Hands-on Exercise 4: Visual Analytics with R

Dr. Kam Tin Seong
Assoc. Professor of Information Systems

School of Computing and Information Systems,
Singapore Management University

2020-2-15 (updated: 2022-05-04)

Learning Outcome

In this hands-on exercise, you will gain hands-on experience on using:

- ggstatsplot to create visual graphics with rich statistical information,
- ggdist to visualise uncertainty on data, and
- ungeviz to build hypothetical outcome plots (HOPs).

Getting Started

In this exercise, **infer**, **ggstatsplot** and **tidyverse** will be used.

```
packages = c('ggstatsplot', 'ggside', 'knitr',
             'tidyverse', 'broom', 'ggdist',
             'ungeviz', 'gganimate')
for (p in packages){
  if(!require(p, character.only = T)){
    install.packages(p)
  }
}
```

In this exercise, the Exam.csv data will be used.

```
exam <- read_csv("data/Exam_data.csv")
```

Visual Statistical Analysis with ggstatsplot

- **ggstatsplot** is an extension of **ggplot2** package for creating graphics with details from statistical tests included in the information-rich plots themselves.
  - To provide alternative statistical inference methods by default.
  - To follow best practices for statistical reporting. For all statistical tests reported in the plots, the default template abides by the APA gold standard for statistical reporting. For example, here are results from a robust t-test:

parameter statistic significance effect size + confidence intervals number of observations

$$t(14.79) = 3.36,\ p = 0.004,\ \xi = 0.77,\ \mathrm{CI}_{95\%}[0.47,\ 0.90],\ n = 32$$

One-sample test: *gghistostats()* method

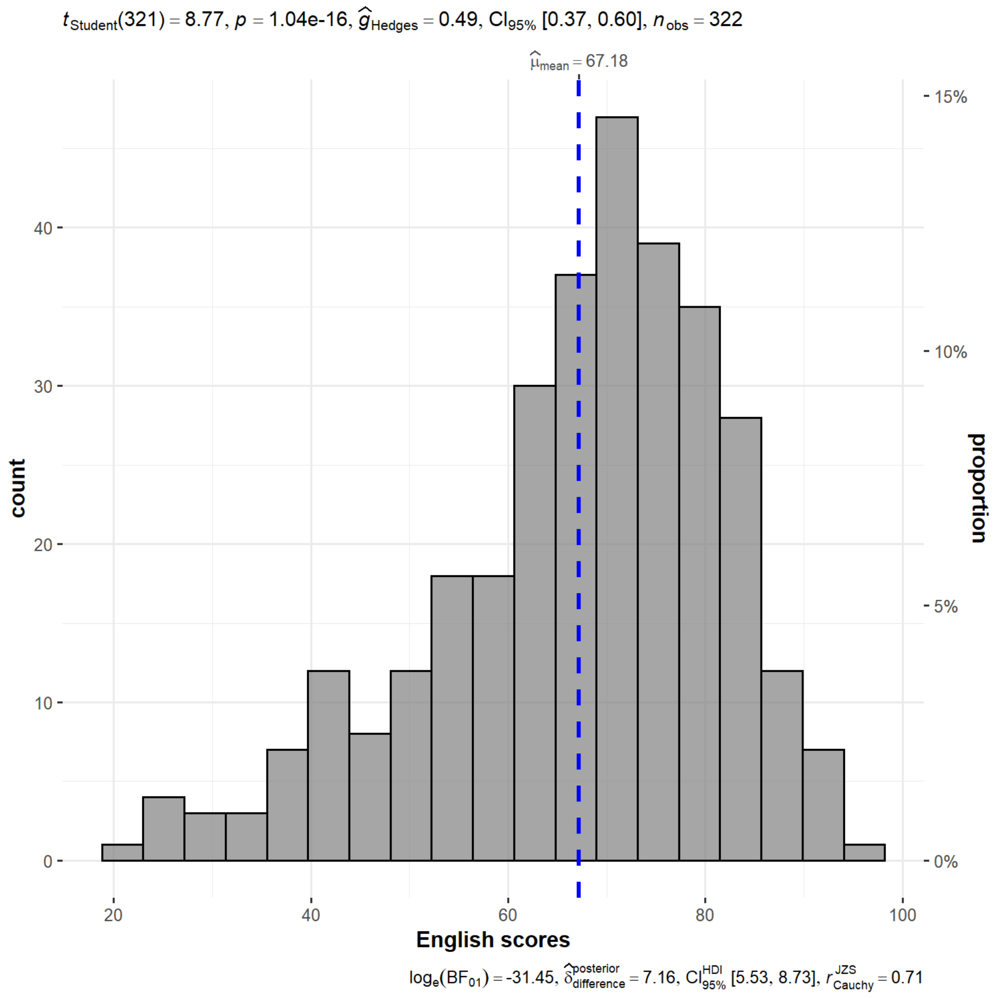
In the code chunk below, *gghistostats()* is used to to build an visual of one-sample test on English scores.

```
set.seed(1234)

gghistostats(
  data = exam,
  x = ENGLISH,
  type = "bayes",
  test.value = 60,
  xlab = "English scores"
)
```

Default information:

- statistical details
- Bayes Factor
- sample sizes
- distribution summary



Unpacking the Bayes Factor

- A Bayes factor is the ratio of the likelihood of one particular hypothesis to the likelihood of another. It can be interpreted as a measure of the strength of evidence in favor of one theory among two competing theories.

- That’s because the Bayes factor gives us a way to evaluate the data in favor of a null hypothesis, and to use external information to do so. It tells us what the weight of the evidence is in favor of a given hypothesis.

- When we are comparing two hypotheses, H1 (the alternate hypothesis) and H0 (the null hypothesis), the Bayes Factor is often written as B10. It can be defined mathematically as

$$\frac{\text{likelihood of data given } H_1}{\text{likelihood of data given } H_0} = \frac{P(D|H_1)}{P(D|H_0)}$$

- The **Schwarz criterion** is one of the easiest ways to calculate rough approximation of the Bayes Factor.

How to interpret Bayes Factor

A **Bayes Factor** can be any positive number. One of the most common interpretations is this one—first proposed by Harold Jeffereys (1961) and slightly modified by Lee and Wagenmakers in 2013:

IF B_{10} IS…	THEN YOU HAVE…
> 100	Extreme evidence for H_1
30 – 100	Very strong evidence for H_1
10 – 30	Strong evidence for H_1
3 – 10	Moderate evidence for H_1
1 – 3	Anecdotal evidence for H_1
1	No evidence
1/3 – 1	Anecdotal evidence for H_1
1/3 – 1/10	Moderate evidence for H_1
1/10 – 1/30	Strong evidence for H_1
1/30 – 1/100	Very strong evidence for H_1
< 1/100	Extreme evidence for H_1

