# Experiment 4

## Exploratory Data Analysis

Name : JVN GANESH

Roll No. : 21BDS0085

Checked if the dataset is imported or not imported

Console | Terminal | Background Jobs

R 4.4.1 · ~/

```
> data("mtcars")
> data("iris")
> data("airquality")
> print("21BDS0085 JVNGANESH")
[1] "21BDS0085 JVNGANESH"
> # Checking if the dataset is imported or not
> head(mtcars)
                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
> head(iris)
  Sepal.Length Sepal.Width Petal.Length Petal.Width Species
1          5.1         3.5          1.4         0.2  setosa
2          4.9         3.0          1.4         0.2  setosa
3          4.7         3.2          1.3         0.2  setosa
4          4.6         3.1          1.5         0.2  setosa
5          5.0         3.6          1.4         0.2  setosa
6          5.4         3.9          1.7         0.4  setosa
> head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
> 
```

Console Terminal Background Jobs

R 4.4.1 · ~/

```
6          5.4         3.9          1.7         0.4  setosa
> head(airquality)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
3    12     149 12.6   74     5   3
4    18     313 11.5   62     5   4
5    NA      NA 14.3   56     5   5
6    28      NA 14.9   66     5   6
> # Basic Descriptive Statistics on mtcars dataset
> mean(mtcars$mpg)                            # Mean of miles per gallon (mpg)
[1] 20.09062
> median(mtcars$hp)                           # Median of horsepower (hp)
[1] 123
> sd(mtcars$wt)                               # Standard deviation of weight (wt)
[1] 0.9784574
> var(mtcars$disp)                            # Variance of displacement (disp)
[1] 15360.8
> range(mtcars$qsec)                          # Range of quarter-mile time (qsec)
[1] 14.5 22.9
> min(mtcars$mpg)                             # Minimum miles per gallon
[1] 10.4
> max(mtcars$mpg)                             # Maximum miles per gallon
[1] 33.9
> quantile(mtcars$hp)                         # Quantiles of horsepower
   0%   25%   50%   75%  100% 
 52.0  96.5 123.0 180.0 335.0 
> IQR(mtcars$drat)                            # Interquartile range of rear axle ratio (drat)
[1] 0.84
> sum(mtcars$cyl)                             # Sum of cylinder counts
[1] 198
> prod(mtcars$gear)                           # Product of gear counts
[1] 7.52296e+17
> cumsum(mtcars$mpg)                          # Cumulative sum of mpg
 [1]  21.0  42.0  64.8  86.2 104.9 123.0 137.3 161.7 184.5 203.7 221.5 237.9 255.2 270.4 280.8 291.2 305.9
[18] 338.3 368.7 402.6 424.1 439.6 454.8 468.1 487.3 514.6 540.6 571.0 586.8 606.5 621.5 642.9
> cumprod(mtcars$gear)                        # Cumulative product of gear counts
 [1] 4.000000e+00 1.600000e+01 6.400000e+01 1.920000e+02 5.760000e+02 1.728000e+03 5.184000e+03
 [8] 2.073600e+04 8.294400e+04 3.317760e+05 1.327104e+06 3.981312e+06 1.194394e+07 3.583181e+07
[15] 1.074954e+08 3.224863e+08 9.674588e+08 3.869835e+09 1.547934e+10 6.191736e+10 1.857521e+11
[22] 5.572563e+11 1.671769e+12 5.015307e+12 1.504592e+13 6.018368e+13 3.009184e+14 1.504592e+15
[29] 7.522960e+15 3.761480e+16 1.880740e+17 7.522960e+17
> cummax(mtcars$hp)                           # Cumulative maximum of horsepower
 [1] 110 110 110 110 175 175 245 245 245 245 245 245 245 245 245 245 245 245 245 245 245 245 245 245 245
[26] 245 245 245 264 264 335 335
> cummin(mtcars$hp)                           # Cumulative minimum of horsepower
 [1] 110 110  93  93  93  93  93  62  62  62  62  62  62  62  62  62  62  62  52  52  52  52  52  52  52
[26]  52  52  52  52  52  52  52
> cor(mtcars$mpg, mtcars$hp)                  # Correlation between mpg and hp
[1] -0.7761684
> cov(mtcars$mpg, mtcars$wt)                  # Covariance between mpg and wt
[1] -5.116685
> 
```

Console Terminal Background Jobs

R 4.4.1 · ~/

```
> cor(mtcars$mpg, mtcars$hp)                  # Correlation between mpg and hp
[1] -0.7761684
> cov(mtcars$mpg, mtcars$wt)                  # Covariance between mpg and wt
[1] -5.116685
> # Table and summary functions on iris dataset only
> table(iris$Species)                         # Contingency table of species

    setosa versicolor  virginica 
        50         50         50 
> prop.table(table(iris$Species))             # Proportions table of species

    setosa versicolor  virginica 
 0.3333333  0.3333333  0.3333333 
> fivenum(iris$Sepal.Length)                  # Five-number summary of Sepal Length
[1] 4.3 5.1 5.8 6.4 7.9
> summary(iris$Sepal.Width)                   # Summary of Sepal Width
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max. 
  2.000   2.800   3.000   3.057   3.300   4.400 
```

Console Terminal Background Jobs

R 4.4.1 · ~/

```
  2.000   2.800   3.000   3.057   3.300   4.400 
> # Hypothesis Testing on mtcars dataset
> t.test(mtcars$mpg ~ mtcars$am)             # t-test between mpg for automatic vs manual

	Welch Two Sample t-test

data:  mtcars$mpg by mtcars$am
t = -3.7671, df = 18.332, p-value = 0.001374
alternative hypothesis: true difference in means between group 0 and group 1 is not equal to 0
95 percent confidence interval:
 -11.280194  -3.209684
sample estimates:
mean in group 0 mean in group 1 
       17.14737        24.39231 

> wilcox.test(mtcars$mpg ~ mtcars$am)        # Wilcoxon test between mpg for automatic vs manual

	Wilcoxon rank sum test with continuity correction

data:  mtcars$mpg by mtcars$am
W = 42, p-value = 0.001871
alternative hypothesis: true location shift is not equal to 0

Warning message:
In wilcox.test.default(x = DATA[[1L]], y = DATA[[2L]], ...) :
  cannot compute exact p-value with ties
> chisq.test(table(mtcars$am, mtcars$gear))  # Chi-squared test for am and gear

	Pearson's Chi-squared test

data:  table(mtcars$am, mtcars$gear)
X-squared = 20.945, df = 2, p-value = 2.831e-05

Warning message:
In chisq.test(table(mtcars$am, mtcars$gear)) :
  Chi-squared approximation may be incorrect
> fisher.test(table(mtcars$cyl, mtcars$am))  # Fisher's exact test for cyl and am

	Fisher's Exact Test for Count Data

data:  table(mtcars$cyl, mtcars$am)
p-value = 0.009105
alternative hypothesis: two.sided
```

Console Terminal Background Jobs

R 4.4.1 · ~/

```
> fisher.test(table(mtcars$cyl, mtcars$am))  # Fisher's exact test for cyl and am

	Fisher's Exact Test for Count Data

data:  table(mtcars$cyl, mtcars$am)
p-value = 0.009105
alternative hypothesis: two.sided

> shapiro.test(mtcars$mpg)                   # Shapiro-Wilk test for normality of mpg

	Shapiro-Wilk normality test

data:  mtcars$mpg
W = 0.94756, p-value = 0.1229

> ks.test(mtcars$mpg, "pnorm", mean = mean(mtcars$mpg), sd = sd(mtcars$mpg))  # Kolmogorov-Smirnov test for mp
g

	Asymptotic one-sample Kolmogorov-Smirnov test

data:  mtcars$mpg
D = 0.1263, p-value = 0.687
alternative hypothesis: two-sided

Warning message:
In ks.test.default(mtcars$mpg, "pnorm", mean = mean(mtcars$mpg),  :
  ties should not be present for the one-sample Kolmogorov-Smirnov test
> bartlett.test(mtcars$mpg ~ mtcars$gear)    # Bartlett's test for homogeneity of variance

	Bartlett test of homogeneity of variances

data:  mtcars$mpg by mtcars$gear
Bartlett's K-squared = 3.8253, df = 2, p-value = 0.1477

> fligner.test(mtcars$mpg ~ mtcars$gear)     # Fligner-Killeen test for homogeneity of variance

	Fligner-Killeen test of homogeneity of variances

data:  mtcars$mpg by mtcars$gear
Fligner-Killeen:med chi-squared = 2.2831, df = 2, p-value = 0.3193

> mcnemar.test(table(mtcars$vs, mtcars$am))  # McNemar's test for paired nominal data

	McNemar's Chi-squared test with continuity correction

data:  table(mtcars$vs, mtcars$am)
McNemar's chi-squared = 0, df = 1, p-value = 1
```

