German Uboat Attacks in WWII data project

adenoz

2023-04-21

Introduction

This document aims to provide an example of data analysis that is conducted against an *event* dataset. This event dataset contains curated and carefully structured data on German Uboat attacks during WWII.

This analysis was conducted using the R programming language, in an R Markdown notebook using the RStudio Integrated Development Environment (IDE). While this analysis was conducted using R, it could just as easily be conducted using Python or Julia in Jupyter Notebooks.

Data science notebooks allow analysts to write prose while also include *code blocks* that conduct some action against the imported data. Typically, this can include data ingestion, wrangling, tidying, transformations, plotting and graphing as well as modelling.

The PDF you are reading is an example of what a notebook can be exported as. The actual code blocks can all be included in the output, a selection can be included, or none can be included. This will depend on the audience. If you are sharing an analysis with other technical users, you will probably want to include the code so they can see what has been done. For a non-technical reader, no code can be included.

In this example, all code will be included as the intent of this is to expose analysts to what code is used to generate the various outputs. However, just be aware that if the purpose was to communicate knowledge about data, the 'so what', none of the actual code would be needed and would actually detract. Note that in the code blocks, lines beginning with # are known as comments. This is typically not actually code, but provides some explanations of what is happening to the code. The program does not actually read any comments as it skips any line beginning with #. The comments are there purely for human readers. However some lines of code can also be "commented out" so they are not run, but they are easy to use again by deleting the #.

This document shows the progress and thinking and exploring as we get deeper and deeper into a dataset. This sort of analysis is known as *exploratory data analysis*. In this form, it may not all be required to be in a final report. Typically, you would go through this exploratory phase, find out interesting things, then produce a seperate more focused more concise report (or dashboard) that doesn't take a reader on a journey but just shows them the so what. However this example is aimed at analysts for educational purposes, so it does show the exploratory journey.

The following code block contains instructions for setting up and importing the required packages used in this analysis. It also sets the working directory.

```
knitr::opts_chunk$set(echo = TRUE, warning = FALSE, message = FALSE)
#install.packages("dplyr")
#install.packages("ggplot2")
#install.packages("lubridate")
#install.packages("faraway")
#install.packages("tidyr")
#install.packages("MASS")
```

```
#library(faraway)
library(MASS)
library(lubridate)
library(tidyr)
library(knitr)
library(dplyr)
library(ggplot2)
# Set working directory
#workd <- "path/to/directory"
#setwd(workd)
```

The data

The dataset contains details of all attacks (events) conducted by German Uboats during WWII. This dataset contains data collected from uboats.net. The uboat data was downloaded from this Github repository.

The uboat data is an interesting dataset. It contains records of over a thousand German u-boat submarines regarding their activities during WWII. It contains not only data about the boats themselves but about their operational activities including the number of ships they sunk, tonnage sunk and number of human fatalities from their attacks. It also includes details of their commanders and the time periods they were active.

This dataset offers numerous opportunities to answer some interesting questions that would have been interesting to be able to seek answers to during the actual conflict to inform planning and operations.

There are two CSV files for this dataset. The following files were downloaded from the repository:

- uboat-data.csv
- uboat-target-data.csv

```
uboat.df = read.csv("uboat_data.csv")
uboat.target.df = read.csv("uboat_target_data.csv")
# View(uboat.df)  # View opens the data in a new tab to visually explore.
# View(uboat.target.df)  # This can be handy while you are getting used to a new dataset
# head(uboat.df)
# head(uboat.target.df, 20)
```

Uboat dataframe

The uboats data has 1153 rows, or observations, and 15 columns, or variables. The dataframe contains basic records of each uboat from the date they were commissioned through to their fate. This dataframe also includes a metric of the total ships each uboat sunk. There is one line per uboat. Note that some uboats were not included. For example, there are no records for uboats 112 to 115. Further research could explore why this may be or if there even were uboats with those designations or not and why that may be etc. Gaps in data can be important to know and understand the implications of.

Here are the names of the variables.

```
names(uboat.df)
```

```
##  [1] "id"             "name"           "type"           "ordered"
##  [5] "laid_down"      "commissioned"   "launched"       "fate"
##  [9] "fate_type"      "fate_survivors" "fate_lat"       "fat_lon"
## [13] "fate_dead"      "shipyard"       "ships_sunk"
```

Uboat target dataframe

The uboat target data has 3458 observations and 14 variables. This dataframe contains a single observation for each attack from a uboat. It includes details of each attack including the uboat, date, time and location as well as numerous details of the target including type, nationality, tonnage, complement of personnel, survivors, killed and the uboat commander for each attack.

Here are the names of the variables.

```
names(uboat.target.df)
```

```
##  [1] "id"          "name"        "loss_type"   "attack_time" "attack_date"
##  [6] "attack_lat"  "attack_lo"   "nationality" "tonnage"     "complement"
## [11] "survivors"   "dead"        "commander"   "ship_name"
```

Research questions

By analysing this data, we seek to gain an insight into German uboat operations and answer the following questions:

- Who was the highest threat uboat commander?
- Who was the least effective uboat commander?
- What were the preferred targets of the highest threat uboat commanders?
- When were attacks most successful?
- How did uboat operations change over time?
- Where were uboats most active or lethal? (don't know if I'll get to this one)

I haven't focused on it in this analysis, but there are coordinates for both dataframes which could facilitate interesting geo-spatial analysis.