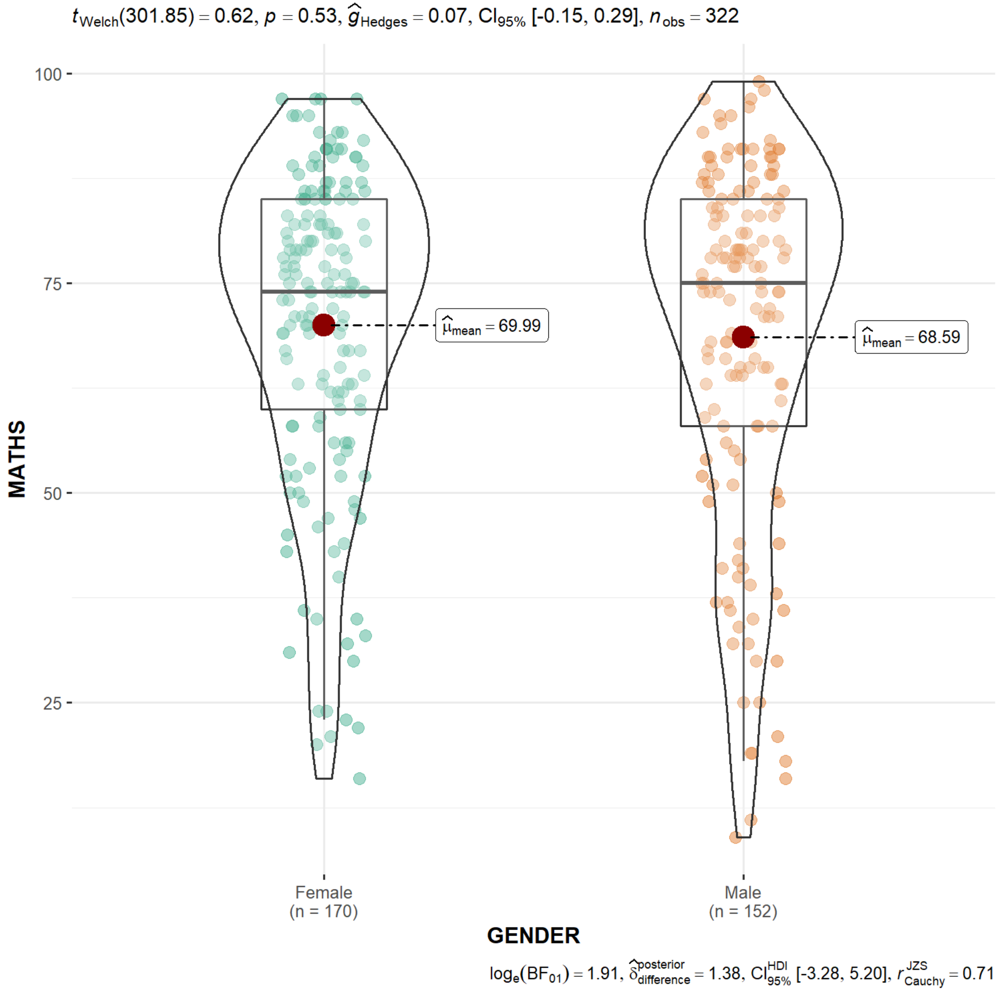
Two-sample mean test: *ggbetweenstats()*

In the code chunk below, *ggbetweenstats()* is used to build a visual for two-sample mean test of Maths scores by gender.

```
ggbetweenstats(
  data = exam,
  x = GENDER,
  y = MATHS,
  type = "np",
  messages = FALSE
)
```

Default information:

- statistical details
- Bayes Factor
- sample sizes
- distribution summary

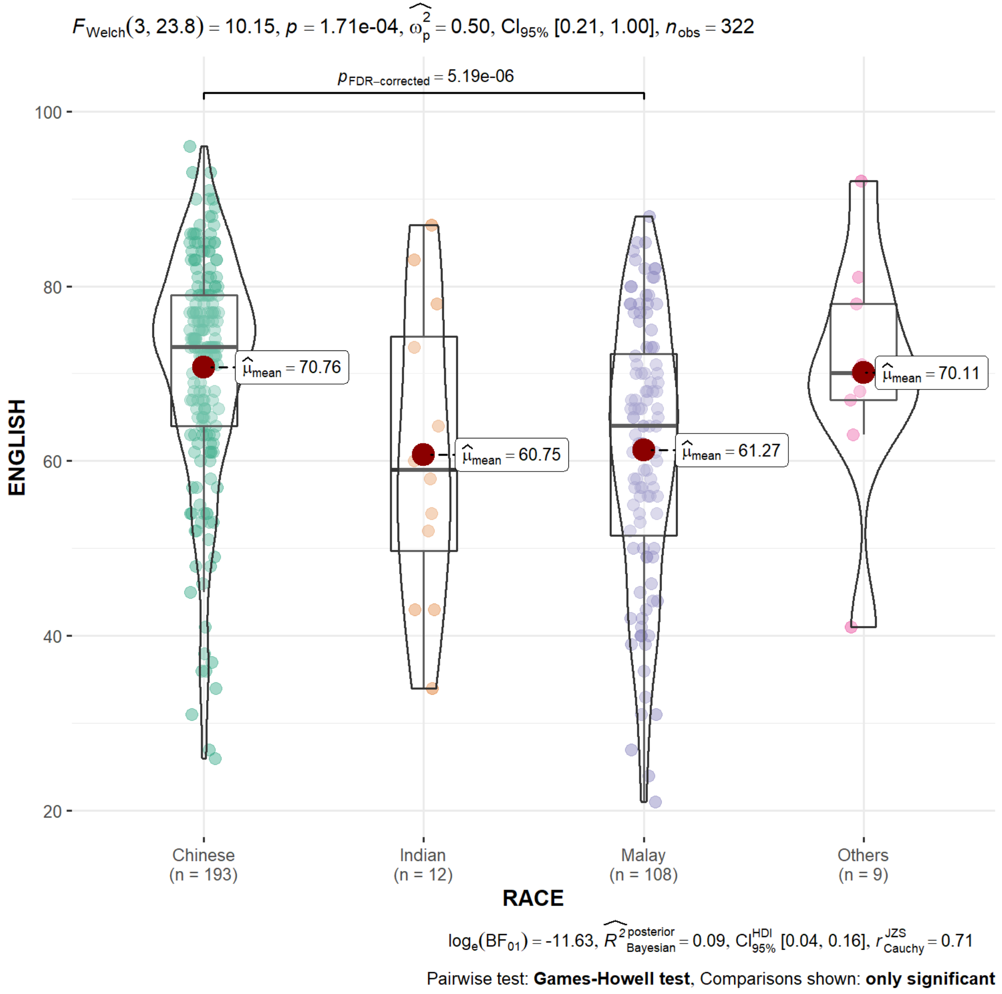


Oneway ANOVA Test: *ggbetweenstats()* method

In the code chunk below, *ggbetweenstats()* is used to build a visual for One-way ANOVA test on English score by race.

```
ggbetweenstats(
  data = exam,
  x = RACE,
  y = ENGLISH,
  type = "p",
  mean.ci = TRUE,
  pairwise.comparisons = TRUE,
  pairwise.display = "s",
  p.adjust.method = "fdr",
  messages = FALSE
)
```

- "ns" → only non-significant
- "s" → only significant
- "all" → everything



ggbetweenstats - Summary of tests

Following (between-subjects) tests are carried out for each type of analyses-

Type	No. of groups	Test
Parametric	> 2	Fisher's or Welch's one-way ANOVA
Non-parametric	> 2	Kruskal–Wallis one-way ANOVA
Robust	> 2	Heteroscedastic one-way ANOVA for trimmed means
Bayes Factor	> 2	Fisher's ANOVA
Parametric	2	Student's or Welch's *t*-test
Non-parametric	2	Mann–Whitney *U* test
Robust	2	Yuen's test for trimmed means
Bayes Factor	2	Student's *t*-test

ggbetweenstats - Summary of tests

Following effect sizes (and confidence intervals/CI) are available for each type of test-

