Lab 8 - Linear Regression in R

ESM 206 (Fall 2018)

Complete this "take-home" lab during the week of November 19 - 23. Feel free to work together with classmates (a good opportunity to try some github collaboration?), but you are expected to know and understand lab material individually.

In lecture we've been learning about basic linear regression. In this 'take-home' lab, you will:

(a) Simplify data for easier data analysis
(b) Create an exploratory scatterplot
(c) Perform linear regression
(d) Evaluate model diagnostics
(e) Explore model fit and statistics
(f) Prepare a final graph
(g) Use the model output to make predictions
(h) Find Pearson's r (correlation)

1. Get and simplify the data

You'll be describing the increase in total U.S. fresh salmon imports from 1990 - 2017. Data source: U.S. Department of Agriculture, Economic Research Service *Aquaculture Trade Tables* (https://www.ers.usda.gov/data-products/aquaculture-data.aspx).

a. Open the Excel file 'salmon_imports.xlsx' containing the annual imports (note units).

b. Make a copy of the data, then create a simplified worksheet containing only year and volume salmon imported values that will be easier to work with in R (remember to update column names, too). If you want to be consistent with the example code shown here: use column names 'year' and 'salmon_fresh'. Save as a CSV file.

2. Create a new github repo for the lab, and get it talking to RStudio

a. Go to github and create a new repo (e.g. lab-8-yourname). Drop the simplified CSV into the repo and commit. Clone the repo to get it talking with RStudio.

b. In the R project that you just created by cloning the repo, open a new Rmarkdown document. Title the markdown document 'Lab 8 - Linear Regression in R'. Remove all text below the first code chunk. Save.

c. Stage, Commit, and Push everything (including the gitignore file) back to github. Ensure that when you refresh the repo in github, the updated files appear.

d. Go back to RStudio to complete the rest of the lab. Practice stage/commit/push several times throughout the process, ensuring that the versions are stored in github when you do.

3. Load the tidyverse, read in data

In the markdown document, add a new code chunk:

- Load the tidyverse
- Read in the data (I read it in as 'salmon')

4. Exploratory scatterplot

If we're going to use linear regression, we should have reason to think that a linear relationship exists between variables. So go ahead and look at the data (year on x-axis, imports on y-axis).

a. Make an exploratory scatterplot in ggplot (recall: geom_point). Do not finalize.

b. As a comment in the code chunk, answer: Overall, does it look like describing the relationship between year and fresh salmon imports using linear regression makes sense? Do you have any concerns about using linear regression to describe the relationship?

5. Linear regression in R

The general syntax for linear regression in R is:

```
model_name <- lm(y_variable ~ x_variable, data = df_name)
```

So, if I have a data frame called 'salmon' containing variables 'year' and 'salmon_fresh,' I would use:

```
salmon_model <- lm(salmon_fresh ~ year, data = salmon)
```

a. Perform linear regression to describe the trend in total volume of fresh salmon imported to the U.S. from 1990 - 2017.

b. Call the model name to see the intercept and slope for the model. Recall:

$y = \beta_0 + \beta_1 x + \epsilon$

c. Write out the model equation, including the actual variables and coefficients instead of β and x/y. For example (these are not the actual model coefficients you'll find):

$Imports(tons) = -400 + 25(Year)$

d. Think about the model equation that you found to describe trends in salmon imports. In your markdown document, answer the following in 1-sentence each:

- What does the *slope* mean in the context of this model?
- What does the *y-intercept* mean in the context of this model? Why is that concerning? What does this mean about *extrapolating* this model for past values?

6. Model diagnostics

Use plot(model_name) to view model diagnostics in the 'Plots' tab (press Enter in the Console to contine to subsequent diagnostic plots). Explore the diagnostic plots to answer the following:

- Do residuals appear normally distributed?
- Any concerns about heteroscedasticity or outliers?

To view all four diagnostic plots at once (and have them appear in your knitted document), you can use the following:

```
par(mfrow = c(2,2))
plot(model_name)
```

7. Explore model fit and significance

Use summary(model_name) to see the detailed model information, including model fit information (e.g. R^2 and coefficient standard errors) and statistics.

See the lecture materials to interpret each component. Answer the following:

- Does year significantly predict salmon imports?
- What does the R^2 value actually mean in words?
- What proportion of the variance in salmon imports is NOT explained by year?

8. Prepare a final graph

See the example code below to create a final graph of the linear regression model with the original data, model and 95% confidence interval for predictions.