Data wrangling

Data cleaning

Before diving into the data and our analysis, we need to firstly do some data cleaning. The year variables are entered inconsistently (sometimes the year is written in full like 1941 and sometimes abbreviated like 41). The time variable is entered as a character string so is also inconsistent. We'll need to change some data types to the proper type and this will enable us to sort and filter as desired.

```
# DATES=======================

# NOTE: I could almost do this all in one go doing all four date columns at once.
# HOWEVER: I could not work it out when using the ifelse statement to fix the year formats
# SO: I am needing to do this manually for each date columns which isn't nice

# ordered
# Note the ifelse statement. This is needed as years in data are inconsistent using
# short (42) and long (1942)
# so need the ifelse to fix both formats.
# start with the long format date change. Then the short years will look like year 0042.
```

```r
# so the if < "1900-01-01" will pick up years like "0042" and change the year to "19%y".
# note also below code works when day or month have zero in front or not like 7 or 07.
uboat.df$ordered = as.Date(uboat.df$ordered, format = "%m/%d/%Y")
uboat.df$ordered = as.Date(ifelse(uboat.df$ordered < "1900-01-01",
                      format(uboat.df$ordered, "19%y-%m-%d"), format(uboat.df$ordered)))

# laid_down
uboat.df$laid_down = as.Date(uboat.df$laid_down, format = "%m/%d/%Y")
uboat.df$laid_down = as.Date(ifelse(uboat.df$laid_down < "1900-01-01",
                      format(uboat.df$laid_down, "19%y-%m-%d"), format(uboat.df$laid_down)))

# commissioned
uboat.df$commissioned = as.Date(uboat.df$commissioned, format = "%m/%d/%Y")
uboat.df$commissioned = as.Date(ifelse(uboat.df$commissioned < "1900-01-01",
                      format(uboat.df$commissioned, "19%y-%m-%d"), format(uboat.df$commissioned)))

# launched
uboat.df$launched = as.Date(uboat.df$launched, format = "%m/%d/%Y")
uboat.df$launched = as.Date(ifelse(uboat.df$launched < "1900-01-01",
                      format(uboat.df$launched, "19%y-%m-%d"), format(uboat.df$launched)))

# fate
uboat.df$fate = as.Date(uboat.df$fate, format = "%m/%d/%Y")
uboat.df$fate = as.Date(ifelse(uboat.df$fate < "1900-01-01",
                      format(uboat.df$fate, "19%y-%m-%d"), format(uboat.df$fate)))

# also for uboat.target.df for attack_date variable
uboat.target.df$attack_date = as.Date(uboat.target.df$attack_date, format = "%m/%d/%Y")
uboat.target.df$attack_date = as.Date(ifelse(uboat.target.df$attack_date < "1900-01-01",
                      format(uboat.target.df$attack_date, "19%y-%m-%d"),
                      format(uboat.target.df$attack_date)))

#head(uboat.target.df)

# TIMES=========================

# Note that time is linked to a date. so we need to link the attack_time variable to the
# attack_date variable
# so we'll combine those columns
# BUT. Even before that, note we have notation for AM and PM. parsing dates won't recognise that.
# SO, we need to separate that data out into a separate column so we can fix the dates later.
# ALSO, some of the observations do not have entries for attack_time so these will be
# empty or have weird values

# This separates (using tidyr) the AM and PM out from the attack_time variable
# the AM and PM will be in a new variable called 'when'
uboat.target.df = uboat.target.df %>%
  separate(attack_time, c('attack_timez', 'when'), sep=" ")

# Combine date and time columns into one (so we can relate the time to a date)
uboat.target.df$dateandtime <- as.character(paste(uboat.target.df$attack_date,
                                uboat.target.df$attack_timez, sep = ' '))
```

```r
# Then format the new variable to include date and time using as.POSIXct
uboat.target.df$dateandtime = as.POSIXct(uboat.target.df$dateandtime, format = "%Y-%m-%d %H.%M.%S")

# now we need to add 12hrs to the time column if the 'when' column has PM
# adding 12hrs involves using 12 * 60 * 60 to the time column

# This is how we add 12hrs need to multiply 60 seconds and 60 minutes and 12 hours
hourz = 12 * 60 * 60

# this is basically an if statement. There's probably a nicer way to do this in R...
# if the 'when' column is "PM" and the dateandtime column is not NA, add hourz to dateandtime
uboat.target.df$dateandtime[uboat.target.df$when == "PM" & !is.na(uboat.target.df$dateandtime)] =
  uboat.target.df$dateandtime[uboat.target.df$when == "PM" & !is.na(uboat.target.df$dateandtime)] + hou
# note because the times were formatted according to AM and PM, when adding 12 hours for only PM...
# we'll never need to worry about going past midnight and into the next day which would affect time.

# and check it has worked correctly. Refer to the View and confirm correct
#head(uboat.target.df)

# NAs====================
# We also want to deal with NA entries for numbers where we want to do calculations later.
# We don't want erroneous errors that may not be evident. So in this caes we'll change
# NA values to zero.
# Other times it may be more appropriate to change NA to the previous value or the
# mean or some other value.
#sum(is.na(uboat.target.df$dead))
#sum(is.na(uboat.target.df$complement))
uboat.target.df$dead[is.na(uboat.target.df$dead)] = 0
uboat.target.df$complement[is.na(uboat.target.df$complement)] = 0
```

That concludes the initial data wrangling. Dates and times are now all fixed. We can now begin thinking about the data we have, our questions and what variables we may want to generate ourselves.

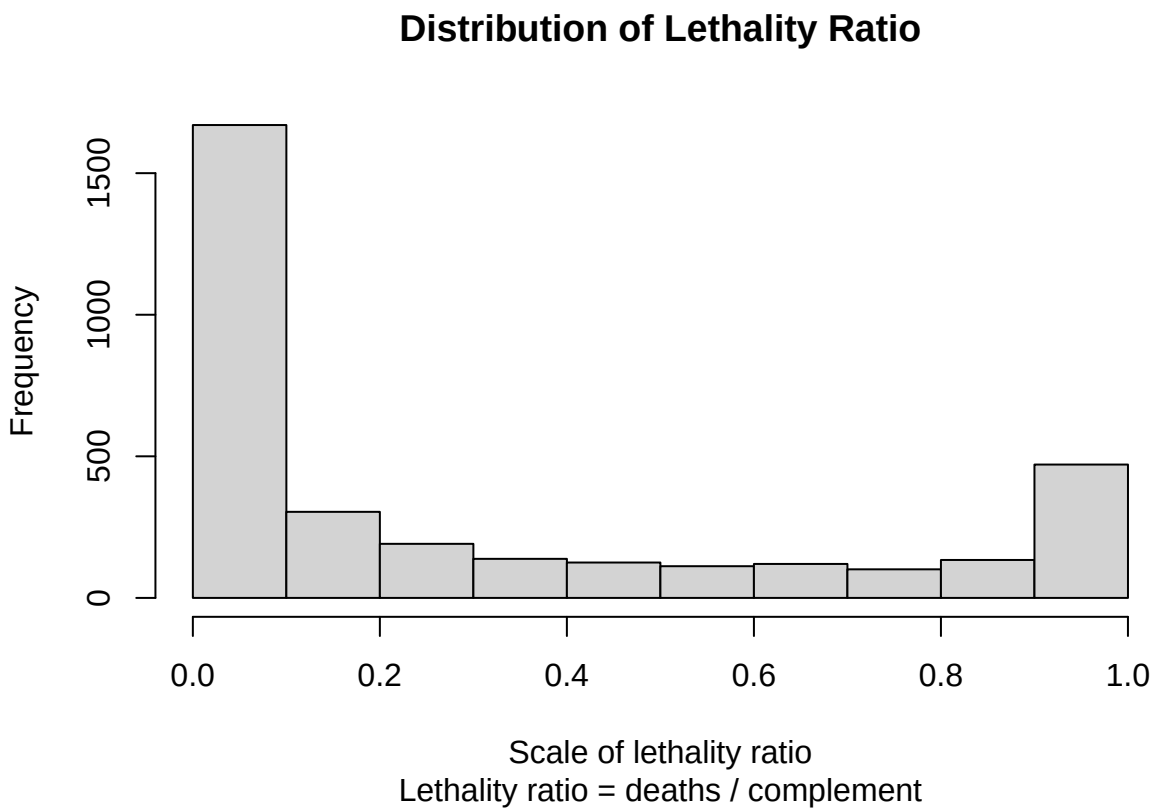
Variable creation

We can now start to consider ways we can deconstruct our data and variables and generate new potentially interesting variables we may like to explore our data with.