R 4.4.1 · ~/

```
        Asymptotic one-sample Kolmogorov-Smirnov test

data:  mtcars$mpg
D = 0.1263, p-value = 0.687
alternative hypothesis: two-sided

Warning message:
In ks.test.default(mtcars$mpg, "pnorm", mean = mean(mtcars$mpg),  :
  ties should not be present for the one-sample Kolmogorov-Smirnov test
> bartlett.test(mtcars$mpg ~ mtcars$gear)    # Bartlett's test for homogeneity of variance

        Bartlett test of homogeneity of variances

data:  mtcars$mpg by mtcars$gear
Bartlett's K-squared = 3.8253, df = 2, p-value = 0.1477

> fligner.test(mtcars$mpg ~ mtcars$gear)     # Fligner-Killeen test for homogeneity of variance

        Fligner-Killeen test of homogeneity of variances

data:  mtcars$mpg by mtcars$gear
Fligner-Killeen:med chi-squared = 2.2831, df = 2, p-value = 0.3193

> mcnemar.test(table(mtcars$vs, mtcars$am))  # McNemar's test for paired nominal data

        McNemar's Chi-squared test with continuity correction

data:  table(mtcars$vs, mtcars$am)
McNemar's chi-squared = 0, df = 1, p-value = 1

> kruskal.test(mpg ~ gear, data = mtcars)    # Kruskal-Wallis test for mpg across gears

        Kruskal-Wallis rank sum test

data:  mpg by gear
Kruskal-Wallis chi-squared = 14.323, df = 2, p-value = 0.0007758

> friedman.test(as.matrix(mtcars[,c("mpg", "hp", "qsec")]))  # Friedman test for repeated measures

        Friedman rank sum test

data:  as.matrix(mtcars[, c("mpg", "hp", "qsec")])
Friedman chi-squared = 48.562, df = 2, p-value = 2.85e-11

> 
```

R 4.4.1 · ~/

```
data:  as.matrix(mtcars[, c("mpg", "hp", "qsec")])
Friedman chi-squared = 48.562, df = 2, p-value = 2.85e-11

> # Regression and Model Fitting on mtcars dataset
> model <- lm(mpg ~ hp + wt, data = mtcars)
> summary(model)

Call:
lm(formula = mpg ~ hp + wt, data = mtcars)

Residuals:
   Min     1Q Median     3Q    Max 
-3.941 -1.600 -0.182  1.050  5.854 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept) 37.22727    1.59879  23.285  < 2e-16 ***
hp          -0.03177    0.00903  -3.519  0.00145 ** 
wt          -3.87783    0.63273  -6.129 1.12e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.593 on 29 degrees of freedom
Multiple R-squared:  0.8268,	Adjusted R-squared:  0.8148 
F-statistic: 69.21 on 2 and 29 DF,  p-value: 9.109e-12

> aov_model <- aov(mpg ~ gear, data = mtcars)
> summary(aov_model)
            Df Sum Sq Mean Sq F value Pr(>F)   
gear         1  259.7  259.75   8.995 0.0054 **
Residuals   30  866.3   28.88                  
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
> glm_model <- glm(vs ~ mpg + hp, data = mtcars, family = binomial)
> summary(glm_model)

Call:
glm(formula = vs ~ mpg + hp, family = binomial, data = mtcars)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)  
(Intercept)  9.53119    7.03368   1.355   0.1754  
mpg         -0.03385    0.18097  -0.187   0.8516  
hp          -0.07234    0.03461  -2.090   0.0366 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.860  on 31  degrees of freedom
Residual deviance: 16.803  on 29  degrees of freedom
AIC: 22.803

Number of Fisher Scoring iterations: 7
```

R 4.4.1 · ~/

```
mpg          -0.03385    0.18097  -0.187   0.8516  
hp           -0.07234    0.03461  -2.090   0.0366 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 43.860  on 31  degrees of freedom
Residual deviance: 16.803  on 29  degrees of freedom
AIC: 22.803

Number of Fisher Scoring iterations: 7

> predict(model, newdata = mtcars[1:5, ])
        Mazda RX4     Mazda RX4 Wag        Datsun 710    Hornet 4 Drive Hornet Sportabout 
         23.57233          22.58348          25.27582          21.26502          18.32727 
> residuals(model)
          Mazda RX4       Mazda RX4 Wag          Datsun 710      Hornet 4 Drive   Hornet Sportabout 
        -2.57232940         -1.58348256         -2.47581872          0.13497989          0.37273336 
            Valiant          Duster 360           Merc 240D            Merc 230            Merc 280 
        -2.37381631         -1.29904236          1.51293266          0.80632669         -0.77945988 
          Merc 280C          Merc 450SE          Merc 450SL         Merc 450SLC  Cadillac Fleetwood 
        -2.17945988          0.67463146          0.25616901         -1.64993945          0.04479541 
Lincoln Continental   Chrysler Imperial            Fiat 128         Honda Civic      Toyota Corolla 
         1.03726743          5.50751301          5.80097202          1.08761978          5.85379085 
      Toyota Corona    Dodge Challenger         AMC Javelin          Camaro Z28    Pontiac Firebird 
        -3.08644148         -3.31136386         -3.94097947         -1.25202805          2.44325481 
          Fiat X1-9       Porsche 914-2        Lotus Europa      Ford Pantera L        Ferrari Dino 
        -0.32665313         -0.03737415          2.63023081         -0.74648866         -1.22541324 
      Maserati Bora          Volvo 142E 
         2.26052287         -1.58364943 
> confint(model)
                  2.5 %      97.5 %
(Intercept) 33.95738245 40.49715778
hp          -0.05024078 -0.01330512
wt          -5.17191604 -2.58374544
> step(model)
Start:  AIC=63.84
mpg ~ hp + wt

       Df Sum of Sq    RSS    AIC
<none>              195.05 63.840
- hp    1    83.274 278.32 73.217
- wt    1   252.627 447.67 88.427

Call:
lm(formula = mpg ~ hp + wt, data = mtcars)

Coefficients:
(Intercept)           hp           wt  
   37.22727     -0.03177     -3.87783  

> 
```

