium    .
## Claims_Adjustment        9.997746e-01
## Policy_TypeLiability-Only -1.042004e+02
## Policy_Adjustment        4.789285e-01
## Safe_Driver_Discount     .
## Multi_Policy_Discount    .
## Bundling_Discount        .
## Total_Discounts          -9.998142e-01
## Source_of_LeadOnline      .
## Source_of_LeadReferral    .
## Time_Since_First_Contact   .
## Conversion_Status         .
## Website_Visits            .
## Inquiries                 .
## Quotes_Requested           .
## Time_to_Conversion         .
## Credit_Score              -2.174431e-05
## Premium_Adjustment_Credit 9.998386e-01
## RegionSuburban            1.593565e-02
## RegionUrban                6.897796e-02
## Premium_Adjustment_Region 9.990975e-01
```

```
plot(el)
```

```
## Warning in regularize.values(x, y, ties, missing(ties), na.rm = na.rm):
## collapsing to unique 'x' values
```



Dari 27 variabel yang ada, model regresi linear berganda dengan teknik regularisasi *elastic net* memilih 14 variabel dengan pengaruh yang signifikan dengan 13 variabel dianggap tidak signifikan karena memiliki nilai parameter 0.

### Model terbaik berdasarkan `cv.glmnet` (Soal 4 - b)

Secara keseluruhan model regresi linear berganda dengan teknik regularisasi *lasso* memberikan performa yang paling baik dengan nilai RMSEnya untuk data testing yang paling rendah, yaitu 0.0004907. Dari model tersebut variabel yang berpengaruh signifikan atau bisa dibilang terpilih ke dalam model meliputi `Marital_Status_Married`, `Married_Premium_Discount`, `Prior_Insurance_Premium_Adjustment`, `Claims__Adjustment`, `Policy_TypeLiability-Only`, `Policy_Adjustment`, dan `Total_Discounts`.

`## Interpretasi Model (Soal 4 - c)`

Pada poin (a), kita sudah berhasil membangun sebuah model menggunakan *stepwise* dan menentukan bahwa model dengan stepwise *forward selection* merupakan model yang paling baik dengan modelnya adalah

$$\begin{aligned}
\widehat{\text{Premium\_Amount}} = & 2274.94 \\
& + (-96.85) \cdot \text{Prior\_Insurance}>5\_years \\
& + (-56.93) \cdot \text{Prior\_Insurance1-5\_years} \\
& + (-46.80) \cdot \text{Claims\_Severity\_Low} \\
& + (-19.01) \cdot \text{Claims\_Severity\_Medium} \\
& + (-46.75) \cdot \text{Safe\_Driver\_Discount} \\
& + (-54.99) \cdot \text{Bundling\_Discount} \\
& + (-22.57) \cdot \text{Conversion\_Status} \\
& + 0.97 \cdot \text{Premium\_Adjustment\_Region}
\end{aligned}$$

Dari model tersebut kita dapat menginterpretasikan bahwa,

1. Model memiliki nilai **intercept** yang tinggi yang mengindikasikan bahwa sebelum dipengaruhi oleh variabel apapun nilai premi adalah 2274.94
2. Koefisien untuk **Prior\_Insurance>5 years** adalah  $-96.85$ , yang menunjukkan bahwa jika seseorang memiliki asuransi sebelumnya lebih dari 5 tahun, nilai premi akan lebih rendah sebesar  $96.85$  dibandingkan dengan orang yang tidak memiliki asuransi sebelumnya
3. Koefisien untuk **Prior\_Insurance1-5 years** adalah  $-56.93$ , yang berarti bahwa jika seseorang memiliki asuransi sebelumnya selama 1-5 tahun, nilai premi mereka akan lebih rendah sebesar  $56.93$  dibandingkan dengan orang yang tidak memiliki asuransi sebelumnya.
4. Koefisien untuk **Claims\_SeverityLow** adalah  $-46.80$ , yang berarti bahwa jika tingkat keparahan klaim dikategorikan sebagai Low (rendah), maka nilai premi akan lebih rendah sebesar  $46.80$  dibandingkan dengan mereka yang memiliki tingkat keparahan klaim yang lebih tinggi.
5. Koefisien untuk **Claims\_SeverityMedium** adalah  $-19.01$ , yang menunjukkan bahwa jika tingkat keparahan klaim dikategorikan sebagai Medium (sedang), nilai premi akan lebih rendah sebesar  $19.01$  dibandingkan dengan pelanggan yang memiliki klaim dengan tingkat keparahan tinggi.
6. Koefisien untuk **Safe\_Driver\_Discount** adalah  $-46.75$ , yang berarti bahwa pelanggan yang memiliki historis mengemudi dengan lebih aman akan membayar premi yang lebih rendah sebesar  $46.75$ .
7. Koefisien untuk **Bundling\_Discount** adalah  $-54.99$ , yang menunjukkan bahwa pelanggan yang mendapatkan diskon bundling (misalnya, menggabungkan beberapa polis asuransi) akan membayar premi yang lebih rendah sebesar  $54.99$ .
8. Koefisien untuk **Conversion\_Status** adalah  $-22.57$ , yang berarti bahwa jika status konversi pelanggan adalah positif (misalnya, mereka beralih dari calon pelanggan menjadi pelanggan yang aktif), nilai premi akan lebih rendah sebesar  $22.57$ .