Lethality ratio

What would constitute the most lethal commander? Should total tonnage sunk be the metric? What about personnel killed? However these values would skew the results to favour those who targeted the largest ships. Is this the best? This *could* be useful. But how about calculating a ratio that takes those killed as a proportion of total personnel, being between zero and one. With this ratio or metric, a figure approaching one would mean almost all personnel on board are killed. A figure closer to zero would mean almost all personnel survive. This could provide another indication of how devastating an attack may be. A ship that eventually sunk may have sunk so slowly that all personnel onboard may have been able to escape. A more devastating attack may not have provided the time and killed more personnel. No single metric is complete on its own but each will enable different questions to be answered.

```r
# create new variable
uboat.target.df = uboat.target.df %>%
  mutate(lethality = round(dead / complement, 2))
uboat.target.df$lethality[is.nan(uboat.target.df$lethality)] = 0
# look at distribution of new variable
hist(uboat.target.df$lethality, main = "Distribution of Lethality Ratio",
     xlab = "Scale of lethality ratio",
     sub = "Lethality ratio = deaths / complement")
```



The above is an interesting distribution. Most attacks didn't kill many personnel, as a proportion. But there is an element of attacks that were very lethal. We'll see if we can build an understanding of this further as we go. We could easily filter and view our data based on lethality ratio > 0.8 but we'll keep going for now.

Time of day factor

We may also want to see how attacks fared based on the time of day. We do already have proper times setup now so partitioning our data based on some categories for time of day may prove insightful. We already have AM and PM which may be interesting so we'll keep that variable and just enrich it further with a supplemental variable.

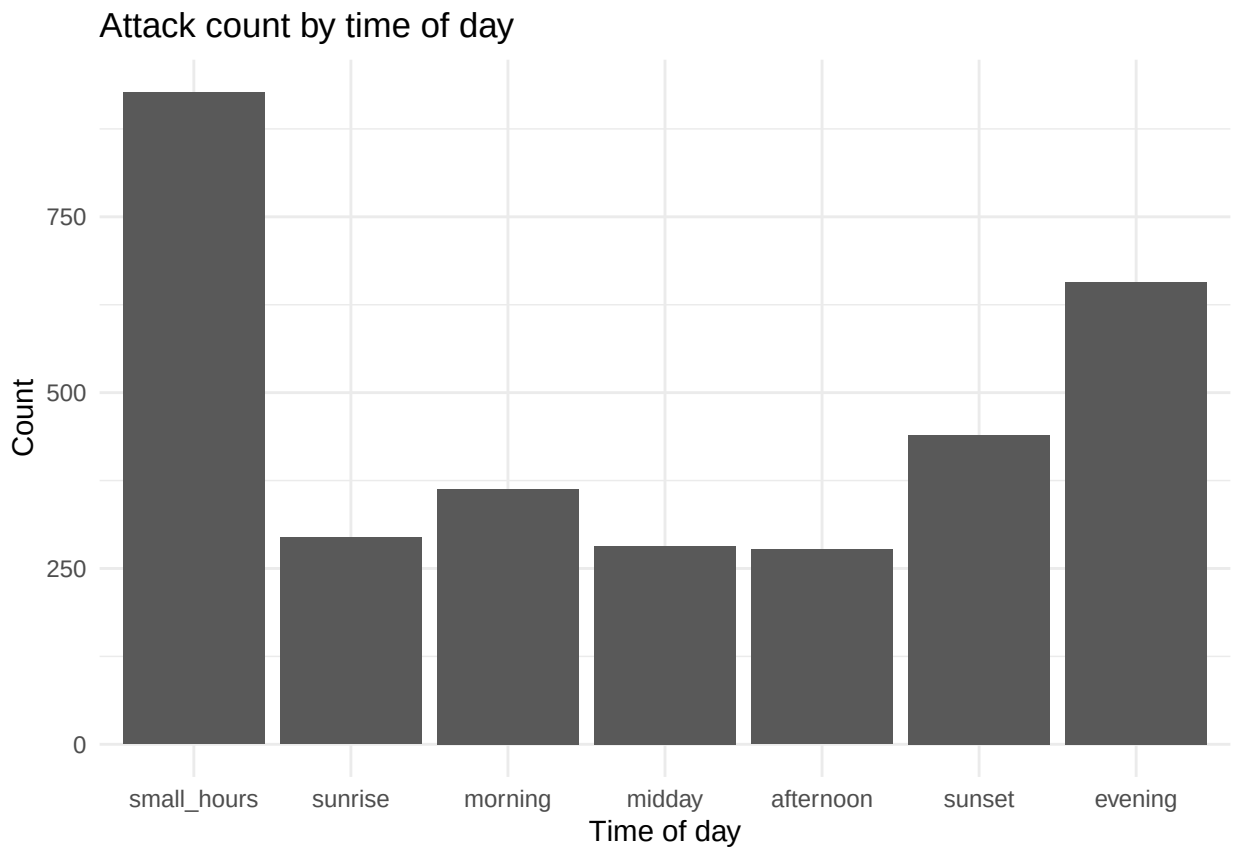
```r
uboat.target.df = uboat.target.df %>%
  mutate(day_period = case_when(
    hour(dateandtime) < 6 ~ "small_hours",
    hour(dateandtime) < 8 ~ "sunrise",
    hour(dateandtime) < 11 ~ "morning",
```

```
    hour(dateandtime) < 13 ~ "midday",
    hour(dateandtime) < 16 ~ "afternoon",
    hour(dateandtime) < 20 ~ "sunset",
    hour(dateandtime) < 24 ~ "evening",
    TRUE ~ "nil"
  ))
#head(uboat.target.df, 30)
kable(uboat.target.df %>%
  count(day_period) %>%
  arrange(desc(n)))
```

day_period	n
small_hours	927
evening	657
sunset	440
morning	363
sunrise	294
midday	282
afternoon	277
nil	218

We can see that the small hours (midnight to sunrise) and evening (sunset to midnight) are the preferred time periods to conduct attacks overall, with the small hours being the most popular. So it would appear that uboats were most dangerous and active at night. While the above is sorted by the count, the time of day is considered ordinal data, not nominal. This means there is an inherent order which shouldn't be changed. So we'll now plot the data in time of day order to get a different sense / perspective of the time of day counts.

```
uboat.target.df %>%
  filter(day_period != "nil") %>%
  group_by(day_period) %>%
  summarise(count = n()) %>%
  mutate(day_period = factor(day_period, levels = c("small_hours", "sunrise", "morning", "midday", "aft
  ggplot(aes(day_period, count)) +
  geom_bar(stat = 'identity') +
  labs(title = "Attack count by time of day",
       x = "Time of day",
       y = "Count") +
  theme_minimal()
```



The above plot visually represents the count of attacks by time of day in a more logical manner. This shows some nuance around the plotting / presenting of ordinal data. Ordinal data *can* be re-organised, but only after deliberate consideration. It's generally a bad practise though and should be discouraged.

Let's quickly count AM and PM. AM and PM will only give us a fairly crude metric but it could improve our understanding of our data. AM will be for attacks that happen in the first half of the day, and PM for the later half of the day.

```
kable(uboat.target.df %>%
  count(when) %>%
  arrange(desc(n)))
```

when	n
AM	1792
PM	1448
NA	198
	2
24	2
few	2
nine	2
two	2
12	1
14	1
31	1
36	1

when	n
38	1
60	1
Some	1
an	1
five	1
for	1

We can see that there seem to be more attacks in the first half of the day, likely due to the attacks in the small_hours, sunrise and morning. The difference doesn't seem large but it may still be a worthwhile metric to keep in the dataset.