Type	No. of groups	Effect size	CI?
Parametric	> 2	η_p^2, η^2, ω_p^2, ω^2	Yes
Non-parametric	> 2	η_H^2 (*H*-statistic based eta-squared)	Yes
Robust	> 2	ξ (Explanatory measure of effect size)	Yes
Bayes Factor	> 2	No	No
Parametric	2	Cohen's *d*, Hedge's *g* (central-and noncentral-*t* distribution based)	Yes
Non-parametric	2	*r* (computed as $Z/\sqrt{N}$)	Yes
Robust	2	ξ (Explanatory measure of effect size)	Yes
Bayes Factor	2	No	No

ggbetweenstats - Summary of tests

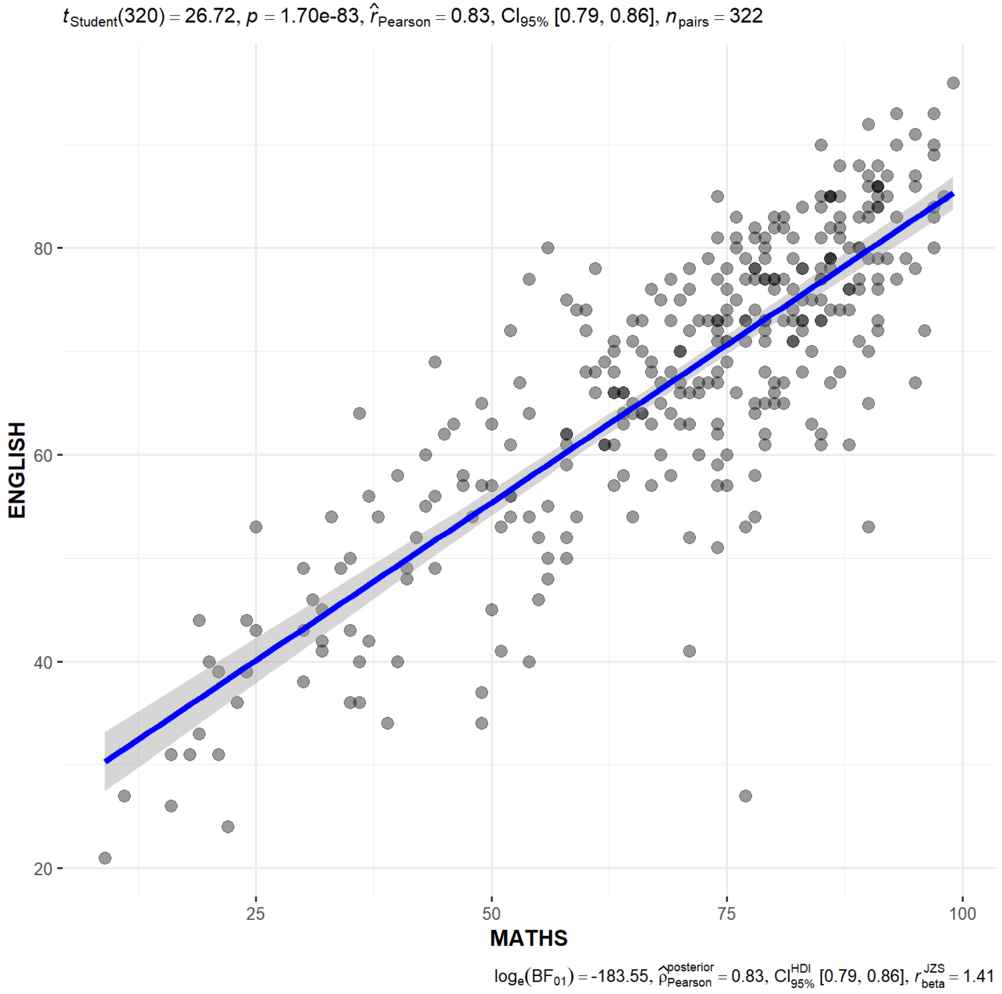
Here is a summary of *multiple pairwise comparison* tests supported in *ggbetweenstats*-

Type	**Equal variance?**	**Test**	***p*-value adjustment?**
Parametric	No	Games-Howell test	Yes
Parametric	Yes	Student’s *t*-test	Yes
Non-parametric	No	Dunn test	Yes
Robust	No	Yuen’s trimmed means test	Yes
Bayes Factor	`NA`	Student’s *t*-test	`NA`

Significant Test of Correlation: *ggscatterstats()*

In the code chunk below, *ggscatterstats()* is used to build a visual for Significant Test of Correlation between Maths scores and English scores.

```
ggscatterstats(
  data = exam,
  x = MATHS,
  y = ENGLISH,
  marginal = FALSE,
  )
```



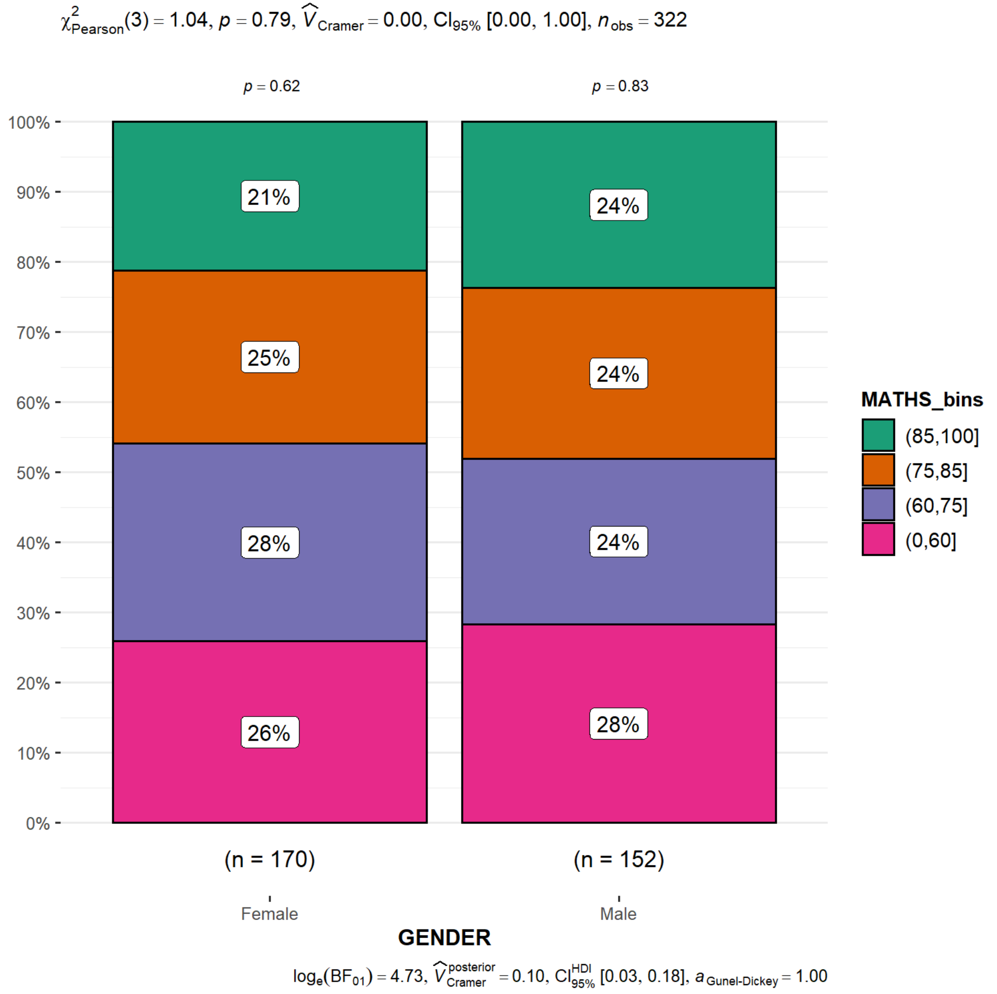
Significant Test of Association (Depedence) : *ggbarstats()* methods

In the code chunk below, the Maths scores is binned into a 4-class variable by using *cut()*.

```
exam1 <- exam %>%
  mutate(MATHS_bins =
           cut(MATHS,
               breaks = c(0,60,75,85,100))
)
```

In this code chunk below *ggbarstats()* is used to build a visual for Significant Test of Association

```
ggbarstats(exam1,
           x = MATHS_bins,
           y = GENDER)
```



Toyota Corolla case study

- Build a model to discover factors affecting prices of used-cars by taking into consideration a set of explanatory variables.

