GEOG 714 - Assignment 5

Haoran Xu

Oct 28, 2024

```
v <- c(1, 4, 3, 2, 6, 5, 4, 5)
x_bar <- mean(v)
lower <- x_bar - 1.96 * (sqrt(var(v)))/sqrt(length(v))
upper <- x_bar + 1.96 * (sqrt(var(v)))/sqrt(length(v))

NUMBER_OBS <- 1000
m <- matrix(0, nrow = NUMBER_OBS, ncol = 3)
for (i in 1:NUMBER_OBS) {
    m[i, 2] <- rnorm(1)
    m[i, 3] <- rnorm(1)
    m[i, 1] <- 2 * m[i, 2] - 1.5 * m[i, 3] + rnorm(1)
}

df <- as.data.frame(m)
colnames(df) <- c("Y", "x1", "x2")
X <- as.matrix(cbind(1, df$x1, df$x2))
Y <- as.matrix(cbind(df$Y))
b <- solve(t(X) %*% X) %*% t(X) %*% Y
```

Q1. Looking at this code, figure out the true expected coefficients of 0, 1, 2. You may need to read up on the rnorm() function. Explain your reasoning.

First of all, r(norm) generates random numbers that follow a N(0, 1) normal distrubution with a mean of 0. This makes the true 1, 2 to be 2 and -1.5 respectively, which are the coefficients assigned for creating m[i, 1] (the y-value). The true 0 would be 0, as the mean for rnorm(1) would be zero. So the true expected coefficients of 0, 1, 2 should be **0, 2, -1.5**, respectively.

```
df$p <- b[1] * X[, 1] + b[2] * X[, 2] + b[3] * X[, 3]  # fitted (predicted) values
df$r <- Y - df$p  # model error (here we did not write df$[Y], instead, a 'Y')
```

Q2. Explain why the predictions in p are not the same as the values of Y (in less than two sentences). Hint: look at the equations we used in the data synthesis step.

Because for the p predictions we use the predicted 0, 1, 2 values, which are estimated based on the 1000 observations. Whereas the real y-values are calculated by the true coefficients of 0, 1, 2.

```r
lmout <- lm(df$Y ~ df$x1 + df$x2)
summary(lmout)
df$p2 <- lmout$fitted.values  # same values as the above (these steps avoid typing formulas)
df$r2 <- lmout$residuals

p <- plot(df$p2, df$Y, col = "red", cex = 0.5, pch = 19)  # here assign it as p
p <- p + abline(lm(df$Y ~ df$p), lwd = 2, col = "blue")  # draw the blue lines
```

```r
p
```

```r
NUMBER_OBS <- 1000
m2 <- matrix(0, nrow = NUMBER_OBS, ncol = 3)
for (i in 1:NUMBER_OBS) {
    m2[i, 2] <- rnorm(1)
    m2[i, 3] <- rnorm(1)
    m2[i, 1] <- 2 * m2[i, 2] - 1 * m2[i, 3] + 2 * m2[i, 3]^2 + rnorm(1)
}
df2 <- as.data.frame(m2)
colnames(df2) <- c("Y", "x1", "x2")

lmout2 <- lm(df2$Y ~ df2$x1 + df2$x2)
summary(lmout2)
```

Q3. Provide a brief description of the above results from the regression analysis (no more than two sentences). Note: Your results will not be exactly the same as above, but similar.

The method of linear regression modelling estimated the relationships between Y, x1, x2 to be $\mathbf{Y = 1.8338x1 - 0.9052x2 + 2.1178}$ and all the estimators are significant (p-value < 2e-16). As the Adjusted R-squared suggests, the model explains **28.05%** of the variances in Y.

```r
df2$p <- lmout2$fitted.values  # predicted values

p2 <- plot(df2$p, df2$Y, col = "red", cex = 0.5, pch = 19)
p2 <- p2 + abline(lm(df2$Y ~ df2$p), lwd = 2, col = "blue")
```

```r
p2
```

