Project 3: Forecasting

TOTAL: 25pts

GOAL

Forecast future crime for specific types and shifts in Washington, D.C. by month, for the remainder of calendar year 2025 (April-December).

These instructions will demonstrate how to forecast using an ***ARIMA model***. An ARIMA model is an Autoregressive Integrated Moving Average, which is a univariate model predicting future values of a single variable over time. Let's breakdown these terms:

- **Autoregressive**: "auto" means self, and "regressive" refers to using prior values.
- **Integrated**: "integrated" refers to the process of making data stationary, which identifies trends by differencing the data
- **Moving Average**: "moving average" is a calculation for a series of averages using subsets of data to identify trends over time. It is a more precise measure for smoothing out fluctuations.

Step 0: PREP

Install and load packages:

- install.packages('tidyverse')
- install.packages('data.table')
- install.packages('leaflet')
- install.packages('sf')
- install.packages('tigris')
- install.packages('zoo')
- install.packages('tseries')
- install.packages('aTSA')
- install.packages('forecast')
- library(tidyverse)
- library(leaflet)
- library(sf)
- library(tigris)
- library(zoo)
- library(aTSA)

- library(tseries)
- library(forecast)
- library(data.table)

Step 1: ACQUIRE

Get the Washington, D.C. crime data, clean, merge, and enrich:

- See Project 2 (Steps 1-3) to get all the crime data, merge it into one big table, and enrich with year, month, day, dow, and hour

Step 2: FILTER

Identify unique values for fields of interest:

- unique(dc.data$OFFENSE)
- unique(dc.data$SHIFT)

Create summary tables for counts of activity:

- summary.offense <- dc.data %>%
- group_by(OFFENSE) %>%
- summarise(COUNT = n()) %>%
- mutate(PCT = round(COUNT/sum(COUNT)*100,2))
- Repeat for SHIFT, and combinations of the two

Identify subsets of data to analyze:

- dc1 <- subset(dc.data, dc.data$OFFENSE == 'xxx' & dc.data$SHIFT == 'yyy')
- Repeat for other crimes and shifts

The purpose of this step is to identify which crime types during which shifts are optimal to forecast. Don't choose non-existent or extremely low counts.

Step 3: MODEL

To use an ARIMA model, we will identify values for **p**, **d**, and **q**:

- **p**: Lagged observations
- **d**: Differencing the data set

- **q**: Prior error handling

Calculate crimes per shift, per day:

```
crime <- dc1 %>%
  group_by(DATE) %>%
  summarise(COUNT = n())
```

Fill in the blank days:

```
crime <- crime %>%
  complete(DATE = seq.Date(min(DATE), max(DATE), by = "day")) %>%
  mutate(WEEKDAY = lubridate::wday(DATE, label = T, week_start = 1),
         MONTH = lubridate::month(DATE, label = T, abbr = F),
         WEEK = isoweek(DATE),
         DAY = day(DATE),
         YEAR = year(DATE))
# replace the NAs with 0s
crime <- replace(crime, is.na(crime), 0)
```

Change the data type to a time series, update the sequencing:

```
cleaned.data <- zoo(crime$COUNT,
                    seq(from = as.Date(min(crime$DATE)),
                        to = as.Date(max(crime$DATE)-1), by = 1))
```

Generate basic summary statistics, and a basic graph:

```
summary(cleaned.data)
plot(cleaned.data)
title("Insert A Title Here") # this adds a title to your graph
```

Make the data stationary (normalized, as deviations from previous values) and graph it:

```
stationary1 <- diff(cleaned.data, differences = 1)
plot(stationary1)
stationary2 <- diff(cleaned.data, differences = 2)
plot(stationary2)
stationary3 <- diff(cleaned.data, differences = 3)
plot(stationary3)
```

Look for the data with the narrowest y axis range, as consistently as close to 0 as possible.

Conduct an Augmented Dickey-Fuller Test for data stationariness:

- adf.test(as.matrix(cleaned.data))
- adf.test(as.matrix(stationary1))
- adf.test(as.matrix(stationary2))
- adf.test(as.matrix(stationary3))

The resultant score should be a negative number, and the lower the number the stronger the data is stationary. In this example, 'cleaned.calls' is run as a comparison to data that is not differenced. Ideally you want to manipulate the data as little as possible.

The combination of the stationary plot analysis and the ADF scores determines the d value.

Lagged autocorrelations:

Use the Autocorrelation (ACF) and Partial Autocorrelation (PACF) functions to determine the **p** and **q** values. For both calculations and visuals, you will use the best stationary dataset you chose in steps above.