Installing and loading the required libraries

Type the code chunk below to install and launch the necessary R packages

```
packages = c('readxl', 'report', 'performance',
             'parameters', 'see')

for(p in packages){
  if(!require(p, character.only = T)){
    install.packages(p)
  }
  library(p, character.only = T)
}
```

Importing Excel file: readxl methods

In the code chunk below, *read_xls()* of **readxl** package is used to import the data worksheet of `ToyotaCorolla.xls` workbook into R.

```
car_resale <- read_xls("data/ToyotaCorolla.xls",
                       "data")
```

Notice that the output object `car_resale` is a tibble data frame.

Multiple Regression Model using lm()

The code chunk below is used to calibrate a multiple linear regression model by using *lm()* of Base Stats of R.

```
model <- lm(Price ~ Age_08_04 + Mfg_Year + KM +
              Weight + Guarantee_Period, data = car_resale)
model
```

```
##
## Call:
## lm(formula = Price ~ Age_08_04 + Mfg_Year + KM + Weight + Guarantee_Period,
##     data = car_resale)
##
## Coefficients:
##      (Intercept)         Age_08_04          Mfg_Year                KM
##       -2.637e+06        -1.409e+01         1.315e+03        -2.323e-02
##           Weight  Guarantee_Period
##        1.903e+01         2.770e+01
```

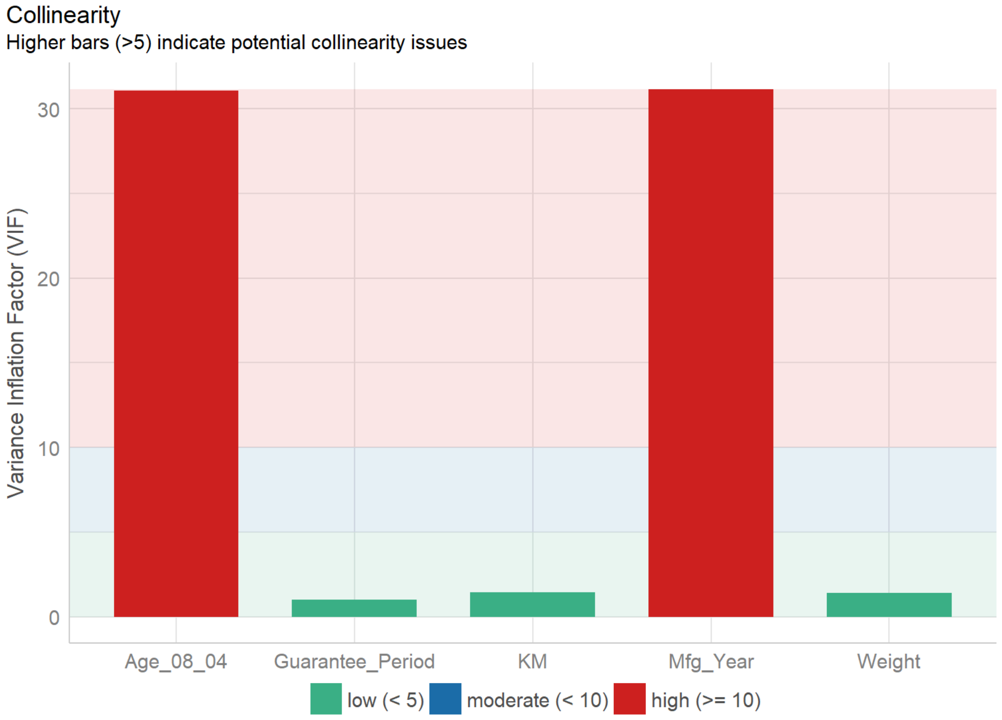
Model Diagnostic: checking for multicolinearity:

In the code chunk, *check_collinearity()* of **performance** package.

```
check_collinearity(model)
```

```
## # Check for Multicollinearity
##
## Low Correlation
##
##               Term  VIF Increased SE Tolerance
##                 KM 1.46         1.21      0.68
##             Weight 1.41         1.19      0.71
##   Guarantee_Period 1.04         1.02      0.97
##
## High Correlation
##
##        Term   VIF Increased SE Tolerance
##   Age_08_04 31.07         5.57      0.03
##    Mfg_Year 31.16         5.58      0.03
```

```
check_c <- check_collinearity(model)
plot(check_c)
```

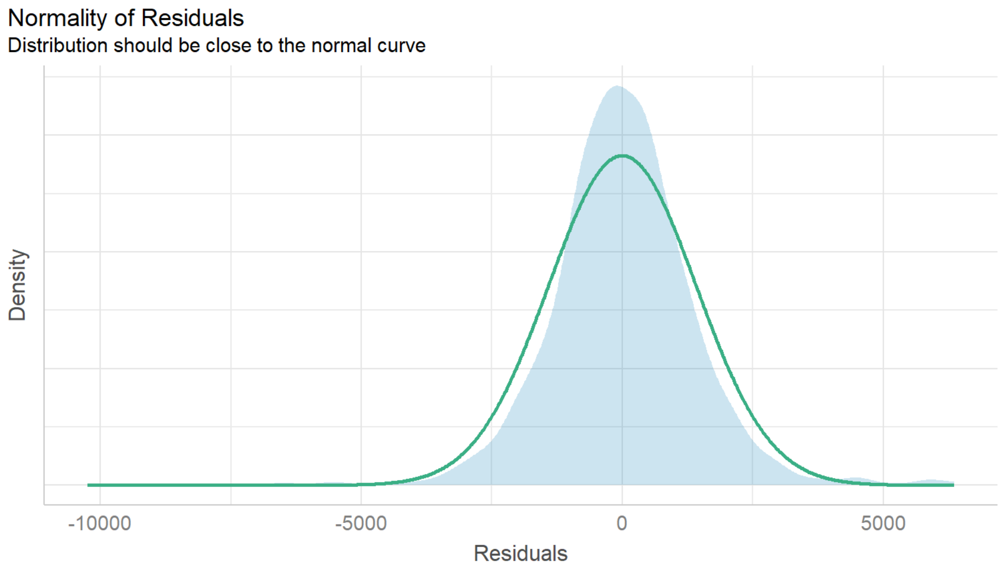


Model Diagnostic: checking normality assumption

In the code chunk, *check_normality()* of **performance** package.

```
check_n <- check_normality(model1)
```

```
plot(check_n)
```

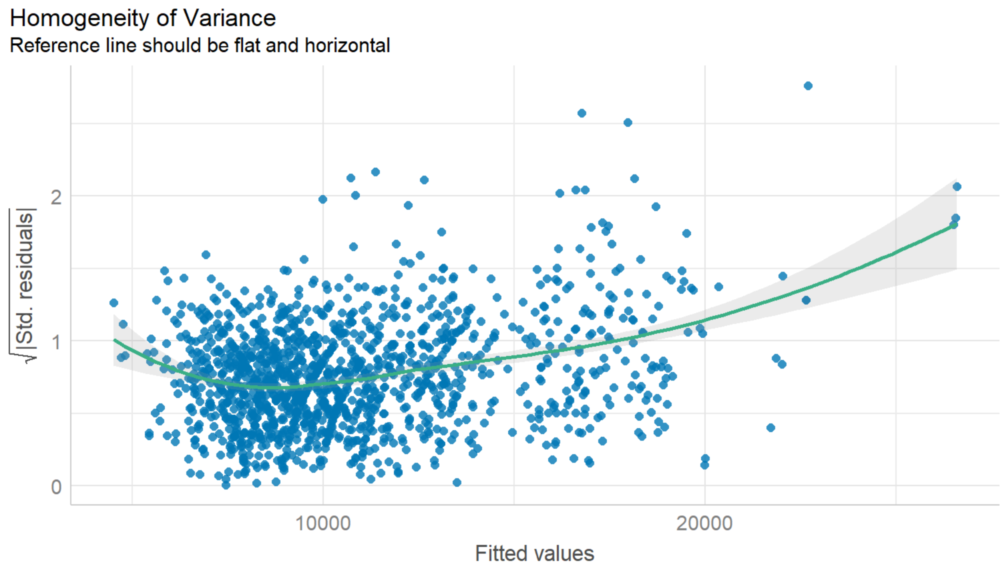


Model Diagnostic: Check model for homogeneity of variances

In the code chunk, *check_heteroscedasticity()* of **performance** package.

```
check_h <- check_heteroscedasticity(model1)
```