9. Koefisien untuk **Premium\_Adjustment\_Region** adalah  $0.97$ , yang menunjukkan bahwa setiap perubahan satu unit dalam variabel **Premium\_Adjustment\_Region** (misalnya perubahan dalam lokasi atau wilayah) akan mengubah nilai premi sebesar  $0.97$ .

Pada poin (b) telah ditentukan juga bahwa model regresi linear berganda dengan regularisasi *lasso* merupakan model terbaik dengan modelnya adalah

$$\begin{aligned}
\widehat{\text{Premium\_Amount}} = & 2150.03 \\
& + 85.98 \cdot \text{Marital\_Status\_Married} \\
& + 2.92 \times 10^{-12} \cdot \text{Married\_Premium\_Discount} \\
& + 0.9997 \cdot \text{Prior\_Insurance\_Premium\_Adjustment} \\
& + 0.9998 \cdot \text{Claims\_Adjustment} \\
& - 199.86 \cdot \text{Policy\_Type\_LiabilityOnly} \\
& + 5.85 \times 10^{-4} \cdot \text{Policy\_Adjustment} \\
& - 0.9997 \cdot \text{Total\_Discounts} \\
& + 0.9998 \cdot \text{Premium\_Adjustment\_Credit} \\
& + 0.0077 \cdot \text{Region\_Urban} \\
& + 0.9997 \cdot \text{Premium\_Adjustment\_Region}
\end{aligned}$$

Dari model tersebut kita dapat menginterpretasikan bahwa:

1. Model memiliki **intercept** sebesar 2150.03, yang menunjukkan bahwa jika semua variabel prediktor bernilai nol, maka nilai awal dari premi yang diprediksi adalah 2150.03.
2. **Marital\_StatusMarried** memiliki koefisien sebesar 85.98, yang berarti bahwa pelanggan yang sudah menikah cenderung membayar premi lebih tinggi sebesar 85.98 dibandingkan dengan kategori dasar (baseline) status pernikahan.
3. **Married\_Premium\_Discount** memiliki pengaruh sangat kecil sebesar  $2.92 \times 10^{-12}$ , yang secara praktis dapat dianggap tidak berdampak signifikan terhadap nilai premi.
4. **Prior\_Insurance\_Premium\_Adjustment** berkontribusi sebesar 0.9997, menunjukkan bahwa setiap kenaikan satu unit dalam penyesuaian premi berdasarkan asuransi sebelumnya akan meningkatkan premi sebesar 0.9997.
5. **Claims\_Adjustment** memiliki koefisien sebesar 0.9998, menunjukkan bahwa setiap satu unit perubahan dalam penyesuaian klaim akan meningkatkan premi sebesar 0.9998.
6. **Policy\_TypeLiability-Only** memiliki koefisien negatif sebesar -199.86, yang berarti bahwa pelanggan dengan polis tipe "Liability-Only" membayar premi yang lebih rendah sebesar 199.86 dibandingkan dengan tipe polis lainnya.
7. **Policy\_Adjustment** memiliki dampak kecil sebesar 0.000585, artinya setiap satu unit perubahan dalam penyesuaian polis meningkatkan premi sebesar angka yang sangat kecil ini.
8. **Total\_Discounts** memiliki koefisien negatif -0.9997, yang berarti bahwa setiap kenaikan satu unit dalam total diskon akan menurunkan premi sebesar 0.9997.
9. **Premium\_Adjustment\_Credit** memiliki nilai koefisien sebesar 0.9998, yang berarti peningkatan satu unit dalam penyesuaian premi berdasarkan skor kredit akan meningkatkan premi sebesar 0.9998.
10. **RegionUrban** memiliki koefisien sebesar 0.0077, yang berarti bahwa tinggal di daerah urban akan menambah premi sekitar 0.0077 dibandingkan wilayah referensi (misalnya rural), namun dampaknya sangat kecil.
11. **Premium\_Adjustment\_Region** memiliki koefisien 0.9997, yang berarti setiap kenaikan satu unit pada penyesuaian premi berbasis wilayah akan meningkatkan premi sebesar 0.9997.

## Penjelasan ke Divisi Finance (Soal 4 - d)

### 1. Tujuan

Membuat model untuk **memperkirakan besarnya premi asuransi** yang akan dibayar oleh pelanggan berdasarkan informasi dan karakteristik mereka, seperti lama punya asuransi sebelumnya, status pernikahan, jenis polis, dan sebagainya. Tujuan utamanya adalah untuk membuat prediksi dan mengetahui karakteristik ataupun profil-profil nasabah yang membayar premi mahal ataupun murah.

### 2. Pendekatan Model Terbaik

- **Model Stepwise (Forward Selection):**

Model tipe ini memulai dengan model yang paling sederhana kemudian menambah faktor-faktor lain yang mungkin dapat mempengaruhi jumlah premi.

Interpretasi :

Secara sederhana, model ini mengatakan:

Premi dasar seseorang dimulai dari \$2,274.94.

Kemudian, premi itu akan turun atau naik tergantung karakteristik pelanggan, seperti:

- Pernah punya asuransi sebelumnya selama lebih dari 5 tahun : premi turun sekitar \$96.85.
- Pernah punya asuransi 1–5 tahun : premi turun \$56.93.
- Klaim yang dibuat tergolong ringan (*low severity*) : premi turun \$46.80.
- Klaim tergolong sedang (i) : premi turun \$19.01.