Q4. Plot X1 variable by Y and the X2 variable by Y and comment on the plots produced (in less than two sentences)

```r
p2_1 <- plot(df2$x1, df2$Y, col = "red", cex = 0.5, pch = 19)
p2_1 <- p2_1 + abline(lm(df2$Y ~ df2$x1), lwd = 2, col = "blue")
```

```r
p2_1

p2_2 <- plot(df2$x2, df2$Y, col = "red", cex = 0.5, pch = 19)
p2_2 <- p2_2 + abline(lm(df2$Y ~ df2$x2), lwd = 2, col = "blue")
```

```
p2_2
```

The graph of X1 variable by Y shows many red dots which are closely situated above and below positively the regressed blue straight line. While the graph of X2 variable by Y shows many red dots following a positively curly line which do not fall closely to the regressed blue stright line.

```
df2$x2sq <- df2$x2^2
lmout2_1 <- lm(df2$Y ~ df2$x1 + df2$x2sq)
summary(lmout2_1)
```

Q5. Generate code that creates a new vector of predictions/fitted values and plots them on a graph along with the observed values of Y. Provide an interpretation of this graph.

```
df2$p_2 <- lmout2_1$fitted.values
plot(df2$p_2, df2$Y, col = "red", cex = 0.5, pch = 19)
abline(lm(df2$Y ~ df2$p_2), lwd = 2, col = "blue")  # you can do separate line adding
```

Since we use the square form (x2^2) of x2 in the updated linear regression modelling, now the fitted (predicted) values fit closely alongside the regressed blue line. The explanation rate (86.07%) shown as the adjusted R-squared has been greatly improved.

```
NUMBER_OBS <- 1000
m3 <- matrix(0, nrow = NUMBER_OBS, ncol = 3)
for (i in 1:NUMBER_OBS) {
    m3[i, 2] <- rnorm(1)
    m3[i, 3] <- round(abs(runif(1) * 2))
    m3[i, 1] <- 2 * m3[i, 2] + 3 * m3[i, 3] + rnorm(1)
}

df3 <- as.data.frame(m3)
colnames(df3) <- c("Y", "x1", "x2")

df3$x2 <- as.character(df3$x2)
df3$x2[df3$x2 == "0"] <- "Uncertain"
df3$x2[df3$x2 == "1"] <- "Certain yes"
df3$x2[df3$x2 == "2"] <- "Certain no"
df3$x2 <- as.factor(df3$x2)

lmout3 <- lm(df3$Y ~ df3$x1 + df3$x2)
summary(lmout3)

# Special Characters that have meanings in r: NA, NULL, Inf, -Inf, NaN
```

Q6. Provide an interpretation of the results of the model. Pay careful attention to how you interpret the categorical variable.

When encountering a factor (categorical) variable in regression modeling, the factor variable would be turned into **dummy variables**. Here, the three levels of x2 variable would be all turned into dummy variables

(0 or 1), except for when `x2 == "Certain no"` as it is the first one in alphabetical orders. The estimated intercept (6.06568) was estimated when the continuous variable `x1` equals 0 and the factor variable `x2` is in its baseline category "Certain no". The estimated coefficients show that `x1` has a significant positive effect on `Y` (1.976), which means every unit increase in `x1`, `Y` increases by approximately 1.976 if `x2` is the same. The factor variable `x2$` indicates that compared to the baseline category "Certain no," being in the "Certain yes" or "Uncertain" category would significantly reduce `Y` by 3.101 and 6.049 units, respectively.The model explains 88.49% of the variance in `Y`, with a highly significant F-statistic (p-value $<$ 2.2e-16), suggesting a good fit.

```
NUMBER_OBS <- 2000
m4 <- matrix(0, nrow = NUMBER_OBS, ncol = 4)
for (i in 1:NUMBER_OBS) {
    m4[i, 2] <- rnorm(1)
    m4[i, 3] <- rnorm(1)
    m4[i, 4] <- round(abs(runif(1)))
    m4[i, 1] <- 2 * m4[i, 2] + 3 * m4[i, 3] + 0.5 * m4[i, 4] + -2 * m4[i, 3] * m4[i,
        4] + rnorm(1)  # x2 and x3 are associated.
}

df4 <- as.data.frame(m4)
colnames(df4) <- c("Y", "x1", "x2", "x3")

lmout4_1 <- lm(df4$Y ~ df4$x1 + df4$x2 + df4$x3)
summary(lmout4_1)

df4$x3 <- as.character(df4$x3)
df4$x3[df4$x3 == "0"] <- "no"
df4$x3[df4$x3 == "1"] <- "yes"
df4$x3 <- as.factor(df4$x3)

boxplot(df4$Y[df4$x2 > mean(df4$x2)] ~ df4$x3[df4$x2 > mean(df4$x2)], main = "X2 above mean",
    col = "red")
```

```
boxplot(df4$Y[df4$x2 < mean(df4$x2)] ~ df4$x3[df4$x2 < mean(df4$x2)], main = "X2 below mean",
    col = "blue")
```

```
lmout4_2 <- lm(df4$Y ~ df4$x1 + df4$x2 + df4$x3 + df4$x2 * df4$x3)
summary(lmout4_2)
```