We can also see there is some crappy data in the `when` variable which we should clean up while we're passing through. These are likely remnants from when we creating the `when` variable by pulling at out of the initial `time` variable that contained a string with AM and PM at the end, separated by a space. There's only a fairly small number of these so we won't worry about going back to our initial cleaning and seek to refine it. However, if this was a live dataset or we were aiming to disseminate our findings, we would go back and explore each individual case above to refine our initial data wrangling.

```
uboat.target.df$when[uboat.target.df$when != "AM" & uboat.target.df$when != "PM"
                     | is.na(uboat.target.df$when)] = "Nil"
#is.na(tt$when)
kable(uboat.target.df %>%
  count(when) %>%
  arrange(desc(n)))
```

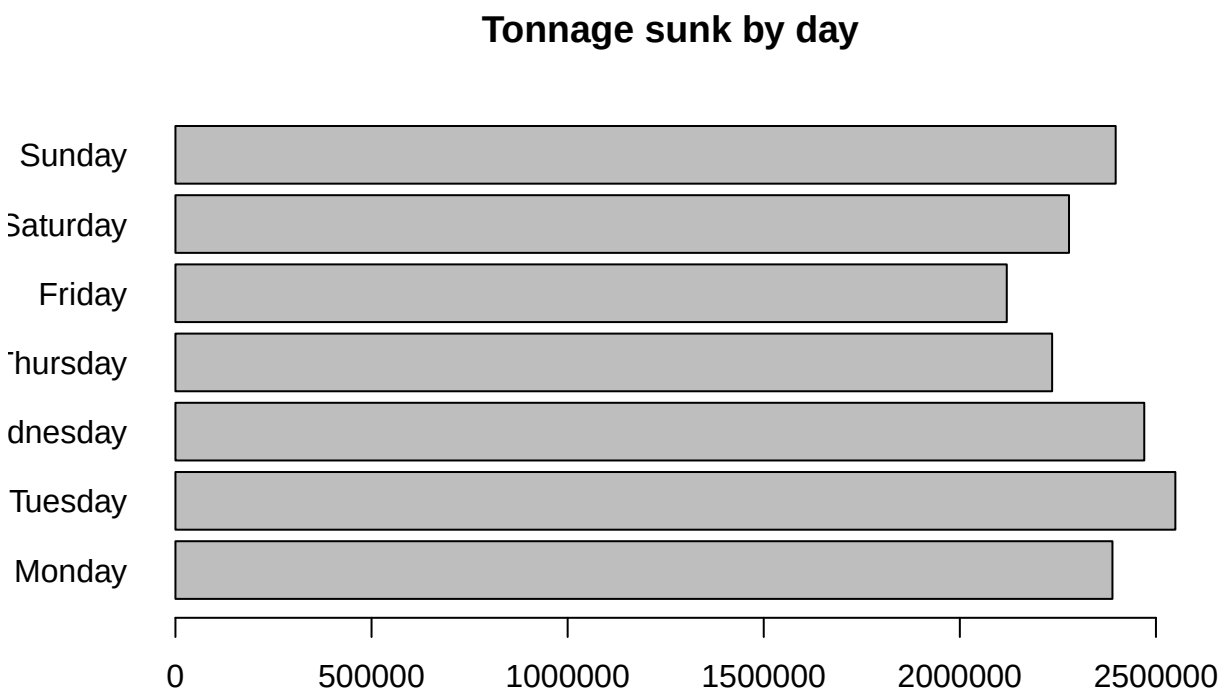
when	n
AM	1792
PM	1448
Nil	218

Day of week

Much like is of interest in other data, we may want to look at days of the week. However in this example, we'll use the sum of tonnage sunk, rather than just count the attacks.

```
# create weekday variable
uboat.target.df$weekday <- weekdays(uboat.target.df$dateandtime)

# group tonnage sunk by day and sum the amounts
uboat.day = uboat.target.df %>%
  group_by(weekday) %>%
  summarize(tonz = sum(tonnage))

uboat.day = na.omit(uboat.day)  # Need this otherwise there'll be a blank day for the NAs

# organise days into logical order
uboat.day = uboat.day %>%
  arrange(factor(weekday, levels = c("Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday"
```

```
barplot(uboat.day$tonz, names.arg = uboat.day$weekday, horiz=TRUE, las=1,
        main="Tonnage sunk by day")
```



There are no especially strong patterns here for now, though we could also measure the daily amount for other variables at a later point.

Time period served

We'll now create a new variable that measures the time a uboat served starting from the time it was launched into the water until it's ultimate fate. This metric may or may not contribute to its overall threat. Because dates are in the proper date format, R will automatically count the days. We'll simply print out the longest serving 6 for now. We may use this new variable later.

```
# create new variable for uboat.df dataframe
uboat.df = uboat.df %>%
  mutate(total_time = uboat.df$fate - uboat.df$launched)

# print out table for production
kable(uboat.df %>%
  mutate(total_time = uboat.df$fate - uboat.df$launched) %>%
  select(name, type, launched, fate, total_time) %>%
  arrange(desc(total_time)) %>%
  head())
```

name	type	launched	fate	total_time
U-8	IIB	1935-07-16	1945-03-31	3546 days
U-11	IIB	1935-08-27	1945-01-05	3419 days
U-17	IIB	1935-11-14	1945-02-06	3372 days
U-14	IIB	1935-12-28	1945-03-03	3353 days
U-9	IIB	1935-07-30	1944-08-20	3309 days
U-3	IIA	1935-07-19	1944-08-01	3301 days

```
#head(uboat.df)
```

Quickest to attack

We may want to find which uboats were the quickest from the time of launch to their first attack. While elements of luck and the tempo of the war are likely to be at play here, it may also identify competent crews and captains.

This will be the first variable where we need data from both dataframes so we'll need to join the elements we need as well.

```
# First attack date
earliest_attack = uboat.target.df[c("name", "attack_date", "commander")]
earliest_attack = earliest_attack %>%
  group_by(name, commander) %>%
  summarize(first_attack = min(attack_date))

# date of launch
launched = uboat.df[c("name", "launched")]

# inner join
quickest = earliest_attack %>%
  inner_join(launched, by = "name")

# Show sorted results
kable(quickest %>%
  mutate(time_to_attack = first_attack - launched) %>% # calculate new variable
  select(name, launched, commander, first_attack, time_to_attack) %>% # simply re-organising
  arrange(time_to_attack) %>%
  head(10))
```

name	launched	commander	first_attack	time_to_attack
U-63	1939-12-06	Gunther Lorentz	1940-02-24	80 days
U-55	1939-10-19	Werner Heidel	1940-01-18	91 days
U-102	1940-03-21	Harro von Klot-Heydenfeldt	1940-07-01	102 days
U-50	1939-11-01	Max-Hermann Bauer	1940-02-11	102 days
U-147	1940-11-16	Reinhard Hardegen	1941-03-02	106 days
U-99	1940-03-12	Otto Kretschmer	1940-07-05	115 days
U-138	1940-05-18	Wolfgang Luth	1940-09-20	125 days
U-206	1941-04-04	Herbert Opitz	1941-08-09	127 days
U-100	1940-04-10	Joachim Schepke	1940-08-16	128 days
U-204	1941-01-23	Walter Kell	1941-05-31	128 days

```
#head(quickest)
```

The above may or may not be of interest. Besides ordering by the time to the first attack, we've included not only the uboat names but the commanders. These names may appear again later in our analysis. There may or may not be a correlation or relationship between the `time_to_attack` variable and a commanders effectiveness.