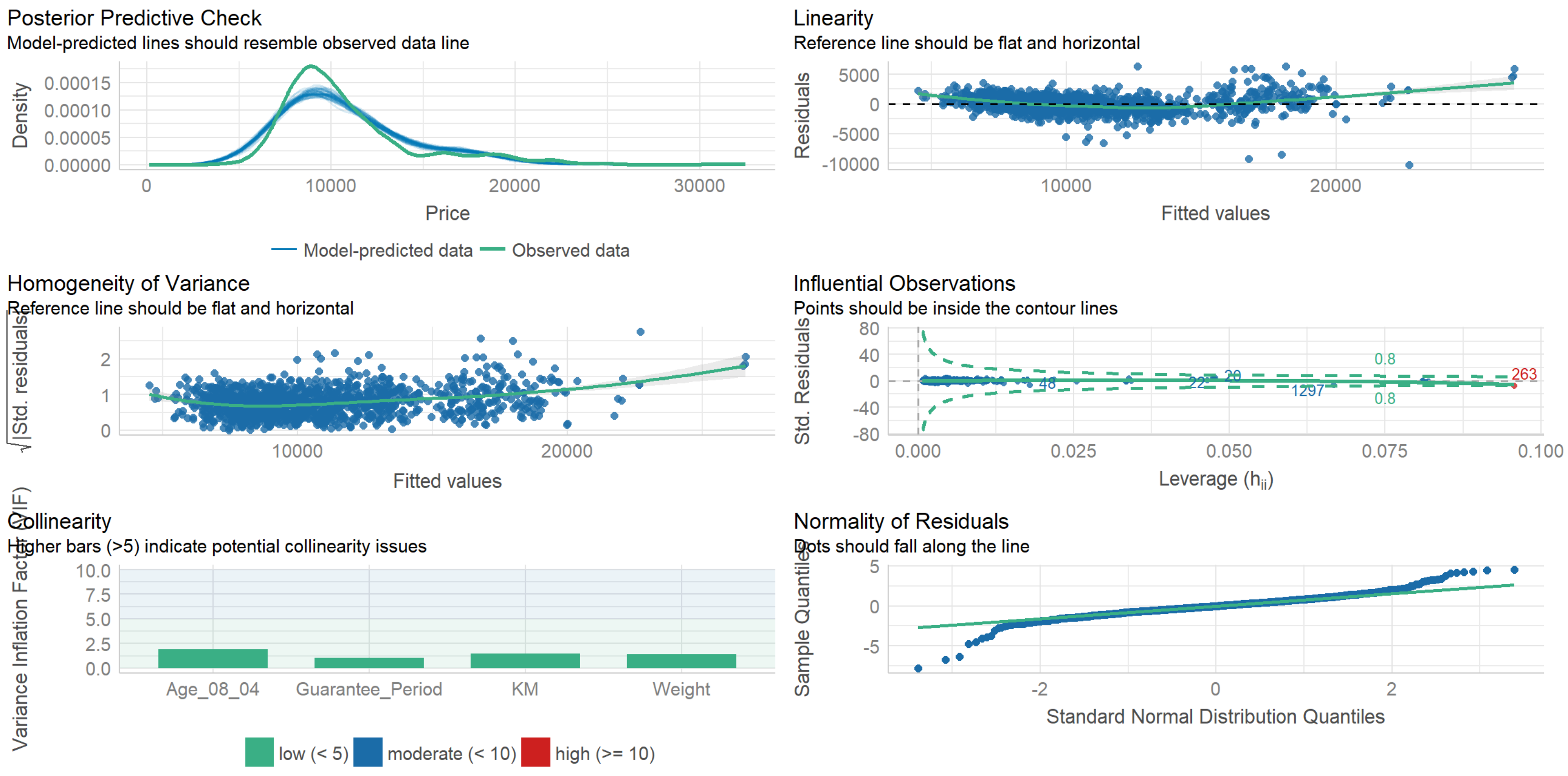
```
plot(check_h)
```



Model Diagnostic: Complete check

We can also perform the complete by using *check_model()*.

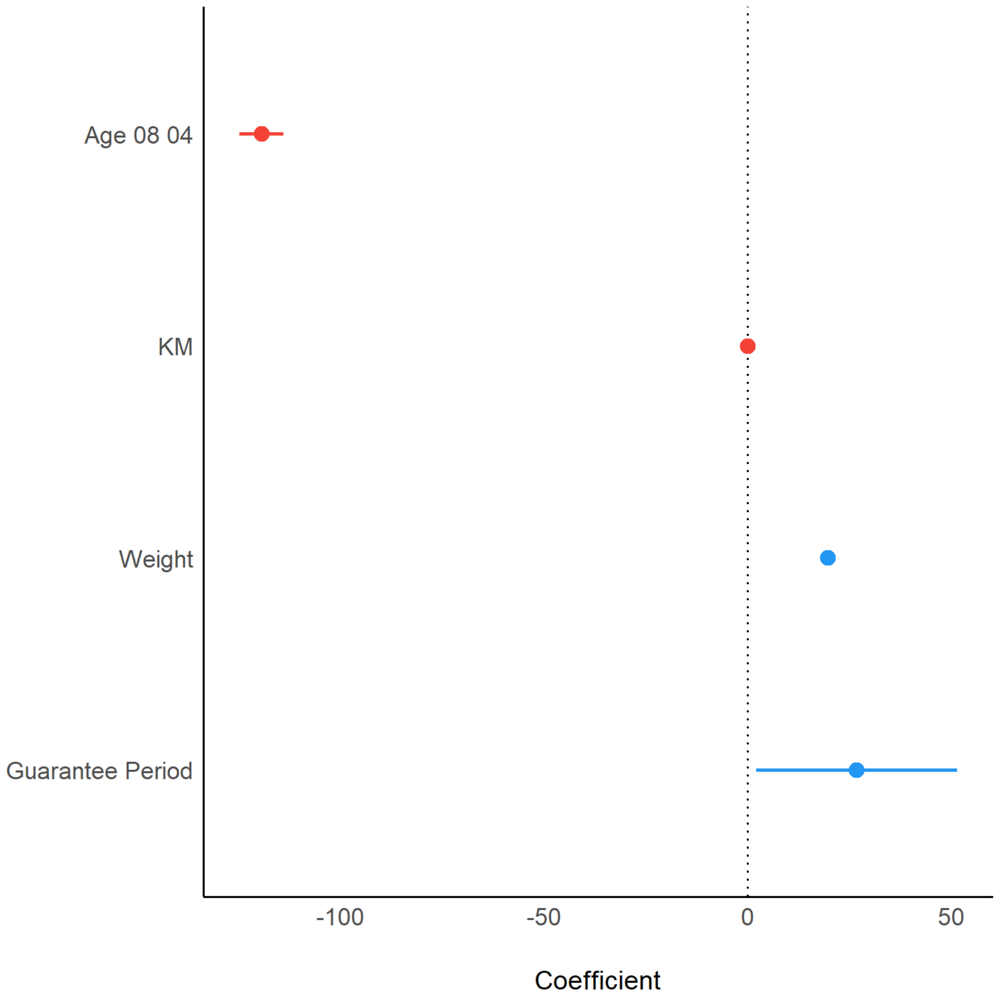
```
check_model(model1)
```



Visualising Regression Parameters: see methods

In the code below, plot() of see package and parameters() of parameters package is used to visualise the parameters of a regression model.

