pstat126 hw2

Haocheng Zhang

2023-02-06

Show the table with the given command.

Cereal <- read.csv("cereal.csv",header=T)

str(Cereal)

Since we are required to focus on 9 varaibles, to make it easier, I'll make a new dataset named 'new_cerel' with the selected 9 variables only.

new_cereal <- Cereal[, c("rating", "protein", "fat", "fiber", "carbo", "sugars", "potass", "vitamins", "cups")]

Q(a)

Firtly, use summary() function to have basic statistics for each variable.

summary(new_cereal)

Secondly, execute the histograms function for each variable.

par(mfrow=c(3,3))

for(i in 1:ncol(new_cereal)){ hist(new_cereal[,i], main=colnames(new_cereal)[i], xlab="", ylab="") }

Scatter plots for each variable pair

pairs(new_cereal)

Correlation matrix

cor(new_cereal)

By observing all the outputs above, there is not obvious outlier.

Q(b) Use the lm function in R to fit the MLR model with rating as the response and the other 8 variables as predictors.

```
model <- lm(rating ~ protein + fat + fiber + carbo + sugars + potass + vitamins + cups, data = new_cereal)
```

Display the summary output.

```
summary(model)
```

Q(c) By observing the summary output above, the predictor variables that are statistically significant at a 0.01 significance level are fat, fiber, sugars, and vitamins.

Q(d) By observing the summary output aboe, the multiple R-squared is 0.9037, which means 90.37% of the total variation in the response is explaned.

Q(e) The null hypothesis states that the regression model has no significant relationship between the predictors and the response variables. And from the summary output above, the p-value for the global F-test is 2.2e-16. Besides the adjusted R-squared value of 0.8923 indicates that 89.23% of the variation in the response variable is explained by the predictor variables after adjusting for the number of predictors in the model. These values suggest that the predictor variables do explain a significant proportion of the variation in the response variable.

Q(f) Use the t distribution with n-2 degrees of freedom, which is 70-2=68. and from the summary above, use the t-value -0.110. Then use the pt() function to calculate the p-value.

```
p_value_carbo <- 2*pt(abs(-0.110), df = 68, lower.tail = FALSE)
p_value_carbo
```

The answer is 0.9127334. And the computed p-value is identical to the p-value provided in the summary output.

Q(g) The corresponding null hypothesis of the statistical test is that neither vitamins nor potass have any relation to the response rating. To test this hypothesis, we can use an F-test to compare the full model with all predictors to a reduced model without the vitamins and potass predictors with anova() function.

reduced_model <- lm(rating ~ protein + fat + fiber + carbo + sugars + cups, data = new_cereal)

full_model <- lm(rating ~ protein + fat + fiber + carbo + sugars + potass + vitamins + cups, data = new_cereal)

anova(full_model, reduced_model)

The output shows that the p-value for the F-test above is 0.0004076, which is very small compared with 0.01 and 0.05, indicating strong evidence against the null hypothesis that neither vitamins nor potass have relations to the response 'rating'. Therefore, we can reject the null hypothesis and conclude that at least one of the predictors, either vitamins or potass is significantly related to the response 'rating'.

Q(h) from the summaru output in Q(b), we use confint() function to construct the confidence intervak for 'protein' at a 99% confidence interval.

confint(model, "protein", level = 0.99)

From the outcome, we can be 99% confident to say that the value of the protein falls within the interval from 0.2017 to 3.7227.

Q(i) Construct a new dataframe named new_data, with the provided data for each variable

new_data <- data.frame(protein = 3, fat = 5, fiber = 2, carbo = 13, sugars = 6, potass = 60, vitamins = 25, cups = 0.8)

Then use pridict() function, and use the model in q(b) to make a prediction based on new_data

predicted_rating <- predict(model, newdata = new_data)

predicted_rating

Q(j) Using the datafram in Q(i) with predict() function again to constaurct a prediction interval.

predicted_interval <- predict(model, new_data, interval = "prediction", level = 0.95)

predicted_interval