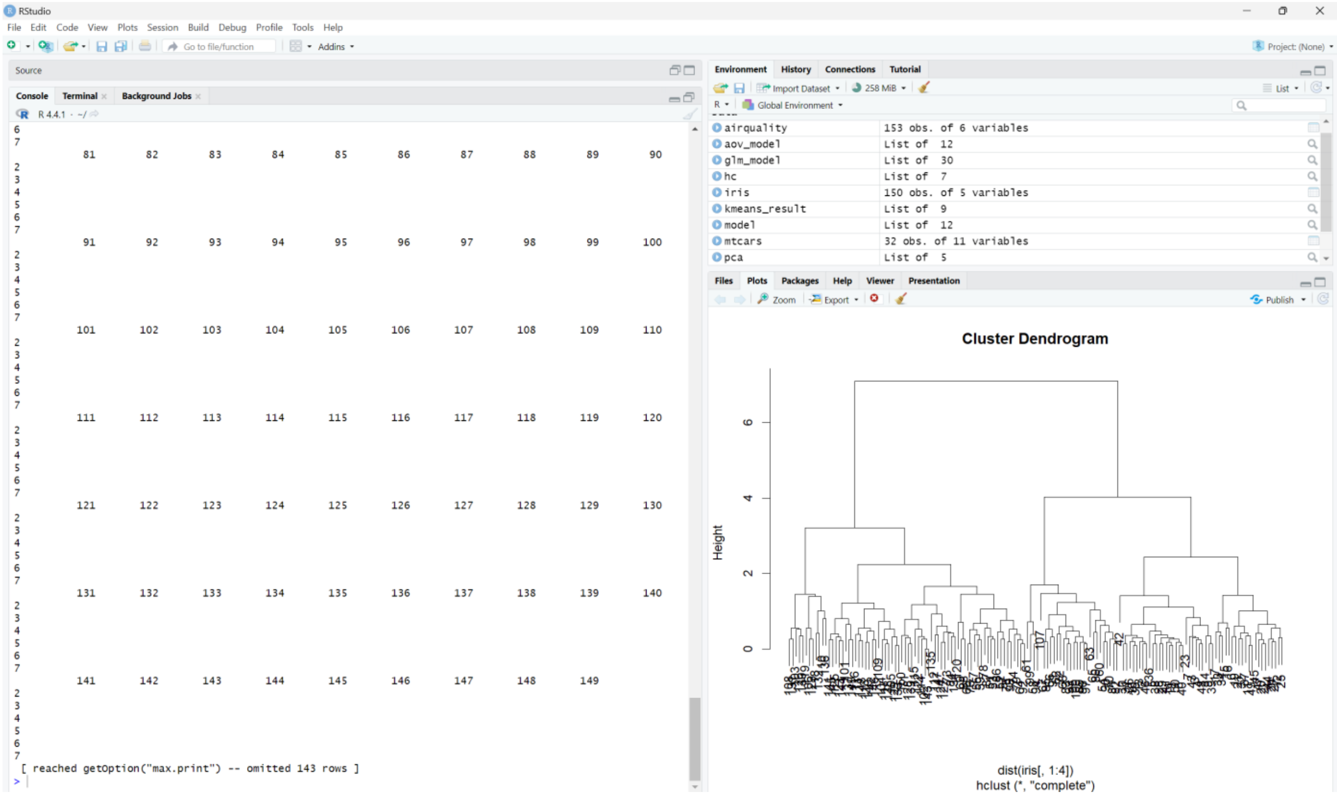
R 4.4.1 · ~/

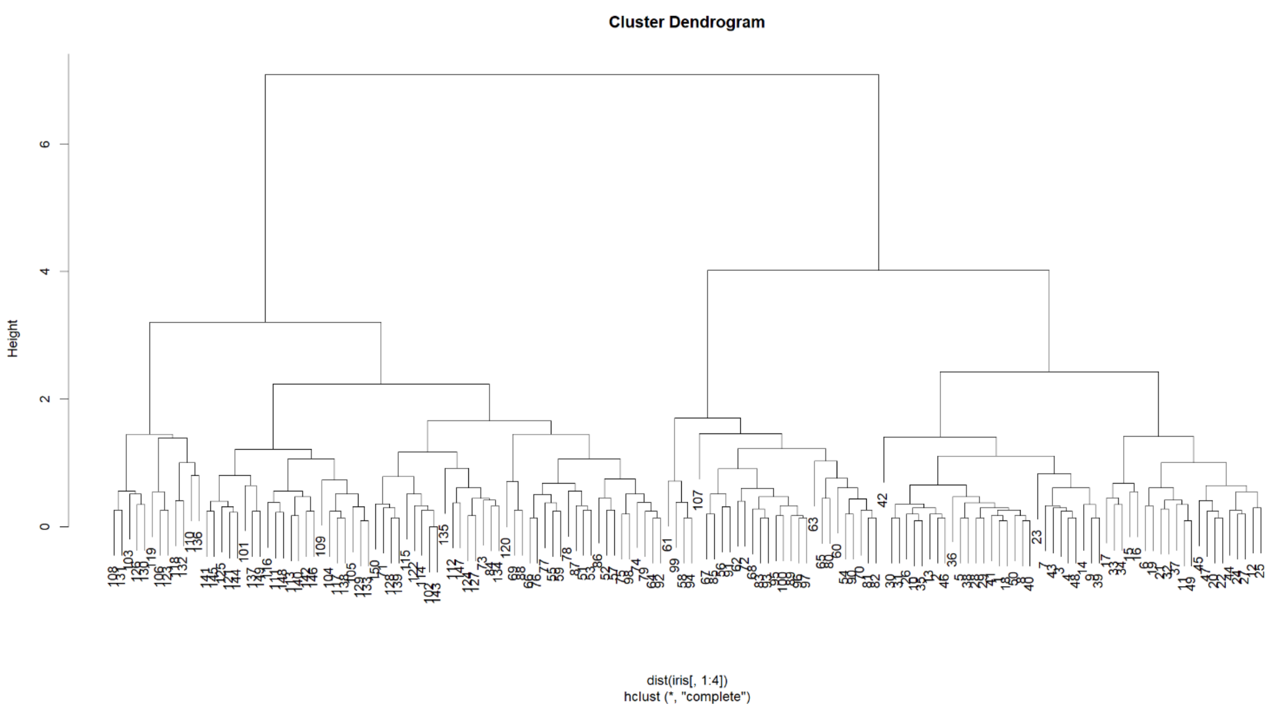
```
> # Multivariate Statistics on iris dataset
> pca <- prcomp(iris[,1:4], scale. = TRUE)
> summary(pca)
Importance of components:
                          PC1    PC2     PC3     PC4
Standard deviation     1.7084 0.9560 0.38309 0.14393
Proportion of Variance 0.7296 0.2285 0.03669 0.00518
Cumulative Proportion  0.7296 0.9581 0.99482 1.00000
> kmeans_result <- kmeans(iris[,1:4], centers = 3)
> kmeans_result
K-means clustering with 3 clusters of sizes 96, 33, 21

Cluster means:
  Sepal.Length Sepal.Width Petal.Length Petal.Width
1     6.314583    2.895833     4.973958   1.7031250
2     5.175758    3.624242     1.472727   0.2727273
3     4.738095    2.904762     1.790476   0.3523810

Clustering vector:
  [1] 2 3 3 3 2 2 2 2 3 3 2 2 3 3 2 2 2 2 2 2 2 2 2 2 3 3 2 2 2 3 3 2 2 2 3 2 2 2 3 2 2 3 3 2 2 3 2 3 2 2 1
 [52] 1 1 1 1 1 1 3 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 3 1 1 1
[103] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1

Within cluster sum of squares by cluster:
[1] 118.651875   6.432121  17.669524
 (between_SS / total_SS =  79.0 %)

Available components:

[1] "cluster"      "centers"      "totss"        "withinss"     "tot.withinss" "betweenss"
[7] "size"         "iter"         "ifault"
> hc <- hclust(dist(iris[,1:4]))
> plot(hc)
> distance_matrix <- dist(iris[,1:4])
> distance_matrix
            1         2         3         4         5         6         7         8         9        10
2   0.5385165
3   0.5099020 0.3000000
4   0.6480741 0.3316625 0.2449490
5   0.1414214 0.6082763 0.5099020 0.6480741
6   0.6164414 1.0908712 1.0862780 1.1661904 0.6164414
7   0.5196152 0.5099020 0.2645751 0.3316625 0.4582576 0.9949874
           11        12        13        14        15        16        17        18        19        20
2
3
4
5
6
7
           21        22        23        24        25        26        27        28        29        30
2
3
4
```