Q7. Find a way to compare these two model results in a way that illustrates the role of the interaction in predicting values of Y. Graphs/plots are preferred over tables and numbers.

```
df4$pred1 <- predict(lmout4_1)
df4$pred2 <- predict(lmout4_2)

par(mfrow = c(1, 2))
plot(df4$Y, df4$pred1, main = "Model 1 (No Interaction)", xlab = "Observed Y", ylab = "Predicted Y",
    col = "blue")
abline(0, 1, col = "red", lwd = 2)
```

```r
plot(df4$Y, df4$pred2, main = "Model 2 (With Interaction)", xlab = "Observed Y",
    ylab = "Predicted Y", col = "green")
abline(0, 1, col = "red", lwd = 2)
```

It can be seen from the two graphs above that the one with interation (that incorporates the `df4$x2 * df4$x3`) has all of the dots more closely aligned to the red line above and below. While the one without interation has more distributed dots spread over the red line, suggesting a poorer fit.

```r
age <- sample(c(20, 30, 40, 50, 60, 70, 80, 90, 100), 1000, replace = TRUE)
income_1000s <- sample(c(20, 30, 40, 50, 60, 70, 80, 90, 100), 1000, replace = TRUE)
education <- sample(c(1, 2, 3, 4, 5), 1000, replace = TRUE)
linear_form <- 22 + age * -0.2 + income_1000s * -0.1 + education * 0.01
walk <- rbinom(1000, size = 1, prob = (1/(1 + exp(-linear_form))))
```

Q8. Go online to find another variable (measured at the year-level) and merge it to these data. Next, use regression to model the price of gold as a function of the new variable you have added AND year (in one model). Explore the data for interactions, non-linear associations, etc. Provide an interpretation of the results that is meaningful. Note: it is not a problem if the variable you find is not associated with the price of gold.

```r
library(here)
```

```
## here() starts at /Users/horanxu/Desktop/GEOG714_Applied_Data_Analysis_for_Geographers_and_Earth_Scie
```

```r
here()
Gold_Price_training <- data.frame(read.csv(paste0(here(), "/Assignments/Assignment5/Gold price training
Gold_Price_testing <- data.frame(read.csv(paste0(here(), "/Assignments/Assignment5/Gold price testing -

names(Gold_Price_training) <- c("Year", "GoldPrice", "GoldProduction")
names(Gold_Price_testing) <- c("Year", "GoldPrice", "GoldProduction")

Gold_Price_training$Year <- as.numeric(Gold_Price_training$Year)
Gold_Price_training$GoldPrice <- as.numeric(Gold_Price_training$GoldPrice)
Gold_Price_training$GoldProduction <- as.numeric(Gold_Price_training$GoldProduction)

Gold_Price_testing$Year <- as.numeric(Gold_Price_testing$Year)
Gold_Price_testing$GoldPrice <- as.numeric(Gold_Price_testing$GoldPrice)

lmout6 <- lm(GoldPrice ~ Year + GoldProduction, data = Gold_Price_training)
summary(lmout6)
```

I found and downloaded a dataset called "Gold production, 1681 to 2015" on "ourworldindata" website. And I embedded it into the "Gold price training" dataset.