- Pelanggan punya riwayat mengemudi aman (dapat diskon aman berkendara) : premi turun \$46.75.
- Menggabungkan beberapa produk asuransi (*bundling*) : premi turun \$54.99.
- Statusnya sudah berubah dari calon pelanggan menjadi pelanggan aktif →: premi turun \$22.57.
- Tinggal di wilayah tertentu (*Premium\_Adjustment\_Region*) : setiap perubahan di wilayah akan menambah atau mengurangi premi sekitar 0.97.

- **Model Lasso Regression (dengan Regularisasi):**

Model ini mempertimbangkan seluruh faktor namun pada akhirnya mengabaikan faktor-faktor yang tidak signifikan terhadap jumlah premi.

Interpretasi :

Premi dasar dimulai dari \$2,150.03, lalu akan berubah tergantung hal-hal seperti:

- Status menikah : premi naik \$85.98.
- Punya asuransi sebelumnya : semakin tinggi penyesuaian, premi naik sekitar \$0.9997 per unit.
- Penyesuaian karena riwayat klaim : menambah premi sekitar \$0.9998 per unit.
- Jika hanya punya polis tanggungan (*liability-only*) →: premi lebih murah \$199.86.
- Total diskon yang diterima : setiap tambahan diskon 1 unit, premi turun hampir \$1
- Skor kredit pelanggan (*credit-based adjustment*) : semakin tinggi, premi naik sekitar \$0.9998 per unit.
- Tinggal di wilayah urban : sedikit menaikkan premi sekitar \$0.0077 (sangat kecil).
- Wilayah tempat tinggal (*regional adjustment*) : menambah premi sekitar \$0.9997 per unit perubahan wilayah.

### 3. Rekomendasi Untuk Tim Finance

- Optimalkan Program Diskon: Diskon seperti **Safe Driver**, **Bundling**, atau **Total Discounts** memang menurunkan premi dan mungkin membuat lebih banyak pelanggan tertarik untuk mengikuti asuransi, tapi perlu dihitung kembali dampak totalnya terhadap pendapatan premi keseluruhan.

- Segmentasi Pelanggan Lebih Baik: Pelanggan dengan polis **Liability Only** atau yang tinggal di daerah tertentu cenderung membayar lebih rendah. Ini dapat menjadi dasar untuk up-sell produk tambahan untuk pelanggan-pelanggan yang masuk ke dalam kategori tersebut.
- Gunakan Penyesuaian Premium secara Dinamis: Variabel seperti **Claims\_Adjustment** dan **Premium\_Adjustment\_Credit** menunjukkan bahwa premi dapat disesuaikan secara proporsional sesuai risiko nyata pelanggan.

## Soal 5 - Rangkuman, Prediksi Hasil Pemodelan, dan Business Insight

Pada soal nomor 2 dan 3 telah dibuat tiga model klasifikasi yaitu *decision tree*, *random forest*, dan *Gradient Boosting Machine (GBM)*, dari ketiga model tersebut telah diperoleh bahwa model terbaiknya adalah *Gradient Boosting Machine (GBM)*. Berdasarkan model klasifikasi terbaik tersebut, nasabah yang berpotensi memiliki premi tinggi paling dipengaruhi oleh jenis polis, status pernikahan, wilayah tempat tinggal, dan skor kredit. Nasabah dengan jenis polis “*Full Coverage*” memiliki peluang yang besar untuk masuk dalam kelompok nasabah dengan premi tinggi dibandingkan jenis polis lainnya. Selain itu, mayoritas nasabah yang memiliki premi tinggi berasal dari daerah “*Urban*”, artinya daerah ini memiliki potensi yang besar dalam pasar asuransi. Skor kredit seorang nasabah juga berpengaruh pada premi nasabah, di mana semakin rendah skor kredit seorang nasabah maka kemungkinan premi nasabah tersebut akan semakin besar. Di sisi lain, meskipun untuk nasabah dengan status pernikahan “*Married*” mendapatkan diskon premi, namun data menunjukkan bahwa mayoritas nasabah dengan premi tinggi justru berasal dari kelompok nasabah ini, yang berarti diskon tersebut belum cukup menurunkan total premi nasabah tersebut menjadi rendah.

Dari informasi ini, strategi seleksi pelanggan dapat difokuskan pada promosi produk “*Full Coverage*” untuk individu yang tinggal di daerah *urban* dan memiliki skor kredit yang menengah ke bawah. Selain itu, perusahaan juga dapat tetap menjual produk pada nasabah yang memiliki status “*Married*”, karena kelompok ini tetap memberikan kontribusi pada pendapatan dari premi. Profit perusahaan dari pendapatan premi tentunya akan meningkat jika perusahaan dapat mengidentifikasi dan menargetkan kelompok nasabah yang cenderung menghasilkan premi tinggi. Di sisi lain, model ini juga dapat membantu perusahaan untuk menghindari kesalahan dalam penetapan harga, sehingga potensi kerugian atau peluang keuntungan yang terlewat dapat dikurangi.

Selanjutnya akan diambil 15 observasi baru dari data test secara acak. Di mana kategori **Premium Class**nya diambil dari hasil prediksi model pohon terbaik yang telah dipilih.