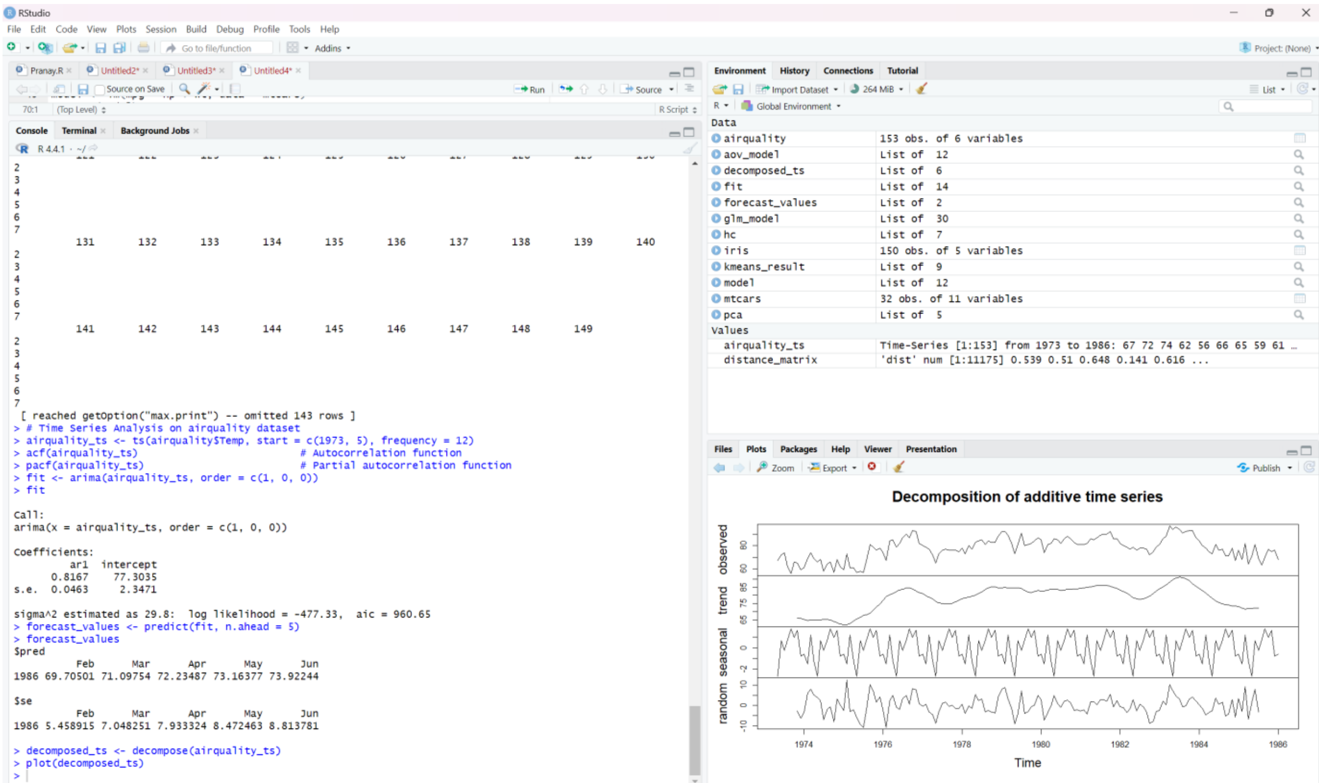
```
R 4.4.1 · ~/
> hc <- hclust(dist(iris[,1:4]))
> plot(hc)
> distance_matrix <- dist(iris[,1:4])
> distance_matrix
          1         2         3         4         5         6         7         8         9        10
2 0.5385165
3 0.5099020 0.3000000
4 0.6480741 0.3316625 0.2449490
5 0.1414214 0.6082763 0.5099020 0.6480741
6 0.6164414 1.0908712 1.0862780 1.1661904 0.6164414
7 0.5196152 0.5099020 0.2645751 0.3316625 0.4582576 0.9949874
         11        12        13        14        15        16        17        18        19        20
2
3
4
5
6
7
         21        22        23        24        25        26        27        28        29        30
2
3
4
5
6
7
         31        32        33        34        35        36        37        38        39        40
2
3
4
5
6
7
         41        42        43        44        45        46        47        48        49        50
2
3
4
5
6
7
         51        52        53        54        55        56        57        58        59        60
2
3
4
5
6
7
         61        62        63        64        65        66        67        68        69        70
2
3
4
5
6
7
```

R 4.4.1 · ~/

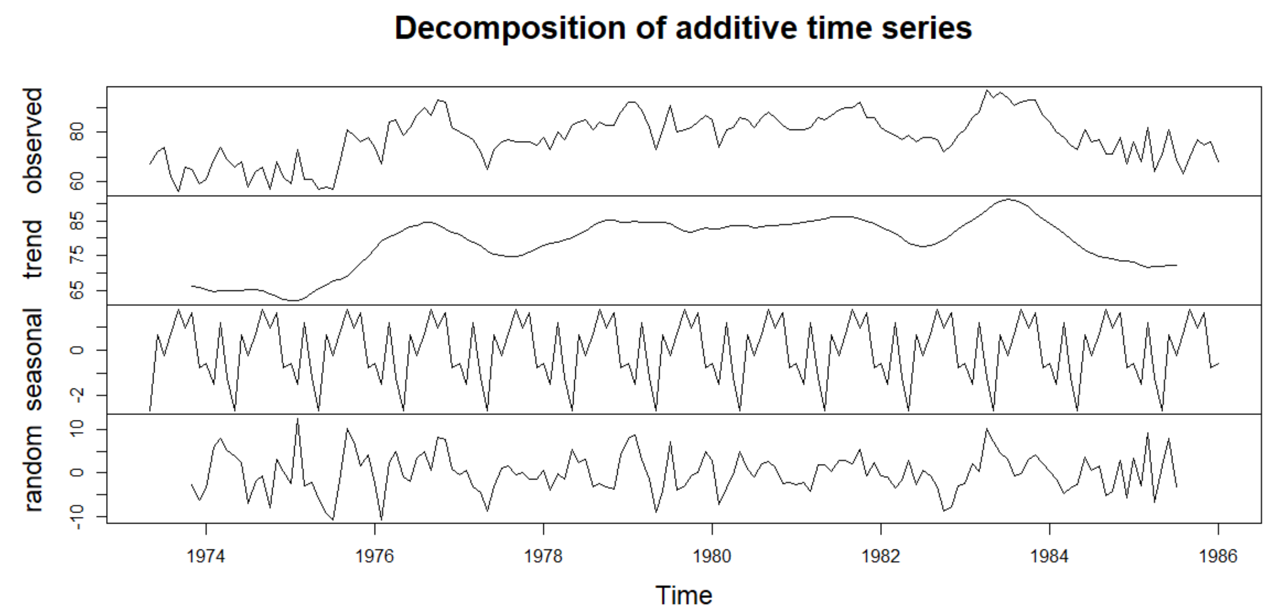
```
     61    62    63    64    65    66    67    68    69    70
2
3
4
5
6
7
     71    72    73    74    75    76    77    78    79    80
2
3
4
5
6
7
     81    82    83    84    85    86    87    88    89    90
2
3
4
5
6
7
     91    92    93    94    95    96    97    98    99   100
2
3
4
5
6
7
    101   102   103   104   105   106   107   108   109   110
2
3
4
5
6
7
    111   112   113   114   115   116   117   118   119   120
2
3
4
5
6
7
    121   122   123   124   125   126   127   128   129   130
2
3
4
5
6
7
    131   132   133   134   135   136   137   138   139   140
2
3
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Source
Console Terminal Background Jobs
R 4.4.1 · ~/
[ reached getOption("max.print") -- omitted 143 rows ]
Environment History Connections Tutorial
Import Dataset
258 MiB
List
Global Environment
airquality 153 obs. of 6 variables
aov_model List of 12
glm_model List of 30
hc List of 7
iris 150 obs. of 5 variables
kmeans_result List of 9
model List of 12
mtcars 32 obs. of 11 variables
pca List of 5
Files Plots Packages Help Viewer Presentation
Zoom
Export
Publish
Cluster Dendrogram
Height
dist(iris[, 1:4])
hclust (*, "complete")


Cluster Dendrogram
Height
dist(iris[, 1:4])
hclust (*, "complete")



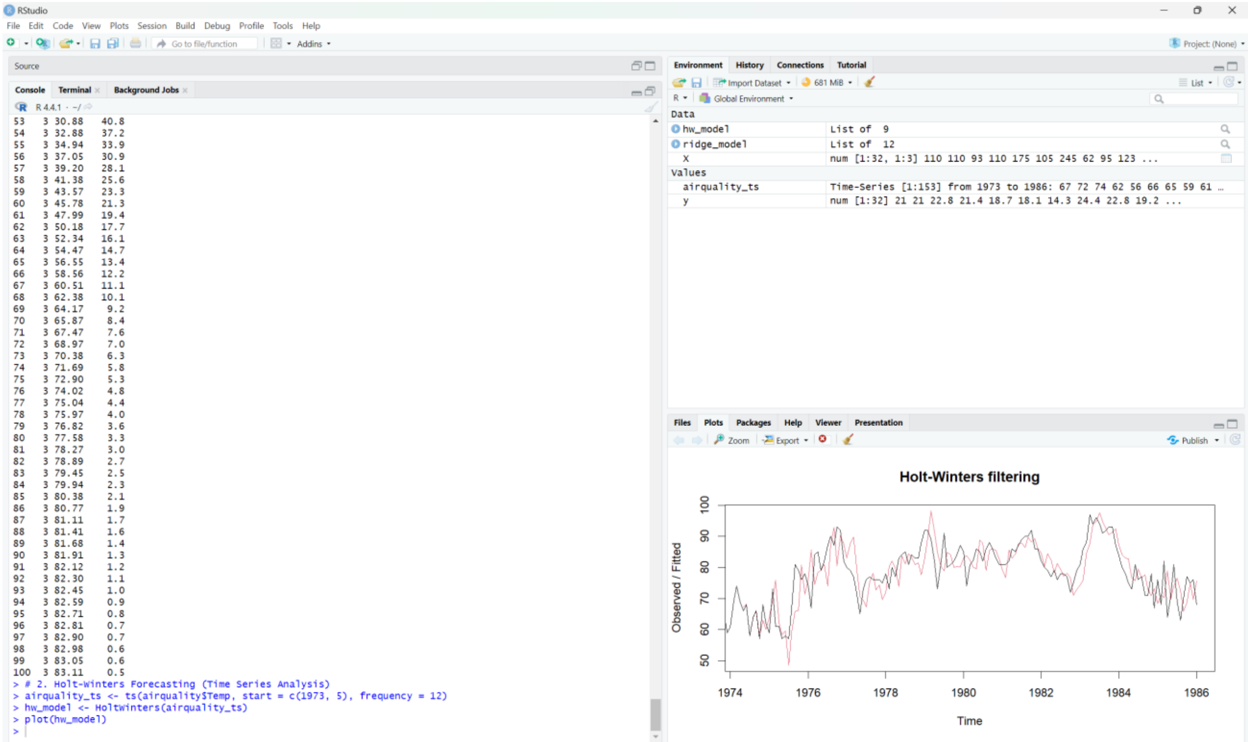

