Lab Session 5: Tying it all together!

Izzie Fulcher

3/30/2021

Set working directory

We will first set a working directory. Your working directory should be where your plan to save your R code and also where the example datasets have been stored. In the "Files" tab, navigate to this folder. Once you are there, select the "More" dropdown and select "Set As Working Directory".

Install and load R packages

You should have already installed the tidyverse package. Now, you will need to load the package into R. This will allow you to use the functionality of the package.

```r
library(tidyverse)
library(lubridate)
```

Load the monthly number of acute respiratory infections ".rds" file and save it as a data frame called "facility".

```r
facility <- readRDS("session5_data/facility_ari_final.rds")
```

View the first six observations in the facility dataset.

```r
head(facility)
```

PART 1. Data cleaning

What is mean, median, min, max acute respiratory infection count in the dataset?

```r
facility %>%
  summarize(mean(date),
            median(date),
            min(date),
            max(date))
```

Create a scatterplot of the date and actue respiratory infection count. Is there evidence of seasonality or trend?

```r
ggplot(facility,aes(x=date,y=ari_count)) +
  geom_point() +
  geom_line() +
  theme_bw()
```

Are there any potential outliers? If so, what do you think is causing this outlier and what should be done? (**We will fill in this value with 1004**)

```r
facility %>%
  mutate(ari_count = case_when(ari_count == 0 ~ 1004,
                               TRUE ~ ari_count)) -> facility_new
```

PART 2. Fit time series model to baseline data

Create a new data frame for that includes any additional variables you would like to include in your model (month terms, year terms, cosine, sine, etc.). Call this "facility_all".

```r
facility_new %>%
  arrange(date) %>%
  mutate(month=1:n()) -> facility_all
```

The scenario is the same as our syndromic surveillance for COVID-19. We typically choose a baseline period up to Janaury 1, 2020, but you can feel free to choose your own date range. Filter this dataset to create a baseline dataframe. Call this "facility_baseline"

```r
facility_all %>%
  filter(date < as.Date("2020-01-01")) -> facility_baseline
```

Using the code from previous labs (Session 2, Session 3) and looking at your scatterplot, fit a baseline model.

```r
model.fit <- lm(ari_count ~ month, data=facility_baseline)
```

Plot the fitted (predicted) values and prediction interval with the observed baseline data. How does it look?

```r
# Create data frame for predictions
pred_baseline <- data.frame(predict(object = model.fit,
                                    newdata = facility_baseline,
                                    interval = "predict")) %>%
  # need to include observed counts and time (date)
  mutate(observed=facility_baseline$ari_count,
         date = facility_baseline$date)

# Generate plot
ggplot(pred_baseline,aes(x=date,y=observed)) +
  geom_point() +
  geom_ribbon(aes(ymin=lwr,ymax=upr),fill = "grey70",alpha=.4) +
  geom_line(aes(x=date,y=fit)) +
  geom_vline(xintercept=as.Date("2020-12-01")) +
  theme_bw()
```

Check the residuals by creating a residual plot or autocorrelation function plots. How does it look?

```r
pred_baseline %>%
  mutate(residuals = observed-fit) %>%
  ggplot(.,aes(x=date,y=residuals)) +
  geom_point() +
  theme_bw() +
  geom_hline(yintercept=0)
```

```r
acf(model.fit$residuals, lag.max=48)
```

Repeat the above steps until you are satisfied with your model choice. (Independent activity)

PART 3. Calculate deviations in the evaluation period

You will first need to calculate the predicted (fitted) values and prediction intervals in the evaluation period.

```r
# Create data frame for predictions
pred_all <- data.frame(predict(object = model.fit,
                               newdata = facility_all,
                               interval = "predict")) %>%
  # need to include observed counts and time (date)
  mutate(observed=facility_all$ari_count,
         date = facility_all$date)
```

Calculate deviation in two ways: (1) the difference between observed and predicted and (2) the proportion difference between observed and predicted. Are the counts higher or lower than expected by month? Do they exceed the prediction intervals?

```r
pred_all %>%
  mutate(deviation = round(observed-fit,2),
         deviation_scaled = round((observed-fit)/fit,2),
         outside_pi = case_when(observed < lwr | observed > upr ~ "Deviation",
                                TRUE ~ "No deviation")) %>%
  filter(date >= "2020-01-01") %>%
  dplyr::select(date,deviation,deviation_scaled,outside_pi)
```

Create the classic time series plot with an observed line, predicted line, and prediction intervals for the entire time period. You should also include a line at the evaluation period.

```r
ggplot(pred_all,aes(x=date,y=observed)) +
  geom_point() +
  geom_ribbon(aes(ymin=lwr,ymax=upr),fill = "grey70",alpha=.4) +
  geom_line(aes(x=date,y=fit)) +
  geom_vline(xintercept=as.Date("2020-12-01")) +
  theme_bw()
```

PART 4. Communicate results (data visualization!)

Now, modify the plot to best communicate the results to a clinician. Feel free to get creative with plot style, colors, labeling, etc.!