```
set.seed(123)
# Ambil indeks untuk Premium_Class == "High"
high_index <- which(test_data$Premium_Class == "High")
new_high <- sample(high_index, 10)

# Ambil indeks untuk Premium_Class == "Low"
low_index <- which(test_data$Premium_Class == "Low")
new_low <- sample(low_index, 5)

# Gabungkan indeks test
new_index <- c(new_high, new_low)

# Buat data baru
new_obs <- test_data[new_index, ]
new_obs[] <- lapply(new_obs, function(x) {
  if (is.character(x)) as.factor(x) else x
})
```

```

# Bangun model GBM terbaik
best_gbm_model <- gbm(Premium_Class ~ ., data = train_biner,
                      distribution = "bernoulli",
                      n.trees = 2000, interaction.depth = 1, n.minobsinnode=30,
                      shrinkage = 0.1, verbose = FALSE)

# Mengambil Output dari prediksi model GBM terbaik
gbm_prob <- predict(best_gbm_model, newdata = new_obs)

```

## Using 2000 trees...

```

gbm_pred <- ifelse(gbm_prob > 0.5, "High", "Low")
new_obs$Premium_Class <- factor(gbm_pred, levels = c("High", "Low"))

```

Setelah itu, akan dibangun model regresi logistik untuk memprediksi *premium class* dari 15 observasi baru yang telah dibentuk.

```

#Modeling Logistic Regression untuk melakukan prediksi
model_logit <- glm(Premium_Class ~ ., data = train_data, family = binomial(link = "logit"))

```

```

## Warning: glm.fit: algorithm did not converge

## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred

summary(model_logit)

```

```

##
## Call:
## glm(formula = Premium_Class ~ ., family = binomial(link = "logit"),
##      data = train_data)
##
## Coefficients: (5 not defined because of singularities)
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -1.938e+02 8.507e+04 -0.002  0.998
## Age                  -4.395e-03 1.991e+02  0.000  1.000
## Is_Senior            -1.746e-02 7.649e+03  0.000  1.000
## Marital_StatusMarried -8.864e+01 1.143e+04 -0.008  0.994
## Marital_StatusSingle   -7.668e-01 7.582e+03  0.000  1.000
## Marital_StatusWidowed -4.856e-01 9.111e+03  0.000  1.000
## Married_Premium_Discount NA          NA        NA        NA
## Prior_Insurance>5 years 8.796e+01 1.047e+04  0.008  0.993
## Prior_Insurance1-5 years 4.404e+01 7.677e+03  0.006  0.995
## Prior_Insurance_Premium_Adjustment NA          NA        NA        NA
## Claims_Frequency      -8.522e-01 1.517e+04  0.000  1.000
## Claims_SeverityLow     -4.825e-01 9.027e+03  0.000  1.000
## Claims_SeverityMedium -4.034e-01 1.014e+04  0.000  1.000
## Claims_Adjustment      -8.626e-01 2.643e+02 -0.003  0.997
## Policy_TypeLiability-Only 1.742e+02 1.625e+04  0.011  0.991
## Policy_Adjustment       NA          NA        NA        NA
## Safe_Driver_Discount    4.388e+01 6.286e+03  0.007  0.994
## Multi_Policy_Discount   4.350e+01 5.583e+03  0.008  0.994

```