```
7
 [ reached getOption("max.print") -- omitted 143 rows ]
> # Time Series Analysis on airquality dataset
> airquality_ts <- ts(airquality$Temp, start = c(1973, 5), frequency = 12)
> acf(airquality_ts)                        # Autocorrelation function
> pacf(airquality_ts)                       # Partial autocorrelation function
> fit <- arima(airquality_ts, order = c(1, 0, 0))
> fit

Call:
arima(x = airquality_ts, order = c(1, 0, 0))

Coefficients:
         ar1  intercept
      0.8167    77.3035
s.e.  0.0463     2.3471

sigma^2 estimated as 29.8:  log likelihood = -477.33,  aic = 960.65
> forecast_values <- predict(fit, n.ahead = 5)
> forecast_values
$pred
          Feb      Mar      Apr      May      Jun
1986 69.70501 71.09754 72.23487 73.16377 73.92244

$se
          Feb      Mar      Apr      May      Jun
1986 5.458915 7.048251 7.933324 8.472463 8.813781

> decomposed_ts <- decompose(airquality_ts)
> plot(decomposed_ts)
> 
```

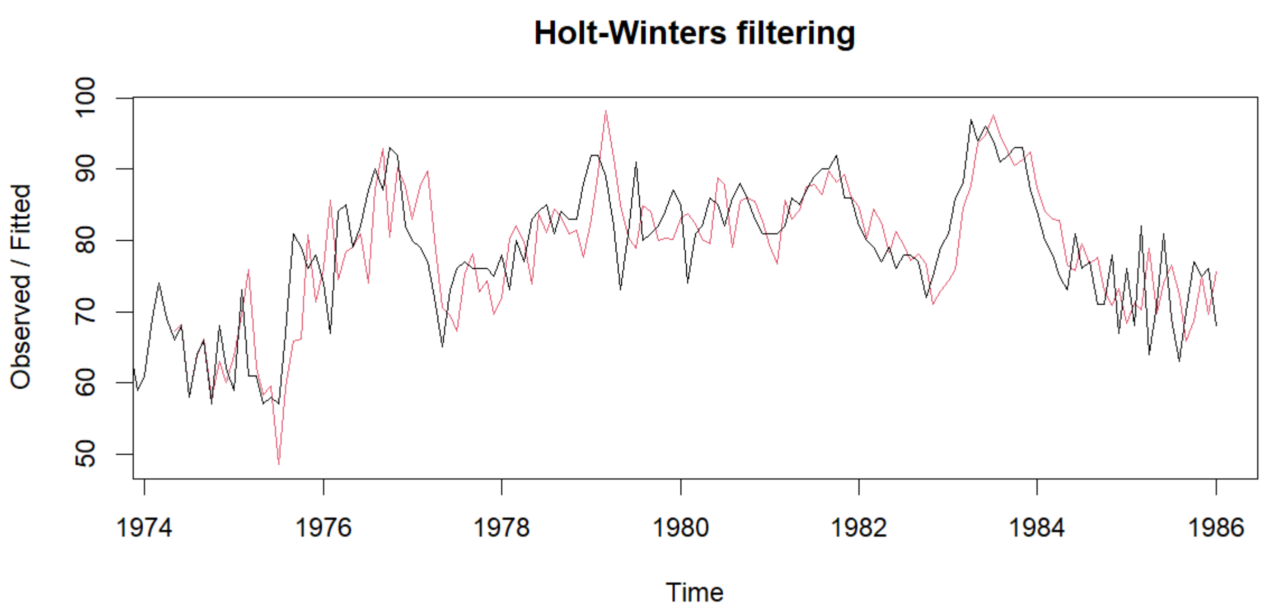



---

2nd part

```
R 4.4.1 · ~/
> print("21BDS0085 JVNGANESH")
[1] "21BDS0085 JVNGANESH"
> # 1. Ridge Regression (Advanced Regression)
> X <- as.matrix(mtcars[, c("hp", "wt", "qsec")])
> y <- mtcars$mpg
> ridge_model <- glmnet(X, y, alpha = 0)
> print(ridge_model)

Call:  glmnet(x = X, y = y, alpha = 0)

    Df  %Dev Lambda
1    3  0.00 5147.0
2    3  0.39 4690.0
3    3  0.42 4273.0
4    3  0.46 3894.0
5    3  0.51 3548.0
6    3  0.56 3232.0
7    3  0.61 2945.0
8    3  0.67 2684.0
9    3  0.74 2445.0
10   3  0.81 2228.0
11   3  0.89 2030.0
12   3  0.97 1850.0
13   3  1.07 1685.0
14   3  1.17 1536.0
15   3  1.28 1399.0
16   3  1.40 1275.0
17   3  1.54 1162.0
18   3  1.69 1058.0
19   3  1.85  964.5
20   3  2.03  878.8
21   3  2.22  800.7
22   3  2.43  729.6
23   3  2.66  664.8
24   3  2.91  605.7
25   3  3.19  551.9
26   3  3.49  502.9
27   3  3.82  458.2
28   3  4.17  417.5
29   3  4.56  380.4
30   3  4.99  346.6
31   3  5.45  315.8
32   3  5.95  287.8
33   3  6.49  262.2
34   3  7.08  238.9
35   3  7.72  217.7
36   3  8.41  198.3
37   3  9.16  180.7
38   3  9.96  164.7
39   3 10.83  150.0
40   3 11.76  136.7
41   3 12.77  124.6
42   3 13.84  113.5
43   3 14.99  103.4
```


RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Source
Console Terminal Background Jobs
R 4.4.1 · ~/
53 3 30.88 40.8
54 3 32.88 37.2
55 3 34.94 33.9
56 3 37.05 30.9
57 3 39.20 28.1
58 3 41.38 25.6
59 3 43.57 23.3
60 3 45.78 21.3
61 3 47.99 19.4
62 3 50.18 17.7
63 3 52.34 16.1
64 3 54.47 14.7
65 3 56.55 13.4
66 3 58.56 12.2
67 3 60.51 11.1
68 3 62.38 10.1
69 3 64.17 9.2
70 3 65.87 8.4
71 3 67.47 7.6
72 3 68.97 7.0
73 3 70.38 6.3
74 3 71.69 5.8
75 3 72.90 5.3
76 3 74.02 4.8
77 3 75.04 4.4
78 3 75.97 4.0
79 3 76.82 3.6
80 3 77.58 3.3
81 3 78.27 3.0
82 3 78.89 2.7
83 3 79.45 2.5
84 3 79.94 2.3
85 3 80.38 2.1
86 3 80.77 1.9
87 3 81.11 1.7
88 3 81.41 1.6
89 3 81.68 1.4
90 3 81.91 1.3
91 3 82.12 1.2
92 3 82.30 1.1
93 3 82.45 1.0
94 3 82.59 0.9
95 3 82.71 0.8
96 3 82.81 0.7
97 3 82.90 0.7
98 3 82.98 0.6
99 3 83.05 0.6
100 3 83.11 0.5
> # 2. Holt-Winters Forecasting (Time Series Analysis)
> airquality_ts <- ts(airquality$Temp, start = c(1973, 5), frequency = 12)
> hw_model <- HoltWinters(airquality_ts)
> plot(hw_model)
>
Environment History Connections Tutorial
Import Dataset
681 MiB
List
R
Global Environment
Data
hw_model List of 9
ridge_model List of 12
X num [1:32, 1:3] 110 110 93 110 175 105 245 62 95 123 ...
Values
airquality_ts Time-Series [1:153] from 1973 to 1986: 67 72 74 62 56 66 65 59 61 …
y num [1:32] 21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
Files Plots Packages Help Viewer Presentation
Zoom
Export
Publish
Holt-Winters filtering
Observed / Fitted
Time
50
60
70
80
90
100
1974
1976
1978
1980
1982
1984
1986


Holt-Winters filtering
Observed / Fitted
50
60
70
80
90
100
1974
1976
1978
1980
1982
1984
1986
Time

Console  Terminal  Background Jobs

R 4.4.1 · ~/