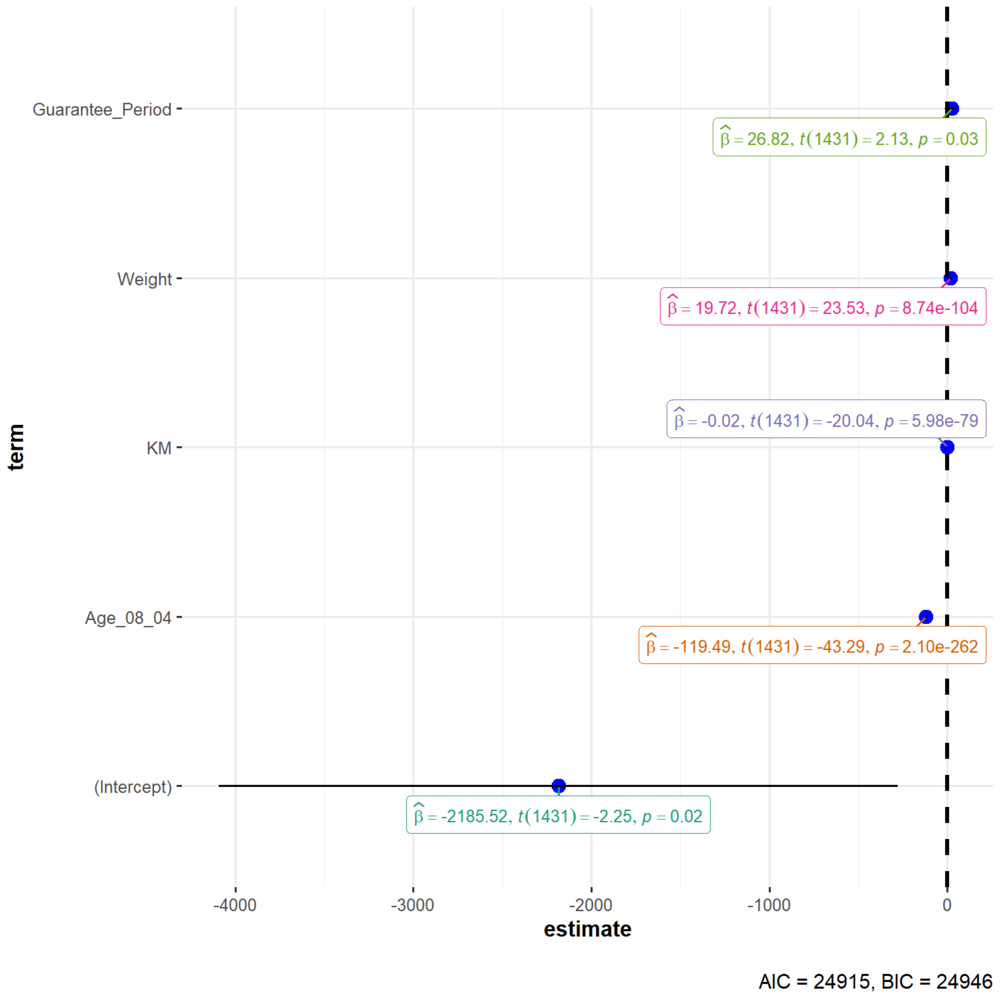
```
plot(parameters(model1))
```



Visualising Regression Parameters: *ggcoefstats()* methods

In the code below, *ggcoefstats()* of ggstatsplot package to visualise the parameters of a regression model.

```
ggcoefstats(model1,
            output = "plot")
```



Visualizing the uncertainty of point estimates

- A point estimate is a single number, such as a mean.
- Uncertainty is expressed as standard error, confidence interval, or credible interval
- Important:
  - Don't confuse the uncertainty of a point estimate with the variation in the sample

Visualizing the uncertainty of point estimates: ggplot2 methods

The code chubk below computes the count of observations, mean, standard deviation and standard error of a variable.

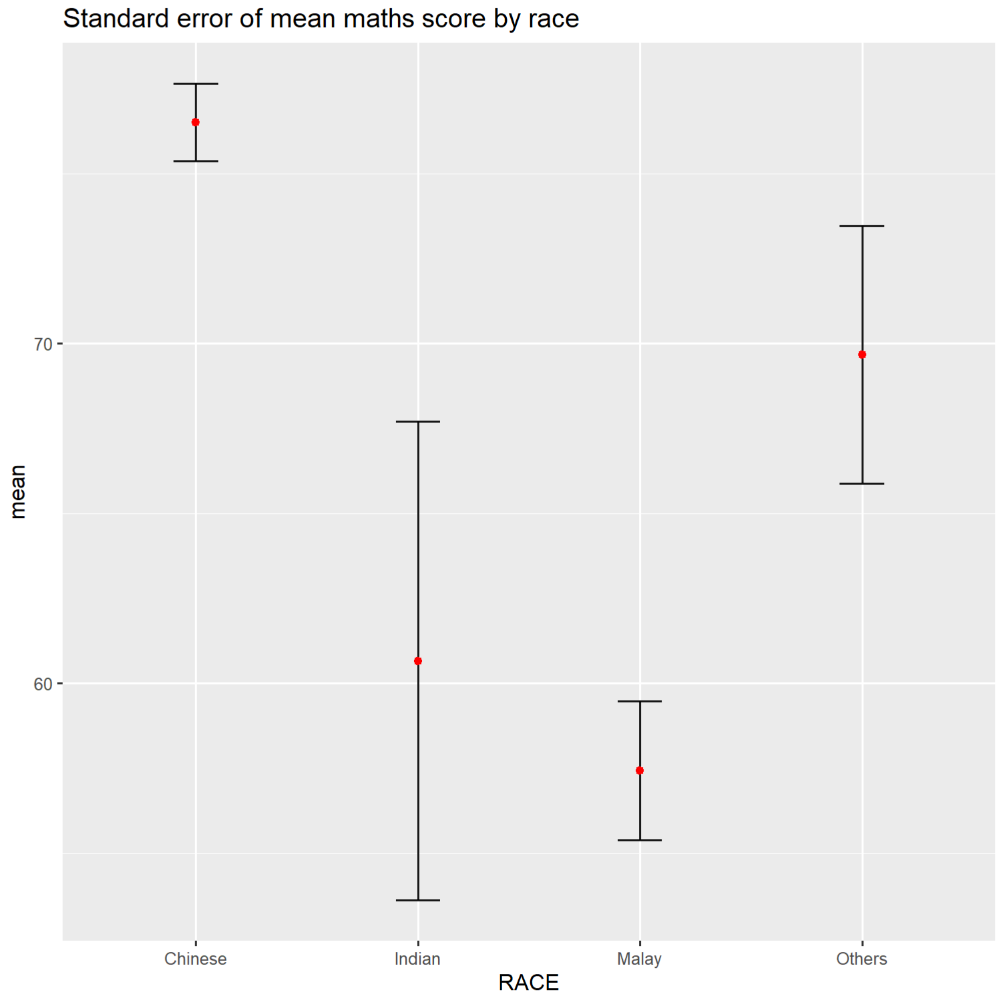
```
my_sum <- exam %>%
  group_by(RACE) %>%
  summarise(
    n=n(),
    mean=mean(MATHS),
    sd=sd(MATHS)
    ) %>%
  mutate(se=sd/sqrt(n-1))
```

Note: For the mathematical explanation, please refer to Slide 20 of Lesson 4.