The linear regression model between `Year`, `GoldProduction` and the `GoldPrice` has shown that the former two varibles both have a statistically significant positive effect on `GoldPrice` (p -value is 0.00830 for `Year` and 0.00262 for `GoldProduction`), which are both below 0.05. The estimated coefficients suggest that for each additional year, the price of gold increases by about 2.34 units, and for each unit increase in gold production, the price increases by 0.1312 units. The model explains approximately 62.73% of the variance in gold prices (R-squared = 0.6273), indicating a rather good fit considering the number of cases.

Q9. Once you have the model completed, come up with a way to validate the model you created with this dataset:

```
Gold_Price_testing$GoldPrice_estimated <- Gold_Price_testing$Year * 2.105 + Gold_Price_testing$GoldProdu
    0.1636 + -415.6

for (i in 1:(2015 - 2010)) {
    Gold_Price_testing$Mutiple[i] <- Gold_Price_testing$GoldPrice_estimated[i]/Gold_Price_testing$GoldPr
}
```

Q10. Is the model you created any good? Why/why not?

The model is really good. Since the dataset I have for world gold production only has years up to 2015. Thus, I did a prediction using the multi-linear regression model for years in 2011-2015. Then I calculated the multiples of estimated gold prices as of real gold prices. The results ranged from 250% ~ 370%, which are inaccurate at all. This is because the production of gold has followed some kind of exponential increase while the prices follows a different increasing pattern (and also in recent years it sort of stayed stable). I think a better model should be applied, maybe a time-series model.

Q11. Use the lm() function to model walk as a function of age, income_1000s + education. Take a look at the fitted values (of the dependent variable), and comment on whether or not they make sense and why/why not.

```
lmout5 <- lm(walk ~ age + income_1000s + education)
summary(lmout5)
df5 <- data.frame(Y = walk, x1 = age, x2 = income_1000s, x3 = education)
df5$p <- lmout5$fitted.values
```

The fitted values from the `lm()` function for the `walk` variable, which is dichotomous (0 or 1), do not make sense. This is because linear regression is designed for continuous variables modelling. The fitted values in the `lmout5` are continuous and can fall outside the 0 to 1 range, which does not conform to for binary outcomes.

```
fit <- glm(walk ~ age + income_1000s + education, family = "binomial")
summary(fit)
df5$p_glm <- fit$fitted.values
```

Q12. Think about Q11, and describe why these fitted values make more sense.

The fitted values from the `glm()` model make more sense because it uses **logistic regression**, which predicts probabilities between 0 and 1 for `walk`. This aligns with the reality that choices of walking should be either "yes" (1) or "no" (0). Unlike linear regression, logistic regression appropriately handles the categorical dependent variable by predicting probabilities, not continuous values.

Q13. Do the following:

1. Randomly separate the data (all four variables) into a training data set (of 700 observations) and testing data set (300 observations)

```
set.seed(1)
train_indices <- sample(1:nrow(df5), 700)
train_data <- df5[train_indices, ]
test_data <- df5[-train_indices, ]
```

2. Use logistic regression to model the training data

```
fit_train <- glm(Y ~ x1 + x2 + x3, data = train_data, family = "binomial")
```

3. Find a way to predict the dependent variable of the test data set using the model estimated using the training data

```
test_data$predicted_prob <- predict(fit_train, newdata = test_data, type = "response")
test_data$predicted_class <- ifelse(test_data$predicted_prob > 0.5, 1, 0)
```

4. Compare the observed and predicted values, and make a judgment about the quality of the model. Justify your conclusion about the quality of the model

```
confusion_matrix <- table(test_data$Y, test_data$predicted_class)
confusion_matrix
accuracy <- sum(diag(confusion_matrix))/sum(confusion_matrix)
accuracy
```

The confusion matrix shows the comparison between the observed and predicted values in the test data. And the model's accuracy calculated (0.93) means our logistic regression moodel performs well.