```
> # 2. Holt-Winters Forecasting (Time Series Analysis)
> airquality_ts <- ts(airquality$Temp, start = c(1973, 5), frequency = 12)
> hw_model <- HoltWinters(airquality_ts)
> plot(hw_model)
> # 3. Canonical Correlation Analysis (Multivariate Analysis)
> X <- mtcars[, c("mpg", "hp", "wt")]
> Y <- mtcars[, c("qsec", "drat", "gear")]
> cca_result <- cancor(X, Y)
> print(cca_result)
$cor
[1] 0.83245263 0.74932304 0.08078841

$xcoef
            [,1]         [,2]        [,3]
mpg  0.014272510 -0.013704364 0.068814500
hp  -0.002561111 -0.002154455 0.002469339
wt   0.110256608  0.181920410 0.302275068

$ycoef
            [,1]        [,2]        [,3]
qsec  0.09048096  0.02322222  0.05720641
drat  0.12282315 -0.18015783 -0.45065083
gear -0.01491230 -0.12266513  0.34848786

$xcenter
      mpg        hp        wt
 20.09062 146.68750   3.21725

$ycenter
     qsec      drat      gear
17.848750  3.596563  3.687500

> 
```

R 4.4.1 · ~/

```
> # 4. Bayesian Linear Regression (Bayesian Statistics)
> bayesian_model <- stan_glm(mpg ~ hp + wt, data = mtcars)

SAMPLING FOR MODEL 'continuous' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 0.000131 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 1.31 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 1: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 1: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 1:
Chain 1:  Elapsed Time: 0.061 seconds (Warm-up)
Chain 1:                0.058 seconds (Sampling)
Chain 1:                0.119 seconds (Total)
Chain 1:

SAMPLING FOR MODEL 'continuous' NOW (CHAIN 2).
Chain 2:
Chain 2: Gradient evaluation took 2.3e-05 seconds
Chain 2: 1000 transitions using 10 leapfrog steps per transition would take 0.23 seconds.
Chain 2: Adjust your expectations accordingly!
Chain 2:
Chain 2:
Chain 2: Iteration:    1 / 2000 [  0%]  (Warmup)
Chain 2: Iteration:  200 / 2000 [ 10%]  (Warmup)
Chain 2: Iteration:  400 / 2000 [ 20%]  (Warmup)
Chain 2: Iteration:  600 / 2000 [ 30%]  (Warmup)
Chain 2: Iteration:  800 / 2000 [ 40%]  (Warmup)
Chain 2: Iteration: 1000 / 2000 [ 50%]  (Warmup)
Chain 2: Iteration: 1001 / 2000 [ 50%]  (Sampling)
Chain 2: Iteration: 1200 / 2000 [ 60%]  (Sampling)
Chain 2: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 2: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 2: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 2: Iteration: 2000 / 2000 [100%]  (Sampling)
```

Console Terminal Background Jobs

R 4.4.1 · ~/

```
Chain 4: Iteration: 1400 / 2000 [ 70%]  (Sampling)
Chain 4: Iteration: 1600 / 2000 [ 80%]  (Sampling)
Chain 4: Iteration: 1800 / 2000 [ 90%]  (Sampling)
Chain 4: Iteration: 2000 / 2000 [100%]  (Sampling)
Chain 4: 
Chain 4:  Elapsed Time: 0.056 seconds (Warm-up)
Chain 4:                0.062 seconds (Sampling)
Chain 4:                0.118 seconds (Total)
Chain 4: 
> print(summary(bayesian_model))

Model Info:
 function:     stan_glm
 family:       gaussian [identity]
 formula:      mpg ~ hp + wt
 algorithm:    sampling
 sample:       4000 (posterior sample size)
 priors:       see help('prior_summary')
 observations: 32
 predictors:   3

Estimates:
              mean   sd   10%   50%   90%
(Intercept) 37.2    1.7 35.0  37.3  39.4 
hp           0.0    0.0  0.0   0.0   0.0 
wt          -3.9    0.7 -4.7  -3.9  -3.0 
sigma        2.7    0.4  2.2   2.6   3.2 

Fit Diagnostics:
           mean   sd   10%   50%   90%
mean_PPD 20.1    0.7 19.3  20.1  20.9 

The mean_ppd is the sample average posterior predictive distribution of the outcome variable (for details se
e help('summary.stanreg')).

MCMC diagnostics
              mcse Rhat n_eff
(Intercept)   0.0  1.0  3410 
hp            0.0  1.0  1884 
wt            0.0  1.0  1781 
sigma         0.0  1.0  2542 
mean_PPD      0.0  1.0  3239 
log-posterior 0.0  1.0  1569 

For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample size, a
nd Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).
```

```
For each parameter, mcse is Monte Carlo standard error, n_eff is a crude measure of effective sample size,
nd Rhat is the potential scale reduction factor on split chains (at convergence Rhat=1).
> # 3. Canonical Correlation Analysis (Multivariate Analysis)
> X <- mtcars[, c("mpg", "hp", "wt")]
> Y <- mtcars[, c("qsec", "drat", "gear")]
> cca_result <- cancor(X, Y)
> print(cca_result)
$cor
[1] 0.83245263 0.74932304 0.08078841

$xcoef
            [,1]         [,2]        [,3]
mpg  0.014272510 -0.013704364 0.068814500
hp  -0.002561111 -0.002154455 0.002469339
wt   0.110256608  0.181920410 0.302275068

$ycoef
            [,1]        [,2]        [,3]
qsec  0.09048096  0.02322222  0.05720641
drat  0.12282315 -0.18015783 -0.45065083
gear -0.01491230 -0.12266513  0.34848786