Visualizing the uncertainty of point estimates: ggplot2 methods

The code chunk below is used to reveal the standard error of mean maths score by race .

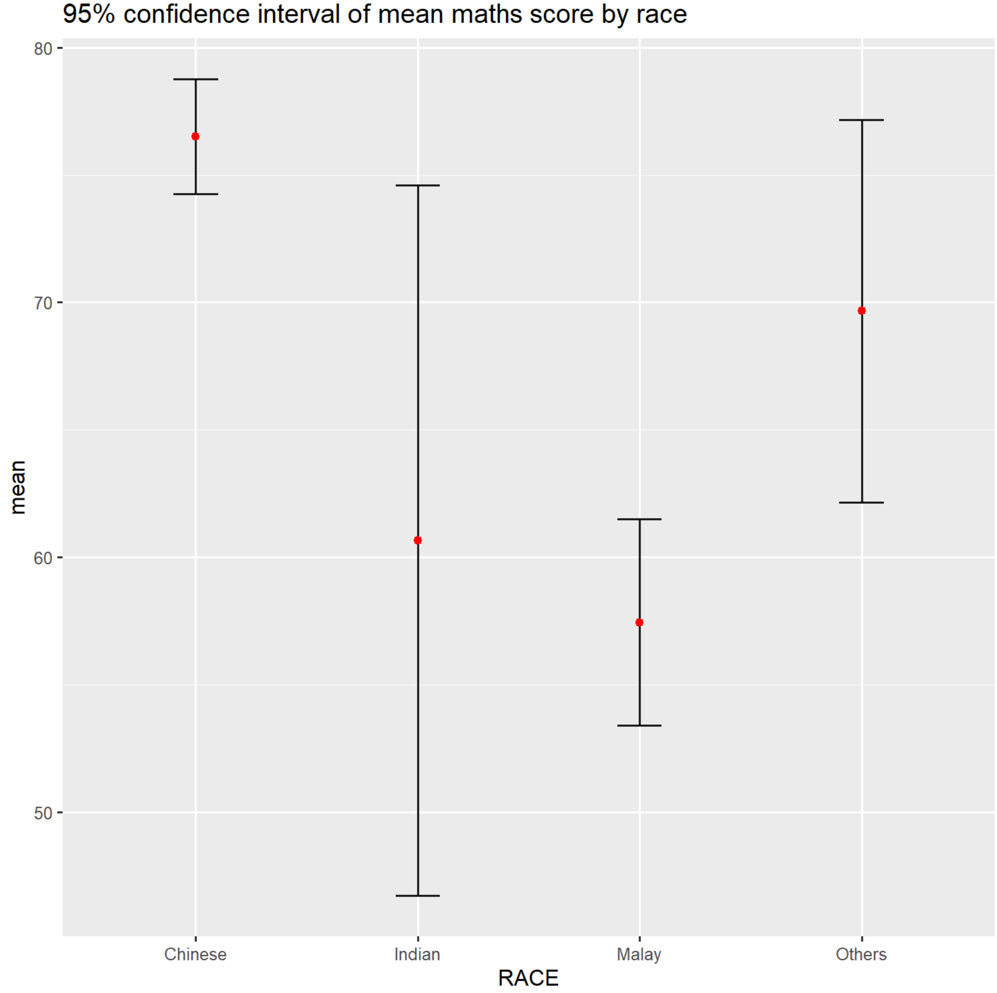
```
ggplot(my_sum) +
  geom_errorbar(
    aes(x=RACE,
        ymin=mean-se,
        ymax=mean+se),
    width=0.2,
    colour="black",
    alpha=0.9,
    size=0.5) +
  geom_point(aes
           (x=RACE,
            y=mean),
           stat="identity",
           color="red",
           size = 1.5,
           alpha=1) +
  ggtitle("Standard error of mean
          maths score by rac")
```



Visualizing the uncertainty of point estimates: ggplot2 methods

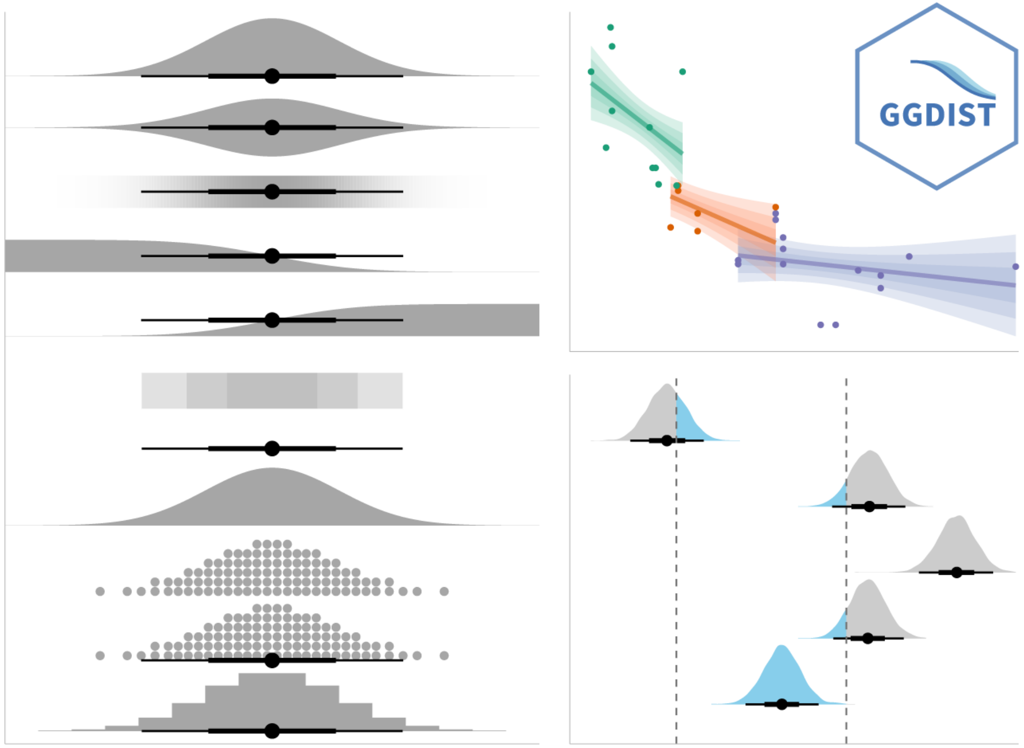
To plot the 95% confidence interval of mean maths score by race, we just need to modify the codes highlighted in the code chunk below.

```
ggplot(my_sum) +
  geom_errorbar(
    aes(x=RACE,
        ymin=mean-1.98*se,
        ymax=mean+1.98*se),
    width=0.2,
    colour="black",
    alpha=0.9,
    size=0.5) +
  geom_point(aes
           (x=RACE,
            y=mean),
           stat="identity",
           color="red",
           size = 1.5,
           alpha=1) +
  ggtitle("95% confidence interval
          of mean maths score by race")
```



Visualising Uncertainty: ggdist package

- **ggdist** is an R package that provides a flexible set of ggplot2 geoms and stats designed especially for visualising distributions and uncertainty.
- It is designed for both frequentist and Bayesian uncertainty visualization, taking the view that uncertainty visualization can be unified through the perspective of distribution visualization:
  - for frequentist models, one visualises confidence distributions or bootstrap distributions (see vignette("freq-uncertainty-vis"));
  - for Bayesian models, one visualises probability distributions (see the tidybayes package, which builds on top of ggdist).