Determine the **p** order by examining the PACF graph and data. Specifically, look for the last *significant* lag:

- pacf(stationaryxxx)
- pacf(stationaryxxx, pl=FALSE)

Determine the **q** order by identifying the last *significant* lag in the ACF:

- acf(stationaryxxx)
- acf(stationaryxxx, pl=FALSE)

Create an ARIMA function:

Build an ARIMA function using the inputs derived from previous steps. Also, based on analysis from the crimes per day graph, it's likely reasonable to assume your data has season trends - so we'll set it to TRUE):

- arima.data <- auto.arima(cleaned.data, d = xxx, max.p = xxx, max.q = xxx, seasonal = T)
- summary(arima.data)

Note that the 'auto.arima' function runs a series of ARIMA models and chooses the best fit for the data. The best fitting model can be interpreted as ARIMA(d, p, q).

Check the model residuals:

- checkresiduals(arima.data)

Proper residuals derived from your model include:

- the top graph not demonstrating any clear, cyclical, repeating patterns
- the bottom right graph generally being a normally distributed bell curve

Sample forecast for next week:

Using the function just created, model next week's crime counts:

- # h = the number of units of time to measure
- forecast.7days <- forecast(arima.data, h=7)
- round(sum(forecast.7days$upper[,2]),0)
- forecast.7days$mean

HERE IS THE SCRIPT WE MADE IN CLASS ON 19 FEBRUARY 2025

Build a table of forecasts for April-December 2025:

Identify the number of days between the last date in your dataset and the end of 2025:

- forecast.window <- as.numeric(as.Date("2025-12-31")-max(crime$DATE))

Forecast the number of crimes per day for the rest of 2025:

- forecast.2025 <- forecast(arima.data, h=forecast.window)

Plot the forecast:

- autoplot(forecast.2025)

Extract the forecasted values as a table, clean up the column names, add the forecast date, and month:

- forecast.values <- as.data.frame(forecast.2025$mean)
- forecast.values$ID <- seq.int(nrow(forecast.values))
- forecast.upper <- as.data.frame(forecast.2025$upper)
- forecast.upper$ID <- seq.int(nrow(forecast.upper))
- forecast.values <- forecast.values %>%
- left_join(forecast.upper, by = 'ID')
- colnames(forecast.values) <- c("MEAN", "ID", "CI80", "CI95")

- forecast.values$DATE <- as.Date(max(crime$DATE) + forecast.values$ID)
- forecast.values$MONTH <- months(forecast.values$DATE)

Filter to April-December, summarize forecasts by month:

- forecast.values.2025 <- subset(forecast.values, forecast.values$DATE > '2025-03-31')
- forecast.months <- forecast.values.2025 %>%
- group_by(MONTH) %>%
- summarise(MEAN = round(sum(MEAN),0), FORECAST.95 = round(sum(CI95),0), FORECAST.80 = round(sum(CI80),0))
- forecast.months$DIFF <- forecast.months$FORECAST.95 - forecast.months$FORECAST.80

Step 4: VISUALIZE

Graph crimes per day with a trend line:

- graph.crime <- ggplot(crime, aes(x=DATE, y=COUNT)) +
- geom_line() +
- scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
- xlab("Years") +
- ylab("Crime Count") +
- ggtitle("TITLE HERE") +
- geom_area(fill="lightblue", color="black")
-
- graph.crime +
- geom_smooth(method = lm, col = "red", se = FALSE) +
- theme(axis.text.x = element_text(angle = 90, hjust = 1))

Graph crimes per day AND forecasted crime:

- crime2024 <- crime[c(1,2)]
- crime2025 <- forecast.values[c(5,1)]
- names(crime2025) <- c("DATE", "COUNT")
- new.crime <- rbind(crime2024, crime2025)
-
- graph.new.crime <- ggplot(new.crime, aes(x=DATE, y=COUNT)) +
- geom_line() +
- scale_x_date(date_breaks = "1 year", date_labels = "%Y") +
- xlab("Years") +
- ylab("Crime Count") +
- ggtitle("TITLE HERE") +
- geom_area(fill="lightblue", color="black")
-
- graph.new.crime +

```
  geom_smooth(method = lm, col = "red", se = FALSE) +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

Graph monthly forecasts, by upper bounds (95% CI):

```
forecast.months$MONTH <- factor(forecast.months$MONTH, levels = forecast.months$MONTH)
forecast.long <- pivot_longer(forecast.months, cols = c(FORECAST.80, DIFF),
                              names_to = "Category", values_to = "Value")
forecast.long$Category <- gsub("DIFF", "95% Confidence Interval", forecast.long$Category)
forecast.long$Category <- gsub("FORECAST.80", "80% Confidence Interval",
forecast.long$Category)

ggplot(forecast.long, aes(x = MONTH, y = Value, fill = fct_rev(Category))) +
  geom_bar(stat = "identity") +
  geom_text(aes(label = Value), size = 3, colour = 'white', position = position_stack(vjust = 0.5)) +
  labs(title = "Washington D.C. 2025 Monthly Forecast Upper Bounds",
       x = "Month",
       y = "Crime Count") +
  scale_fill_manual(values = c("95% Confidence Interval" = "blue",
                               "80% Confidence Interval" = "grey"),
                    name = "Forecasts") +
  scale_x_discrete(limits = c("January", "February", "March", "April", "May", "June", "July",
"August",
                              "September", "October", "November", "December")) +
  coord_cartesian(ylim = c(0, 550)) +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1))
```

Graph monthly forecasts, by mean:

```
ggplot(forecast.months, aes(x = MONTH, y = MEAN)) +
  geom_bar(stat = "identity") +
  scale_x_discrete(limits = c("January", "February", "March", "April", "May", "June", "July",
"August",
                              "September", "October", "November", "December")) +
  coord_cartesian(ylim = c(150, 200)) +
  theme_classic() +
  theme(axis.text.x = element_text(angle = 90, hjust = 1)) +
  labs(title = "Washington D.C. 2025 Mean Monthly Forecast",
       x = "Month",
       y = "Crime Count")
```

Maps:

- See Project 2 for details on making ***hotspot facet maps***

```
dc.roads <- roads("DC", "District of Columbia")
dc.roads.major <- dc.roads %>%  filter(RTTYP %in% c("I","S","U"))
dc.outline <- county_subdivisions("DC", "District of Columbia")
dc.placenames <- landmarks("DC")
dc.water <- area_water("DC", "District of Columbia")
dc.water <- filter(dc.water, AWATER >= 1000)

ggplot() +
  geom_sf(data = dc.outline$geometry, fill = "transparent",
          linewidth = 1, color = "black") +
  geom_sf(data = dc.roads$geometry, alpha = 0.1) +
  geom_sf(data = dc.water$geometry, fill = "lightblue") +
  geom_hex(aes(x = LONGITUDE, y = LATITUDE), data = dc1, bins = 8,
           alpha = 0.5) +
  scale_fill_continuous(type = "viridis") +
  theme(plot.title = element_text(hjust = 0.5, size = 14),
        axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank(),
        panel.grid.major = element_blank(),
        panel.background = element_blank()) +
  facet_wrap(~YEAR)
```

Temporal Topology:

A temporal topology visualizes two variables in your data, allowing for accurate comparisons of both at the same time. Most notably, hour of day and day of week.

- Create a summary table

```
crime.day.time <- dc1 %>%
  group_by(DOW, HOUR) %>%
  summarise(COUNT = n())
crime.day.time <- replace(crime.day.time, is.na(crime.day.time), 0)
```

- Basic graph

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile()
```

- Fixed coordinates

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile() + coord_fixed()
```

- Formatted with spaces

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile(color = "white", lwd = 1.5, linetype = 1) +
  coord_fixed()
```

- New colors, and numbers in cells

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile(color = "black", lwd = .5, linetype = 1) +
  geom_text(aes(label = COUNT), color = "white", size = 1.5) +
  scale_fill_gradient(low = "blue", high = "red") +
  coord_fixed()
```

- New number formats

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile(color = "black", lwd = .5, linetype = 1) +
  geom_text(aes(label = COUNT), color = "black", size = 2.5) +
  scale_fill_gradient(low = "blue", high = "red") +
  guides(fill = guide_colourbar(title = "Crime Count")) +
  coord_fixed()
```

- New color ramp, better legend

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile(color = "black", lwd = .5, linetype = 1) +
  geom_text(aes(label = COUNT), color = "black", size = 1.5) +
  scale_fill_gradient(low = "white", high = "red") +
  guides(fill = guide_colourbar(title = "Crime Count", label = FALSE, ticks = FALSE)) +
  coord_fixed()
```

- No legend

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile(color = "black", lwd = .5, linetype = 1) +
  geom_text(aes(label = COUNT), color = "black", size = 2.5) +
  scale_fill_gradient(low = "white", high = "red") +
  theme(legend.position = "none") +
  coord_fixed()
```

- Reorganize the days

```
ggplot(crime.day.time, aes(HOUR, DOW, fill = COUNT)) +
  geom_tile(color = "black", lwd = .5, linetype = 1) +
  geom_text(aes(label = COUNT), color = "black", size = 2.5) +
  scale_fill_gradient(low = "white", high = "red") +
  theme(legend.position = "none") +
  coord_fixed() +
  scale_y_discrete(limits =
c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday"))
```