$xcenter
      mpg        hp        wt
 20.09062 146.68750   3.21725

$ycenter
     qsec      drat      gear
17.848750  3.596563  3.687500

>
```

Console Terminal Background Jobs

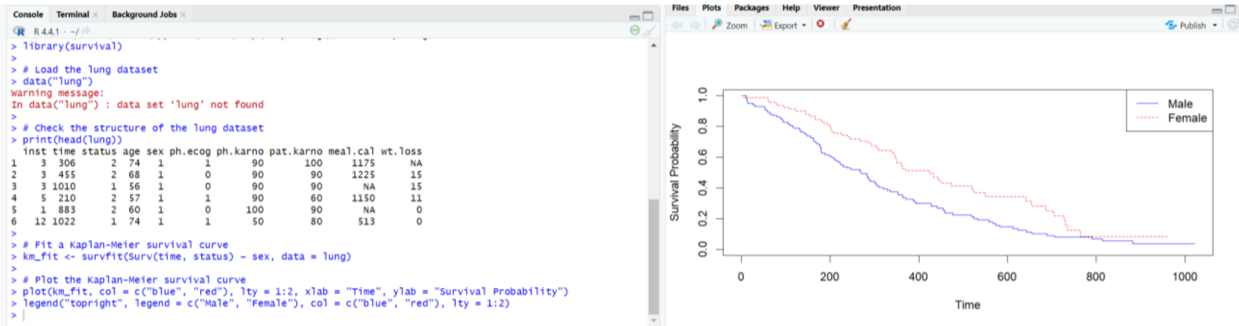
R 4.4.1 · ~/

```
gear -0.01491230 -0.12266513  0.34848786

$xcenter
      mpg        hp        wt
 20.09062 146.68750   3.21725

$ycenter
     qsec      drat      gear
17.848750  3.596563  3.687500

> # 5. DBSCAN Clustering (Clustering and Classification)
> X <- iris[, 1:4]
> dbscan_result <- dbscan(X, eps = 0.5, minPts = 5)
> print(dbscan_result)
DBSCAN clustering for 150 objects.
Parameters: eps = 0.5, minPts = 5
Using euclidean distances and borderpoints = TRUE
The clustering contains 2 cluster(s) and 17 noise points.

 0  1  2
17 49 84

Available fields: cluster, eps, minPts, metric, borderPoints
> # 6. Mann-Whitney U Test (Non-Parametric Test)
> wilcox_test <- wilcox.test(mpg ~ am, data = mtcars)
Warning message:
In wilcox.test.default(x = DATA[[1L]], y = DATA[[2L]], ...) :
  cannot compute exact p-value with ties
> print(wilcox_test)

        Wilcoxon rank sum test with continuity correction

data:  mpg by am
W = 42, p-value = 0.001871
alternative hypothesis: true location shift is not equal to 0

> # 7. Robust Regression (Robust Statistics)
> robust_model <- rlm(mpg ~ hp + wt, data = mtcars)
> summary(robust_model)

Call: rlm(formula = mpg ~ hp + wt, data = mtcars)
Residuals:
    Min      1Q  Median      3Q     Max
-3.6639 -1.3057  0.1727  1.3162  6.3392

Coefficients:
            Value   Std. Error t value
(Intercept) 36.5840  1.4380    25.4407
hp          -0.0293  0.0081    -3.6050
wt          -3.8801  0.5691    -6.8180

Residual standard error: 2.006 on 29 degrees of freedom
>
```

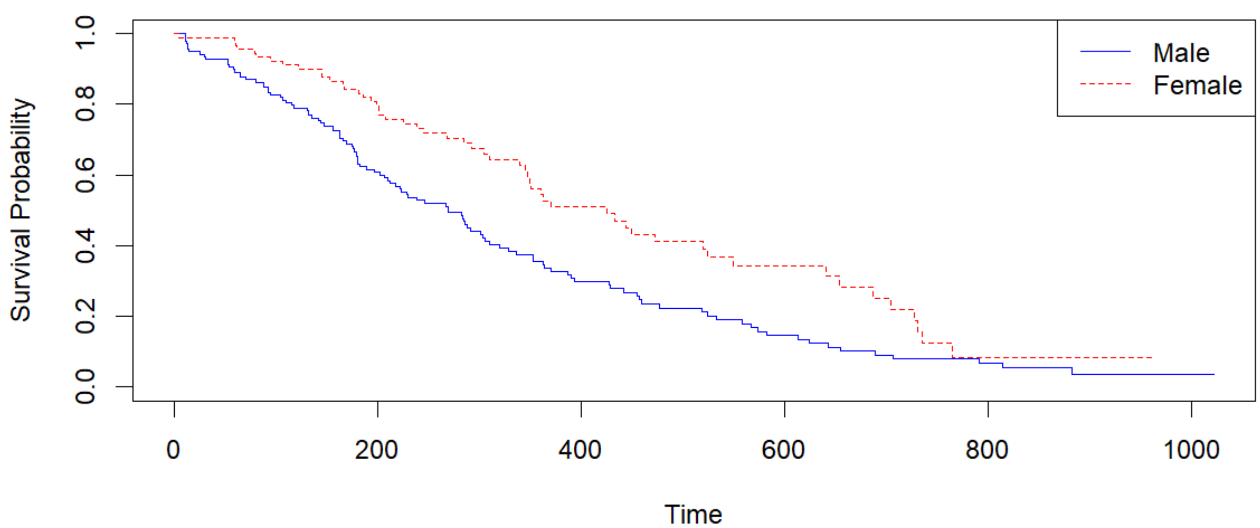
Console Terminal Background Jobs

R 4.4.1 · ~/

```
Call: rlm(formula = mpg ~ hp + wt, data = mtcars)
Residuals:
    Min      1Q  Median      3Q     Max 
-3.6639 -1.3057  0.1727  1.3162  6.3392 

Coefficients:
            Value   Std. Error t value
(Intercept) 36.5840  1.4380    25.4407
hp          -0.0293  0.0081    -3.6050
wt          -3.8801  0.5691    -6.8180

Residual standard error: 2.006 on 29 degrees of freedom
> # 8. Path Analysis (Structural Equation Modeling - SEM)
> model <- '
+   mpg ~ hp + wt
+   hp ~ wt
+ '
> sem_fit <- sem(model, data = mtcars)
> summary(sem_fit)
lavaan 0.6-18 ended normally after 1 iteration

  Estimator                                         ML
  Optimization method                           NLMINB
  Number of model parameters                         5

  Number of observations                            32

Model Test User Model:
                                                      
  Test statistic                                 0.000
  Degrees of freedom                                 0

Parameter Estimates:

  Standard errors                             Standard
  Information                                 Expected
  Information saturated (h1) model          Structured

Regressions:
                   Estimate  Std.Err  z-value  P(>|z|)
  mpg ~                                               
    hp               -0.032    0.009   -3.696    0.000
    wt               -3.878    0.602   -6.438    0.000
  hp ~                                                
    wt               46.160    9.320    4.953    0.000

Variances:
                   Estimate  Std.Err  z-value  P(>|z|)
   .mpg               6.095    1.524    4.000    0.000
   .hp             2577.777  644.444    4.000    0.000

> 
```




```
Console   Terminal   Background Jobs
R 4.4.1 · ~/
> library(survival)
>
> # Load the lung dataset
> data("lung")
Warning message:
In data("lung") : data set ‘lung’ not found
>
> # Check the structure of the lung dataset
> print(head(lung))
  inst time status age sex ph.ecog ph.karno pat.karno meal.cal wt.loss
1    3  306      2  74   1       1       90       100     1175      NA
2    3  455      2  68   1       0       90        90     1225      15
3    3 1010      1  56   1       0       90        90       NA      15
4    5  210      2  57   1       1       90        60     1150      11
5    1  883      2  60   1       0      100        90       NA       0
6   12 1022      1  74   1       1       50        80      513       0
>
> # Fit a Kaplan-Meier survival curve
> km_fit <- survfit(Surv(time, status) ~ sex, data = lung)
>
> # Plot the Kaplan-Meier survival curve
> plot(km_fit, col = c("blue", "red"), lty = 1:2, xlab = "Time", ylab = "Survival Probability")
> legend("topright", legend = c("Male", "Female"), col = c("blue", "red"), lty = 1:2)
> |
```



Code (part 1):

```
data("mtcars")

data("iris")

data("airquality")

pint("21BDS0085 JVNGANESH")

# Checking if the dataset is imported or not

head(mtcars)

head(iris)

head(airquality)