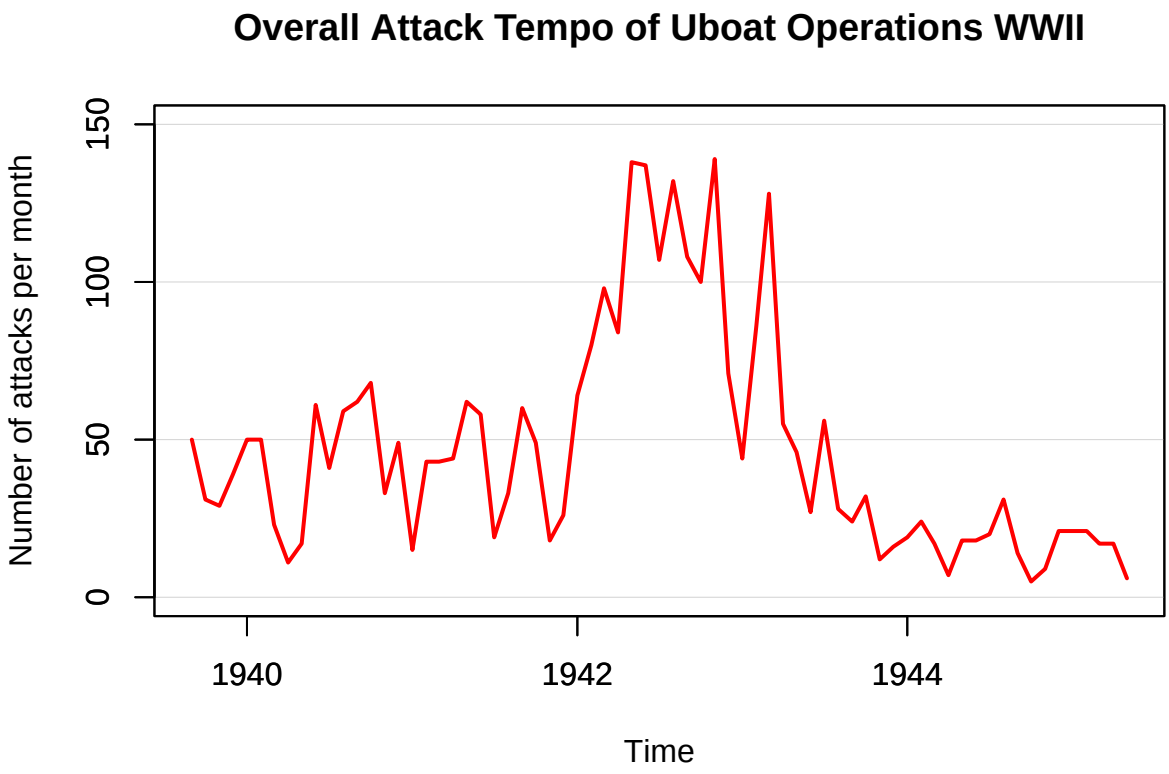
Data analyses

Now that we have our data in a reasonable condition, we start to actually analyse it with minimal stopping and starting. It's likely we'll want to do more with the data we have, but we can start be exploring the data as it is now. We should already have enough to glean some interesting insights.

Overall uboat attack tempo

To start with, let's take a big picture view of the tempo of attacks conducted by the uboats during WWII. Note that rather than use ggplot, we use the base R plot. We'll use both base R and ggplot throughout this project, just for fun and to show how they both work with different applications.

```
all_uboat_attacks = uboat.target.df %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  count(month)

# Note the below is exploring putting the grid lines behind the plot, using par(new = TRUE) and two plo
plot(all_uboat_attacks, type = 'n', , xlab = "", ylab="", ylim = range(c(0,150)))
grid(0, NULL, lty=1, lwd=0.5)
par(new = TRUE)
plot(all_uboat_attacks, type='l', col = 'red', lwd = 2,
     main = "Overall Attack Tempo of Uboat Operations WWII",
     xlab = "Time",
     ylab = "Number of attacks per month",
     ylim = range(c(0,150)))
```



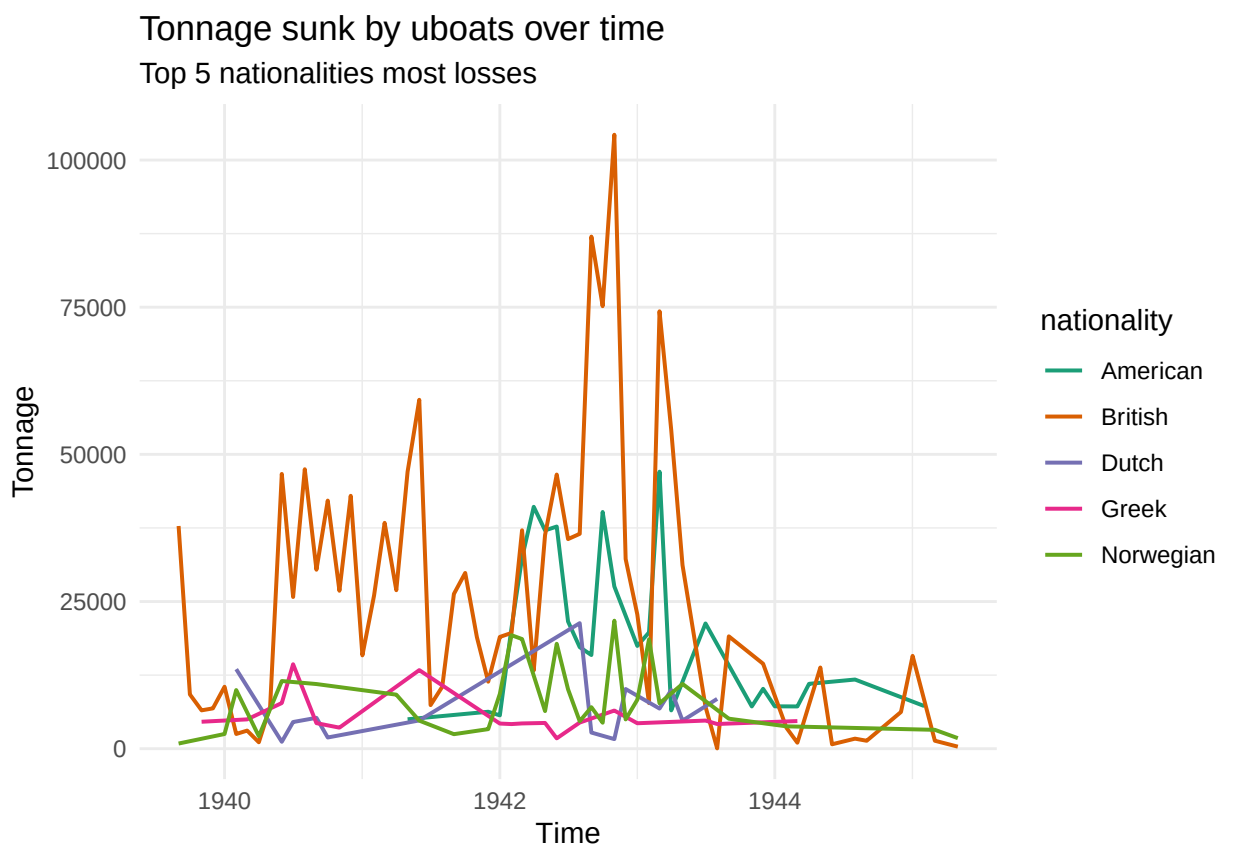
In the above plot, we do see an increase in uboat operations from early 1942 until around early to mid 1943 where the tempo drops off to below the pre 1942 levels. This gives us a broad baseline understanding of uboat operations before diving into some specifics.