The code and graph shown below are created with:

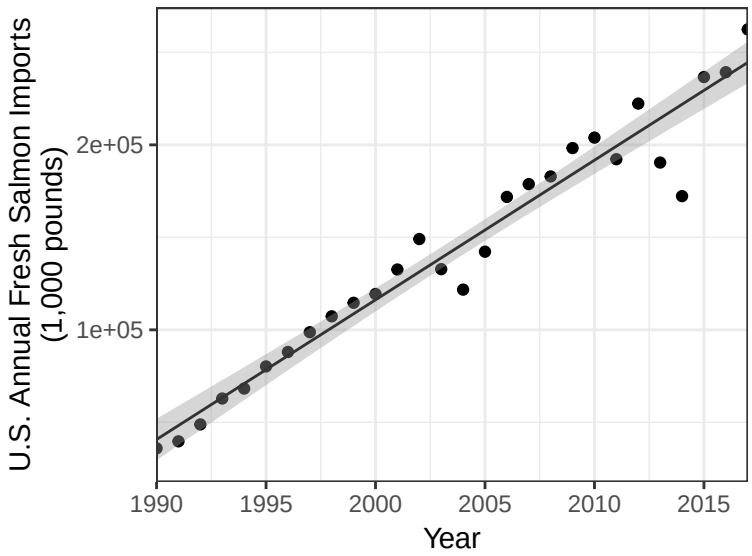
- An original data frame called 'salmon'
- ... which contains x and y variables 'year' and 'salmon_fresh'
- ... for which a linear model was found, called 'salmon_model'

If your data frame, columns, or model name differ, you'll need to update accordingly.

You should know what each component and argument of this graphics code actually does. Explore (i.e. Google) geom_smooth() to better understand what the plotted CI ribbon actually means, and to see other alternatives within the geom.

```
salmon_final_graph <- ggplot(salmon, aes(x = year, y = salmon_fresh)) +
  geom_point() +
  geom_smooth(method = lm, se = TRUE, size = 0.5, color = "gray20") +
  theme_bw() +
  scale_x_continuous(expand = c(0,0), limits = c(1990, 2017)) +
  labs(x = "Year", y = "U.S. Annual Fresh Salmon Imports\n(1,000 pounds)")

salmon_final_graph
```



9. Make predictions using the model

Let's say we want to use our model to make predictions for U.S. fresh salmon imports in years 2022, 2024, 2026 and 2028.

- Option 1 (not recommended): Plug each of those years into the model equation separately to calculate the expected U.S. imports.
- Option 2 (recommended): Create a data frame containing the 'new values' that you want to make predictions for, feed that into the predict() function

Let's use Option 2. First, we'll make a data frame called 'new_years' containing the sequence of years that we want to make predictions for. Notice that the column name we create is the **SAME** as the variable name 'year' that the model uses.

```
new_years <- data.frame(year = c(2022, 2024, 2026, 2028))
```

View the new data frame you just created (it's a single column, with heading 'year', containing those 4 values).

```
new_years
```

```
##   year
## 1 2022
## 2 2024
## 3 2026
## 4 2028
```

Then feed that data frame into the predict() function, letting it know which model to use to make predictions for the new values.

Include interval = "confidence" to also report the lower and upper bounds of the 95% CI for model fit at each year.

```
future_predict <- predict(salmon_model, newdata = new_years, interval = "confidence")
future_predict
```

```
##        fit      lwr      upr
## 1 282298.5 267877.4 296719.6
## 2 297397.6 281656.7 313138.5
## 3 312496.8 295418.5 329575.0
## 4 327595.9 309166.6 346025.2
```

Bind the prediction outcomes (those are the values in column 'fit') with the 'new_years' data to actually create a useful table of predicted imports and upper and lower CI:

```
predictions <- data.frame(new_years, future_predict)
predictions
```

```
##   year      fit      lwr      upr
## 1 2022 282298.5 267877.4 296719.6
## 2 2024 297397.6 281656.7 313138.5
## 3 2026 312496.8 295418.5 329575.0
## 4 2028 327595.9 309166.6 346025.2
```

10. Find Pearson's r (correlation)

Use the cor.test() function to find Pearson's r for the linear relationship described between year and salmon imports.

See ?cor.test to view R documentation in the 'Help' window. Notice that the default correlation is Pearson's r.

Generally, if you're trying to explore the correlation between variable_1 and variable_2 in data frame 'df', the function syntax is as follows:

```
test_name <- cor.test(df$variable_1, df$variable_2)
```

a. Calculate Pearson's r for the year vs. salmon imports linear trend.

b. In words: Would you describe this as a weak/strong negative/positive correlation?

11. Write a concluding statement about the relationship

Using the document *Communicating Results of Basic Linear Regression* (posted on GauchoSpace) as a guide, write a final 1 - 2 sentence statement describing the results of your linear regression and Pearson's r findings.

That's it!

SAVE YOUR PROJECT %>%

STAGE/COMMIT/PUSH TO GITHUB (if using version control) %>%

ENJOY YOUR WEEK %>%

SEE YOU NEXT MONDAY