- To make graphs per month, do a new summary table

```
crime.year.day.time <- dc1 %>%
  group_by(YEAR, MONTH, DOW, HOUR) %>%
  summarise(COUNT = n())
crime.year.day.time <- replace(crime.year.day.time, is.na(crime.year.day.time), 0)

ggplot(crime.year.day.time, aes(HOUR, DOW, fill = COUNT)) +
```

```
    geom_tile() +
    scale_fill_gradient(low = "white", high = "red") +
    theme(legend.position = "none") +
    coord_fixed() +
    scale_y_discrete(limits =
  c("Sunday","Monday","Tuesday","Wednesday","Thursday","Friday","Saturday")) +
    theme(axis.text.x = element_text(angle = 90, size = 5)) +
    theme(axis.text.y = element_text(size = 5)) +
    facet_wrap(~ YEAR, nrow = 6)
```

HERE IS THE SCRIPT WE BUILT IN CLASS ON 26 FEBRUARY 2025

Step 5: Customized forecasts and visualizations

- Repeat this process above for each CRIME-SHIFT combination
- The result should be THREE sets of CRIME-SHIFT forecasts, graphs, and maps
    - **Forecasts** should be counts of expected crime for each month, April-December 2025
    - **Graphs** should be *no more than two*
        - One graph should highlight your model performance for April-December 2025
        - One graph should be a temporal topology, faceted by month
    - **Maps** should be hotspots faceted by month
- For graphs and maps, consider filtering your data to remove activity before COVID.
    - Because changes to crime activity before and after COVID is absolutely a thing.
        - To determine if this applies to your data, examine the first crimes per day graph you created on page 3. If you notice a dramatic shift (probably decrease) in events in early 2020, subset your data before creating your visuals.

Step 6: Analysis

Use the monthly forecast counts, hotspot maps, and temporal topology to make "sense" of the upcoming month. Consider these questions:

- Does the monthly forecast change much? When, and by how much?
- What days and times does activity occur? What days and times does activity *not* occur? Does this change month to month?
- Where are the hotspots?
- Where are the coldspots?
- Do the hotspots and coldspots change month to month?

Answers to questions like these will provide context to your script. Make sure the analysis is consistent with the visuals. Write clearly and consistently, using plain English statements.

As you write, remember that words have meanings. Some are stronger than others. Others aren't real.

Further, here's some helpful writing strategies:

- Read sentences aloud… but don't write like you speak.

- Ask yourself - does this sentence make sense, by itself.
- If you don't know the meaning of a word, don't use it!

And if you're bored, here you go.

Part II: SUBMISSION

Transform your script as a usable HTML file, via RMarkdown. You will want to copy/paste the most relevant parts of the script you've built into a clean, consolidated file. You create this output as a .Rmd file in RStudio, but what you submit will be an .HTML file.

Use this file as a template

1. Adjust the template with your code accordingly
2. Use the 'knit' button in RStudio to produce the output
    a. The .HTML is automatically saved to the same folder as your script
3. Click 'Open in Browser' to display in your web browser
4. Save it as a PDF.

HERE IS THE MARKDOWN FILE WE BUILT IN CLASS ON 26 FEBRUARY 2025

HERE IS THE MARKDOWN FILE WE BUILT IN CLASS ON 05 MARCH 2025

GRADES

- **Part I: Model Deployment (24pts)**
    - Build a forecast for one specific crime type during ***day*** shift, including
        - One bar graph of the number of expected crimes to occur each month, April-December 2025 (**2pts**)
        - One temporal topology of relevant crimes by day of week and hour of day, faceted by month (**1pts**)
        - One hotspot map of relevant event data, including the D.C. outline and streets, faceted by year (**1pts**)
        - One paragraph (6-8 sentences) explaining the findings of your forecast, including where and when you expect this crime to occur in D.C. during 2025 (**4pts**)
    - Build a forecast for a different specific crime type during ***evening*** shift, including
        - One bar graph of the number of expected crimes to occur each month, April-December 2025 (**2pts**)
        - One temporal topology of relevant crimes by day of week and hour of day, faceted by month (**1pts**)
        - One hotspot map of relevant event data, including the D.C. outline and streets, faceted by year (**1pts**)
        - One paragraph (6-8 sentences) explaining the findings of your forecast, including where and when you expect this crime to occur in D.C. during 2025 (**4pts**)
    - Build a forecast for a third unique crime type during ***midnight*** shift, including

    - One bar graph of the number of expected crimes to occur each month, April-December 2025 (**2pts**)
    - One temporal topology of relevant crimes by day of week and hour of day, faceted by month (**1pts**)
    - One hotspot map of relevant event data, including the D.C. outline and streets, faceted by year (**1pts**)
    - One paragraph (6-8 sentences) explaining the findings of your forecast, including where and when you expect this crime to occur in D.C. during 2025 (**4pts**)
- **Part II: Submission (1pt)**
    - Project created in RMarkdown and submitted as an HTML or PDF file via Canvas
    - Each forecast should be in its own section, clearly identifying what crime offense types are being visualized

Please email me with any questions.