Tonnage sunk by nationality over time

Let's now have a look at the tonnage of ships sunk over time, looking at only the countries with the five biggest losses overall.

```
# use ggplot to plot the above but seperate by target nationality
most_sunk = uboat.target.df %>%
  count(nationality) %>%
  arrange(desc(n)) %>%
  head(10)

uboat.target.df %>%
  #group_by(nationality) %>%
  filter(nationality == most_sunk[1:5,1]) %>%
  filter(loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month"), nationality) %>%
  summarize(tonz = sum(tonnage)) %>%
  ggplot(aes(month, tonz, col=nationality)) +
  #geom_smooth(se = FALSE) +
  geom_line(linewidth = 0.7) +
  #scale_colour_viridis_d()+
```

```
  scale_color_brewer(palette = "Dark2") +
  labs(title = "Tonnage sunk by uboats over time",
       subtitle = "Top 5 nationalities most losses",
       x = "Time",
       y = "Tonnage") +
  theme_minimal()
```



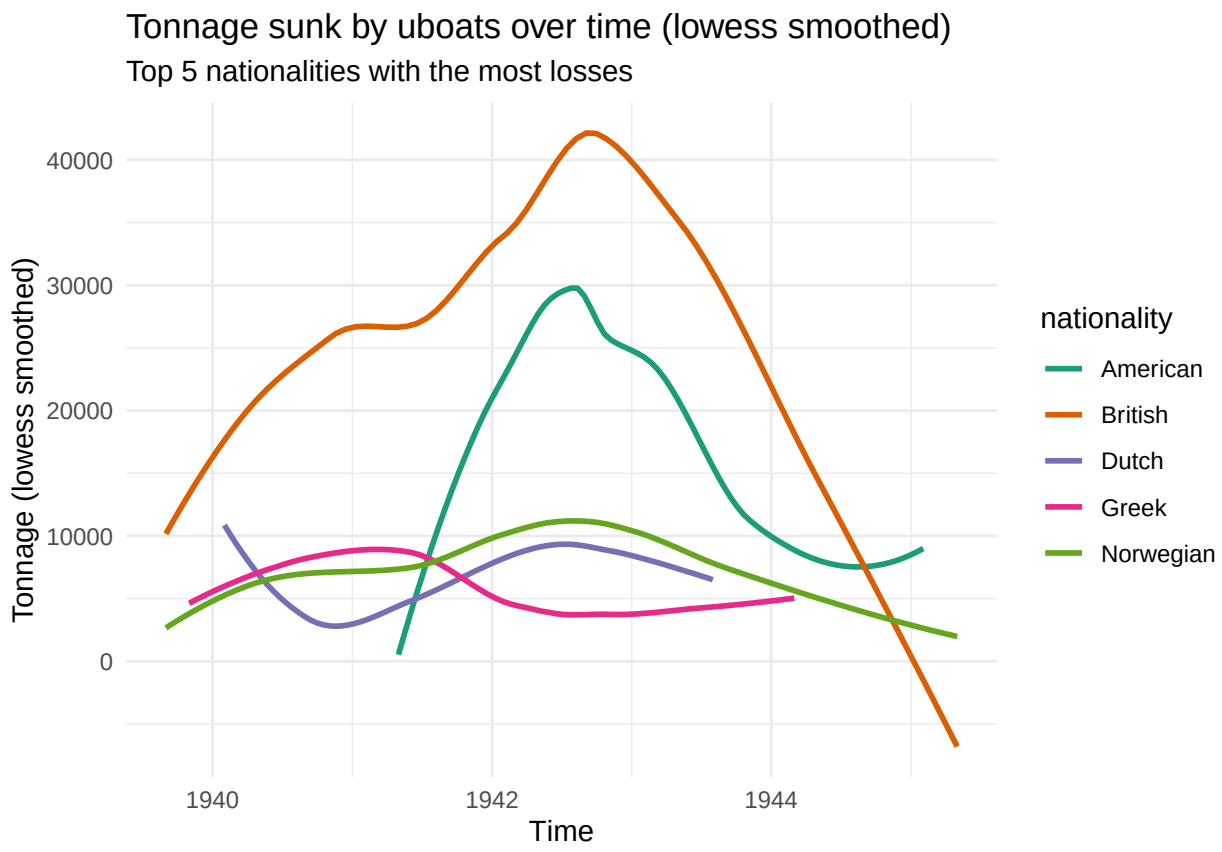

```
  # theme_classic()
```

In the above plot we can see the raw values. It is quite messy though we can see that uboats sunk British shipping the most, by tonnage. Second is the US. It does get a little messy comparing that middle section so it can be useful to fit smoothed lines to look at broader smoother trends over time. Such an approach will lose detail but can be more informative.

The below plot contains the exact same data, but now the raw tonnage is smoothed using the *lowess* method.

```
uboat.target.df %>%
  #group_by(nationality) %>%
  filter(nationality == most_sunk[1:5,1]) %>%
  filter(loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month"), nationality) %>%
  summarize(tonz = sum(tonnage)) %>%
  ggplot(aes(month, tonz, col=nationality)) +
  geom_smooth(se = FALSE) +
  #geom_line() +
```

```
  #scale_colour_viridis_d()+
  scale_color_brewer(palette = "Dark2") +
  labs(title = "Tonnage sunk by uboats over time (lowess smoothed)",
       subtitle = "Top 5 nationalities with the most losses",
       x = "Time",
       y = "Tonnage (lowess smoothed)") +
  theme_minimal()
```