```

## Bundling_Discount           4.428e+01 7.973e+03 0.006 0.996
## Total_Discounts            NA          NA          NA          NA
## Source_of_LeadOnline        1.101e-02 4.623e+03 0.000 1.000
## Source_of_LeadReferral      9.229e-02 8.159e+03 0.000 1.000
## Time_Since_First_Contact   1.279e-03 2.614e+02 0.000 1.000
## Conversion_Status          -3.845e+00 5.934e+04 0.000 1.000
## Website_Visits             5.529e-03 8.646e+02 0.000 1.000
## Inquiries                   9.685e-03 1.471e+03 0.000 1.000
## Quotes_Requested            1.154e-01 2.517e+03 0.000 1.000
## Time_to_Conversion          -4.703e-02 6.524e+02 0.000 1.000
## Credit_Score                 5.640e-03 6.906e+01 0.000 1.000
## Premium_Adjustment_Credit   -8.740e-01 1.072e+02 -0.008 0.993
## RegionSuburban              -4.389e+01 6.541e+03 -0.007 0.995
## RegionUrban                  -8.793e+01 9.328e+03 -0.009 0.992
## Premium_Adjustment_Region    NA          NA          NA          NA
##
## (Dispersion parameter for binomial family taken to be 1)
##
## Null deviance: 2.2181e+03 on 1599 degrees of freedom
## Residual deviance: 1.9886e-07 on 1572 degrees of freedom
## AIC: 56
##
## Number of Fisher Scoring iterations: 25

```

```

# Prediksi Premium Class dengan Logistic Regression
slr.prob <- predict(model_logit, new_obs, type="response") #Low
slr.pred <- ifelse(slr.prob > 0.5, "Low", "High")
actual <- factor(new_obs$Premium_Class, levels = c("High", "Low"))
predicted <- factor(slr.pred, levels = c("High", "Low"))

confusionMatrix(predicted, actual)

```

```

## Confusion Matrix and Statistics
##
##               Reference
## Prediction High Low
##       High     10    0
##       Low      0    5
##
##               Accuracy : 1
##                     95% CI : (0.782, 1)
## No Information Rate : 0.6667
## P-Value [Acc > NIR] : 0.002284
##
##               Kappa : 1
##
## McNemar's Test P-Value : NA
##
##               Sensitivity : 1.0000
##               Specificity : 1.0000
## Pos Pred Value : 1.0000
## Neg Pred Value : 1.0000
## Prevalence : 0.6667
## Detection Rate : 0.6667

```

```

##      Detection Prevalence : 0.6667
##      Balanced Accuracy : 1.0000
##
##      'Positive' Class : High
##

```

Setelah diperoleh hasil prediksinya, akan digunakan formula dari divisi *finance* untuk menghitung nilai keuntungan dari hasil prediksi observasi baru tersebut.

```

# Fungsi hitung keuntungan per observasi
profit_score <- mapply(function(act, pred) {
  if (act == "High" && pred == "High") {
    return(50)  # Benar prediksi High
  } else if (act == "High" && pred == "Low") {
    return(-15) # Salah prediksi High jadi Low
  } else if (act == "Low" && pred == "Low") {
    return(-5)  # Benar prediksi Low
  } else if (act == "Low" && pred == "High") {
    return(-25) # Salah prediksi Low jadi High
  }
}, act = actual, pred = predicted)

# Lihat hasil
print(paste('Total nilai keuntungan=' , sum(profit_score)))

```

```

## [1] "Total nilai keuntungan= 475"

```

Perhatikan bahwa, dengan menggunakan *threshold* sebesar 50% diperoleh nilai keuntungan sebesar 475. Pada *threshold* 50%, model regresi logistik memiliki akurasi sebesar 1 sehingga 475 merupakan nilai keuntungan maksimum yang bisa diperoleh untuk 15 observasi baru yang telah dibentuk.