# Basic Descriptive Statistics on mtcars dataset

mean(mtcars$mpg)            # Mean of miles per gallon (mpg)

median(mtcars$hp)           # Median of horsepower (hp)

sd(mtcars$wt)               # Standard deviation of weight (wt)

var(mtcars$disp)            # Variance of displacement (disp)

range(mtcars$qsec)          # Range of quarter-mile time (qsec)

min(mtcars$mpg)             # Minimum miles per gallon

max(mtcars$mpg)             # Maximum miles per gallon

quantile(mtcars$hp)         # Quantiles of horsepower

IQR(mtcars$drat)            # Interquartile range of rear axle ratio (drat)

sum(mtcars$cyl)             # Sum of cylinder counts
```

```
prod(mtcars$gear)                    # Product of gear counts
cumsum(mtcars$mpg)                   # Cumulative sum of mpg
cumprod(mtcars$gear)                 # Cumulative product of gear counts
cummax(mtcars$hp)                    # Cumulative maximum of horsepower
cummin(mtcars$hp)                    # Cumulative minimum of horsepower
cor(mtcars$mpg, mtcars$hp)           # Correlation between mpg and hp
cov(mtcars$mpg, mtcars$wt)           # Covariance between mpg and wt

# Table and summary functions on iris dataset only
table(iris$Species)                  # Contingency table of species
prop.table(table(iris$Species))      # Proportions table of species
fivenum(iris$Sepal.Length)           # Five-number summary of Sepal Length
summary(iris$Sepal.Width)            # Summary of Sepal Width

# Hypothesis Testing on mtcars dataset
t.test(mtcars$mpg ~ mtcars$am)           # t-test between mpg for automatic vs manual
wilcox.test(mtcars$mpg ~ mtcars$am)      # Wilcoxon test between mpg for automatic vs manual
chisq.test(table(mtcars$am, mtcars$gear))  # Chi-squared test for am and gear
fisher.test(table(mtcars$cyl, mtcars$am))  # Fisher's exact test for cyl and am
shapiro.test(mtcars$mpg)             # Shapiro-Wilk test for normality of mpg
```

```
ks.test(mtcars$mpg, "pnorm", mean = mean(mtcars$mpg), sd =
sd(mtcars$mpg))  # Kolmogorov-Smirnov test for mpg

bartlett.test(mtcars$mpg ~ mtcars$gear)    # Bartlett's test for homogeneity of
variance

fligner.test(mtcars$mpg ~ mtcars$gear)     # Fligner-Killeen test for
homogeneity of variance

mcnemar.test(table(mtcars$vs, mtcars$am))  # McNemar's test for paired
nominal data

kruskal.test(mpg ~ gear, data = mtcars)    # Kruskal-Wallis test for mpg across
gears

friedman.test(as.matrix(mtcars[,c("mpg", "hp", "qsec")]))  # Friedman test for
repeated measures


# Regression and Model Fitting on mtcars dataset

model <- lm(mpg ~ hp + wt, data = mtcars)

summary(model)

aov_model <- aov(mpg ~ gear, data = mtcars)

summary(aov_model)

glm_model <- glm(vs ~ mpg + hp, data = mtcars, family = binomial)

summary(glm_model)

predict(model, newdata = mtcars[1:5, ])

residuals(model)

confint(model)

step(model)
```

```
# Multivariate Statistics on iris dataset
pca <- prcomp(iris[,1:4], scale. = TRUE)
summary(pca)
kmeans_result <- kmeans(iris[,1:4], centers = 3)
kmeans_result
hc <- hclust(dist(iris[,1:4]))
plot(hc)
distance_matrix <- dist(iris[,1:4])
distance_matrix


# Time Series Analysis on airquality dataset
airquality_ts <- ts(airquality$Temp, start = c(1973, 5), frequency = 12)
acf(airquality_ts)                    # Autocorrelation function
pacf(airquality_ts)                   # Partial autocorrelation function
fit <- arima(airquality_ts, order = c(1, 0, 0))
fit
forecast_values <- predict(fit, n.ahead = 5)
forecast_values
decomposed_ts <- decompose(airquality_ts)
plot(decomposed_ts)
```

Code (part 2):

```
# Install all required packages
install.packages("glmnet")     # For Ridge Regression
```

```r
install.packages("forecast")   # For Holt-Winters Forecasting
install.packages("CCA")        # For Canonical Correlation Analysis
install.packages("rstanarm")   # For Bayesian Linear Regression
install.packages("dbscan")     # For DBSCAN Clustering
install.packages("MASS")       # For Robust Regression and Canonical
Discriminant Analysis
install.packages("lavaan")     # For Structural Equation Modeling (SEM)
install.packages("plm")        # For Panel Data Analysis
install.packages("survival")   # For Kaplan-Meier Estimator
install.packages("lme4")       # For Linear Mixed Models (LMM)
install.packages("coda")       # For MCMC Simulation


# Load necessary libraries
library(glmnet)     # Ridge Regression
library(forecast)   # Holt-Winters Forecasting
library(CCA)        # Canonical Correlation Analysis
library(rstanarm)   # Bayesian Linear Regression
library(dbscan)     # DBSCAN Clustering
library(MASS)       # Robust Regression, Canonical Discriminant Analysis
library(lavaan)     # Structural Equation Modeling (SEM)
library(plm)        # Panel Data Analysis
library(survival)   # Kaplan-Meier Estimator
```

```r
library(lme4)      # Linear Mixed Models (LMM)
library(coda)      # MCMC

print("21BDS0085 JVNGANESH")
# 1. Ridge Regression (Advanced Regression)
X <- as.matrix(mtcars[, c("hp", "wt", "qsec")])
y <- mtcars$mpg
ridge_model <- glmnet(X, y, alpha = 0)
print(ridge_model)

# 2. Holt-Winters Forecasting (Time Series Analysis)
airquality_ts <- ts(airquality$Temp, start = c(1973, 5), frequency = 12)
hw_model <- HoltWinters(airquality_ts)
plot(hw_model)

# 3. Canonical Correlation Analysis (Multivariate Analysis)
X <- mtcars[, c("mpg", "hp", "wt")]
Y <- mtcars[, c("qsec", "drat", "gear")]
cca_result <- cancor(X, Y)
print(cca_result)

# 4. Bayesian Linear Regression (Bayesian Statistics)
bayesian_model <- stan_glm(mpg ~ hp + wt, data = mtcars)
```

```r
print(summary(bayesian_model))

# 5. DBSCAN Clustering (Clustering and Classification)
X <- iris[, 1:4]
dbscan_result <- dbscan(X, eps = 0.5, minPts = 5)
print(dbscan_result)

# 6. Mann-Whitney U Test (Non-Parametric Test)
wilcox_test <- wilcox.test(mpg ~ am, data = mtcars)
print(wilcox_test)

# 7. Robust Regression (Robust Statistics)
robust_model <- rlm(mpg ~ hp + wt, data = mtcars)
summary(robust_model)

# 8. Path Analysis (Structural Equation Modeling - SEM)
model <- '
  mpg ~ hp + wt
  hp ~ wt
'
sem_fit <- sem(model, data = mtcars)
summary(sem_fit)
```

```r
# 9. Panel Data Analysis (Econometrics)
mtcars$car <- rownames(mtcars)
pdata <- pdata.frame(mtcars, index = c("car", "gear"))
panel_model <- plm(mpg ~ hp + wt, data = pdata, model = "random")
summary(panel_model)

# Load the necessary package
library(plm)

# Ensure the mtcars dataset has identifiers
mtcars$car <- rownames(mtcars)  # Add a car identifier
mtcars$time <- seq_len(nrow(mtcars))  # Create a simple time identifier

# Convert the dataset into a panel data frame
pdata <- pdata.frame(mtcars, index = c("car", "time"))

# Check the structure of the panel data
print(head(pdata))

# Fit a panel data model (Random effects model)
panel_model <- plm(mpg ~ hp + wt, data = pdata, model = "random")

# Summarize the model
```

```
summary(panel_model)



# 10. Kaplan-Meier Estimator (Survival Analysis)
# Install and load the necessary package
install.packages("survival")
library(survival)

# Load the lung dataset
data("lung")

# Check the structure of the lung dataset
print(head(lung))

# Fit a Kaplan-Meier survival curve
km_fit <- survfit(Surv(time, status) ~ sex, data = lung)

# Plot the Kaplan-Meier survival curve
plot(km_fit, col = c("blue", "red"), lty = 1:2, xlab = "Time", ylab = "Survival Probability")
legend("topright", legend = c("Male", "Female"), col = c("blue", "red"), lty = 1:2)
```