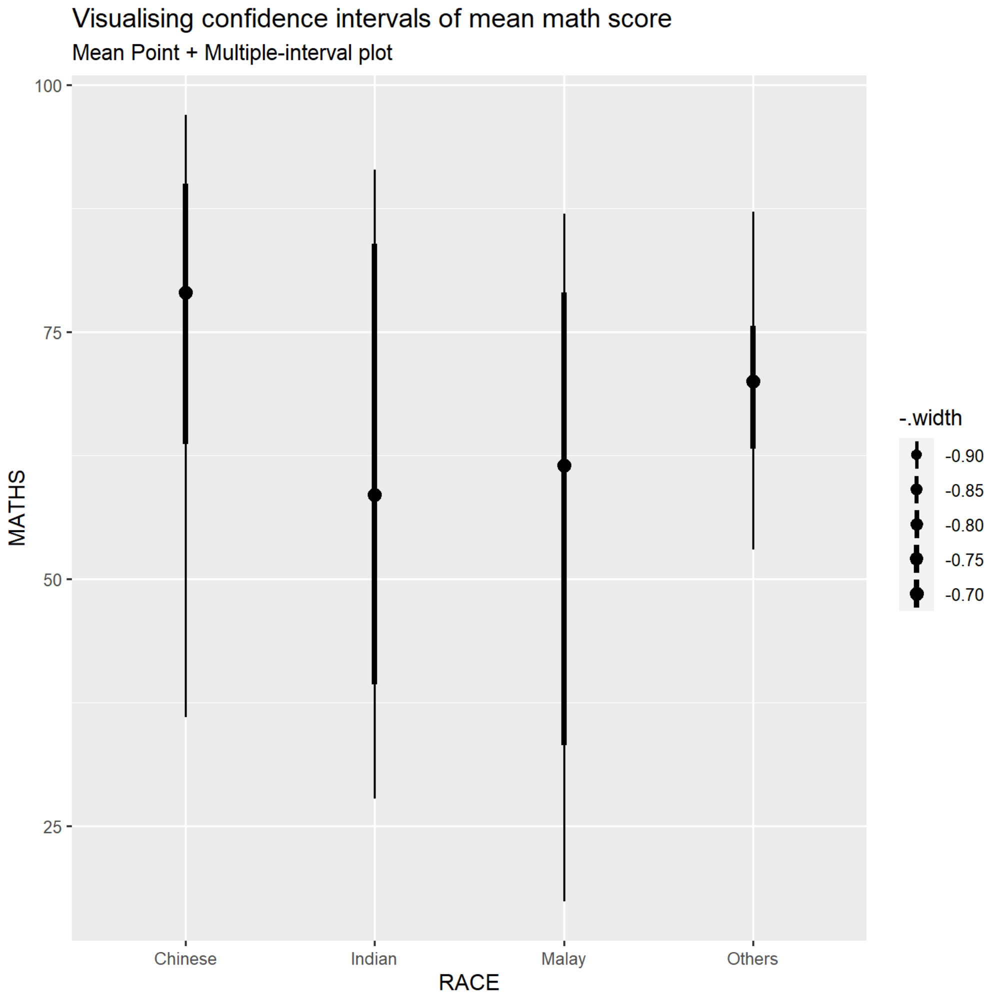


Visualizing the uncertainty of point estimates: ggdist methods

In the code chunk below, `stat_pointinterval()` of **ggdist** is used to build a visual for displaying distribution of maths scores by race.

```
exam %>%
  ggplot(aes(x = RACE,
             y = MATHS)) +
  stat_pointinterval(
    show.legend = TRUE) +
  labs(
    title = "Visualising confidence intervals o
    subtitle = "Mean Point + Multiple-interval
```

Gentle advice: This function comes with many arguments, students are advised to read the syntax reference for more detail.

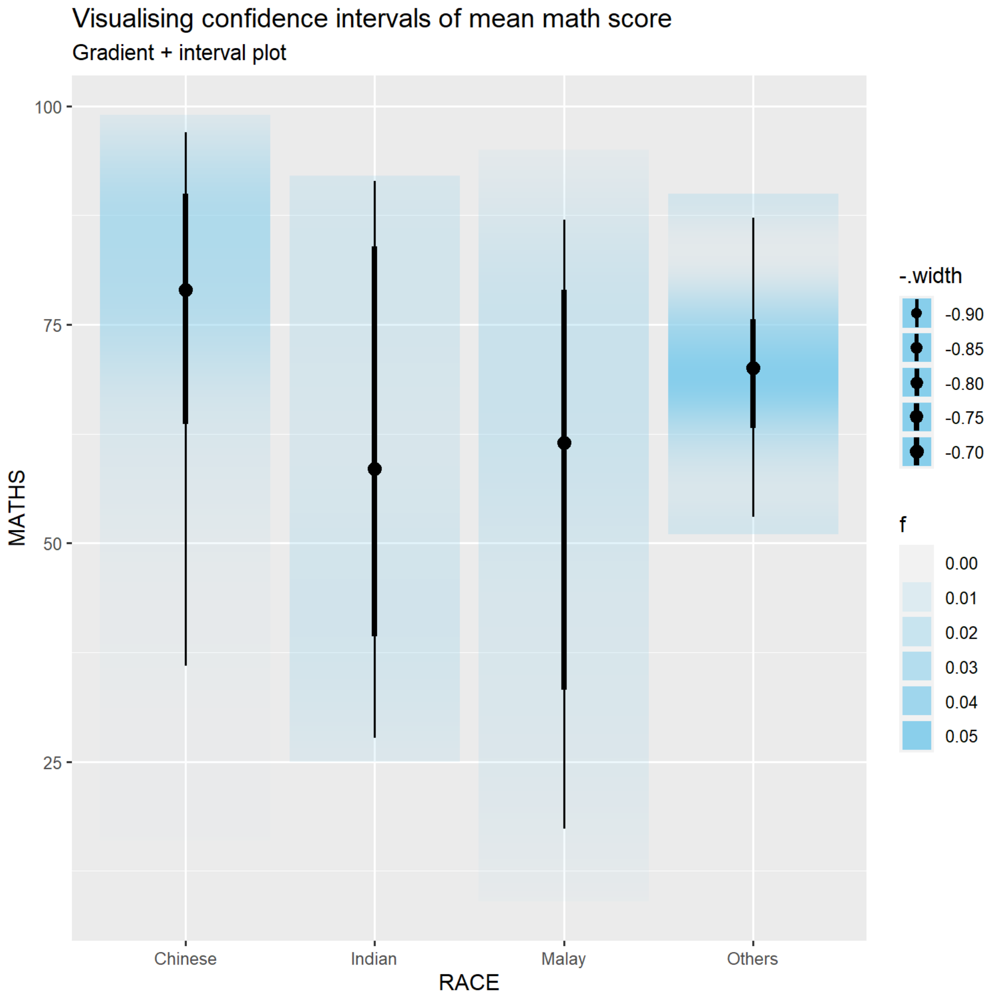


Visualizing the uncertainty of point estimates: ggdist methods

In the code chunk below, `stat_gradientinterval()` of **ggdist** is used to build a visual for displaying distribution of maths scores by race.

```
exam %>%
  ggplot(aes(x = RACE,
             y = MATHS)) +
  stat_gradientinterval(
    fill = "skyblue",
    show.legend = TRUE
  ) +
  labs(
    title = "Visualising confidence intervals o
    subtitle = "Gradient + interval plot")
```

Gentle advice: This function comes with many arguments, students are advised to read the syntax reference for more detail.



Visualising Uncertainty with Hypothetical Outcome Plots (HOPs)

Step 1: Installing ungeviz package

```
devtools::install_github("wilkelab/ungeviz")
```

Note: You only need to perform this step once.

Step 2: Launch the application in R

```
library(ungeviz)
```