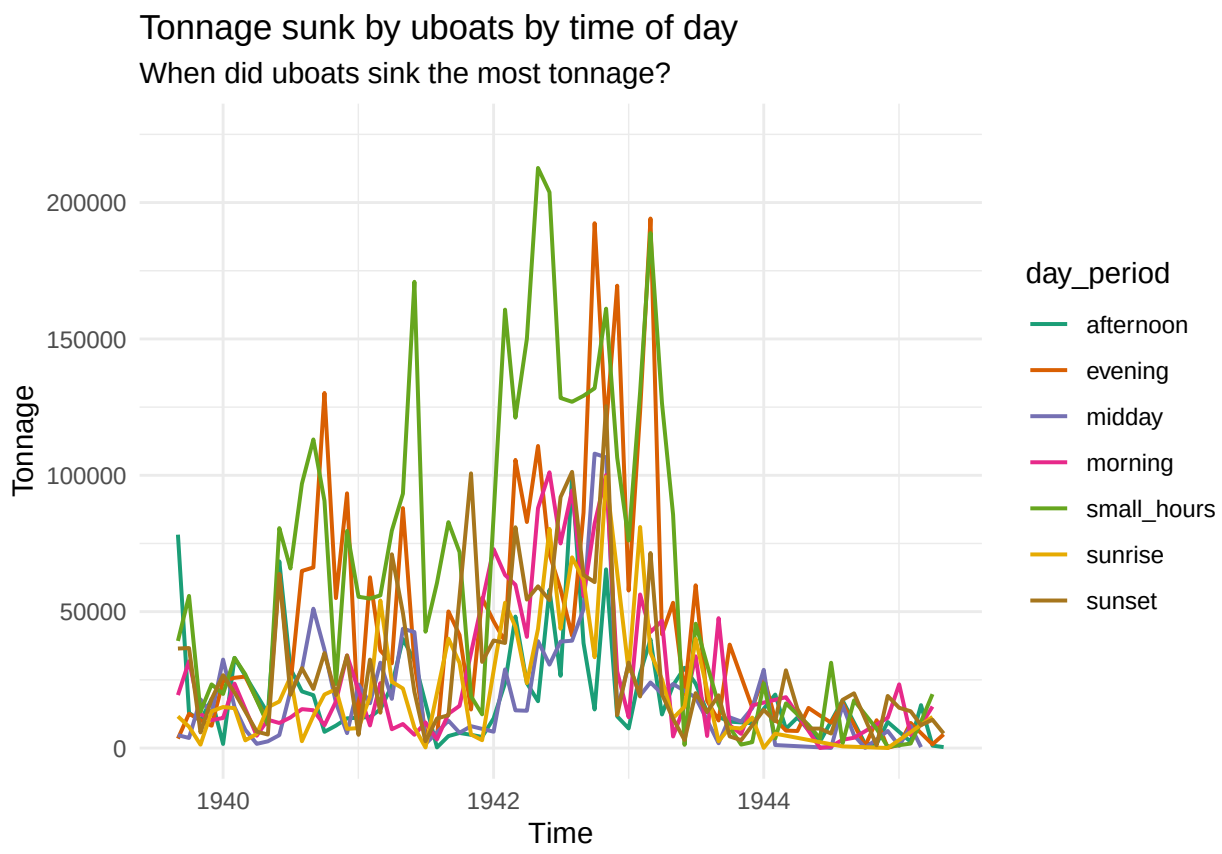
In the above plot, we can more clearly see the trends of the tonnage sunk over time and more easily compare the differences between nationalities. This plot actually shows that towards the end of our data, British tonnage lost drops below the US and Norwegian.

Attacks by day period over time

We'll now look at what time of the day where the most tonnage tended to be sunk, throughout the WWII period.

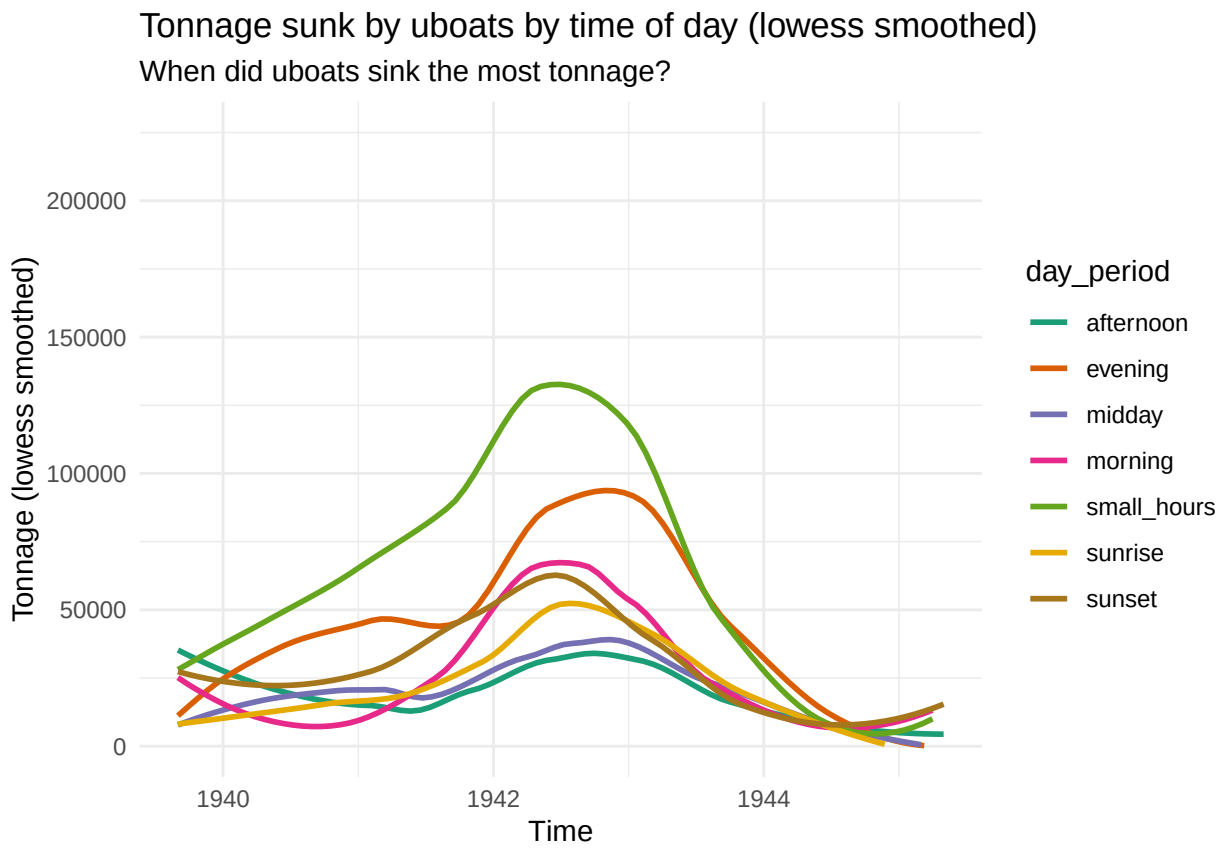
```
uboat.target.df %>%
  filter(loss_type == "Sunk" & day_period != "nil") %>%
  group_by(month=floor_date(dateandtime, "month"), day_period) %>%
  summarize(tonz = sum(tonnage)) %>%
  ggplot(aes(month, tonz, col=day_period)) +
  geom_line(linewidth = 0.7) +
  scale_color_brewer(palette = "Dark2") +
  ylim(0,225000) +
```

```
  labs(title = "Tonnage sunk by uboats by time of day",
       subtitle = "When did uboats sink the most tonnage?",
       x = "Time",
       y = "Tonnage") +
  theme_minimal()
```



The above plot is quite messy. It does appear that the small hours tended to be the time of day that the most tonnage was sunk, but it's not really clear looking at the raw data. Like before, lets smoothen it up and see if we can glean some clearer insights.

```
uboat.target.df %>%
  filter(loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month"), day_period) %>%
  summarize(tonz = sum(tonnage)) %>%
  ggplot(aes(month, tonz, col=day_period)) +
  geom_smooth(se = FALSE, span = 2/3) +
  scale_color_brewer(palette = "Dark2") +
  ylim(0,225000) +
  labs(title = "Tonnage sunk by uboats by time of day (lowess smoothed)",
       subtitle = "When did uboats sink the most tonnage?",
       x = "Time",
       y = "Tonnage (lowess smoothed)") +
  theme_minimal()
```



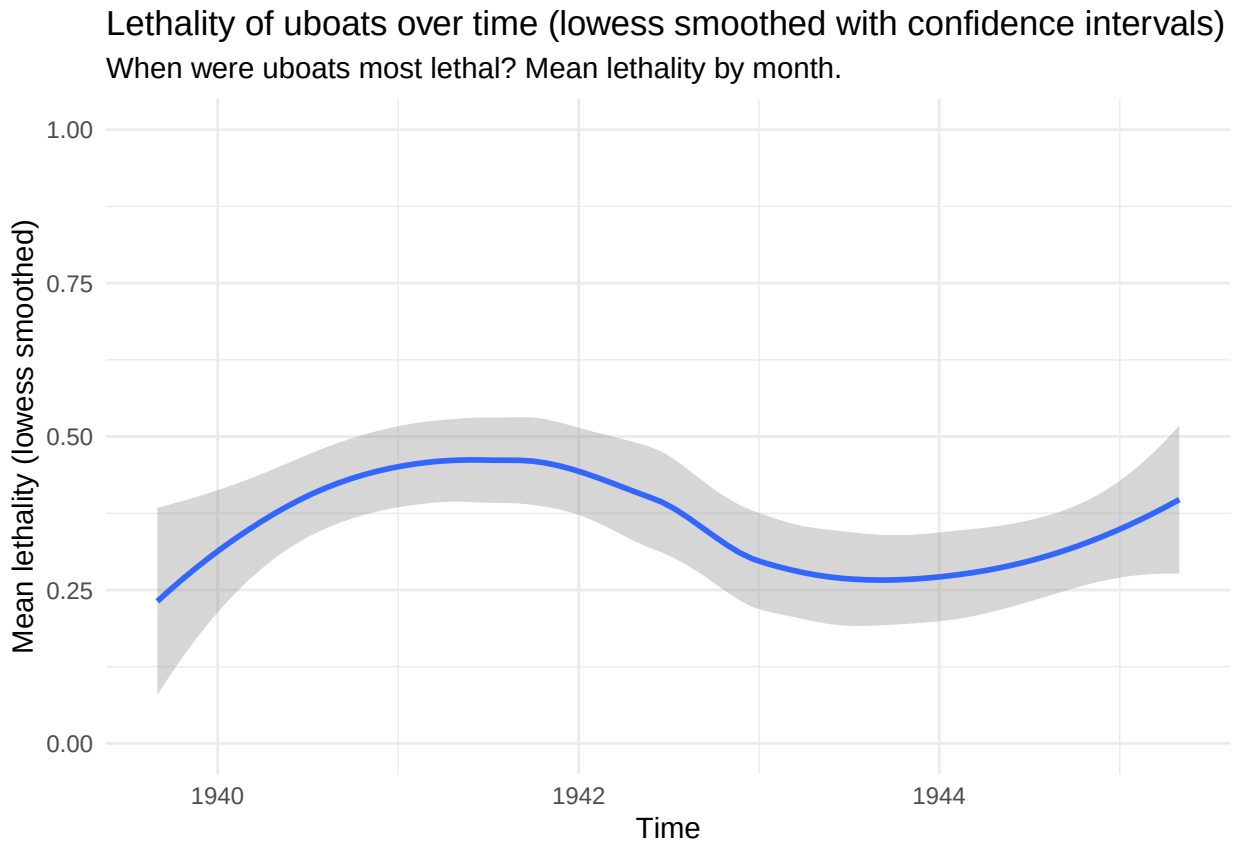
We can now see more clearly, that the most tonnage tended to be sunk in the small hours of the morning. This become a strong trend early in the conflict. The second highest threat time of day was the evening. Besides being quite high early in the conflict, the afternoon tended to be the safest time of day for maritime vessels from the uboat threat. For the first half of the conflict, sunset tended to be more likely to lose more tonnage than sunrise. Looking at this data over time seeks to identify changes. This may indicate a change in TTPs in either the attacking force (Uboats in this case) or defending forces. Perhaps this would be as a response to successful attacks.

Lethality of uboats over time

Let's now look at a similar smoothed lowess line chart but for the mean lethality rather than the time of day.

```
uboat.target.df %>%
  filter(loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  summarize(Lethality = mean(lethality)) %>%
  ggplot(aes(month, Lethality)) +
  #geom_line() +
  geom_smooth(span = 2/3, na.rm = TRUE) +
  #scale_color_brewer(palette = "Dark2") +
  ylim(0,1) +
  labs(title = "Lethality of uboats over time (lowess smoothed with confidence intervals)",
       subtitle = "When were uboats most lethal? Mean lethality by month.",
       x = "Time",
```

```
      y = "Mean lethality (lowess smoothed)") +
theme_minimal()
```



We can see that uboat lethality basically doubled from pre-1940 to the end of 1941 to almost 0.5. Then uboat lethality decreased back to around 0.25 towards the end of 1943 then began to increase again. These changes could warrant additional research to better understand. Perhaps this plot is showing us how allied responses to the uboat threat mitigated some of its lethality. Perhaps interestingly, when lethality began to drop into 1942, uboat tonnage rates increased. So this could also be the result of deliberate targeting choices being made by uboats around that time where a priority was placed on larger vessels for tonnage rather than on killing personnel. Likewise when we see an upward tick in lethality through 1944, this was roughly at the time that tonnage sunk was low.

These plots gave us a broad understanding of how uboat operations progressed over time at a high level.

Note the two previous plots used the *ggplot* graphical package which offers very rich customisation. These two plots only used some basic functionality, much more can be done to customise plots which includes building an entire custom theme which could be good for teams to quickly make nice looking on brand plots with minimal coding. Once a theme is built, all it requires is dropping the theme name in the plot code and that is all.

Most devastating attacks

We'll now begin exploring our data in more detail and seek to answer some of our questions. The first area we'll explore is to understand what were the most devastating attacks conducted by uboats. There are a few ways to measure this so we'll go through a few of them.

Top 10 sunk by biggest tonnage

We'll start by simply seeking to understand what were the biggest ships sunk (not just damaged) by uboats, by tonnage. We'll include a few other potentially interesting variables to assist us in building a picture of the uboat threat as we go through our analyses.

```
kable(uboat.target.df %>%
  select(name, loss_type, nationality, tonnage, commander, ship_name, dead, lethality, day_period) %>%
  filter(loss_type == "Sunk") %>%
  arrange(desc(tonnage)) %>%
  select(-loss_type) %>%
  head(10))
```

name	nationality	tonnage	commander	ship_name	dead	lethality	day_period
U-32	British	42348	Hans Jenisch	Empress of Britain	45	0.07	nil
U-331	British	31100	Hans-Diedrich Freiherr von Tiesenhausen	HMS Barham (04)	862	0.66	sunset
U-331	British	31100	Hans-Diedrich Freiherr von Tiesenhausen	HMS Barham (04)	862	0.66	sunset
U-47	British	29150	Gunther Prien	HMS Royal Oak (08)	833	0.69	small_hours
U-562	British	23722	Horst Hamm	Strathallan	16	0.00	small_hours
U-172	British	23456	Carl Emmermann	Orcades	45	0.04	morning
U-73	British	22600	Helmut Rosenbaum	HMS Eagle (94)	160	0.15	afternoon
U-81	British	22600	Friedrich Guggenberger	HMS Ark Royal (91)	1	0.00	sunset
U-29	British	22500	Otto Schuhart	HMS Courageous (50)	518	0.41	afternoon
U-99	British	20638	Otto Kretschmer	Terje Viken	2	0.02	small_hours

Before we start looking for understanding, we can see what appears to be a duplicate entry for the U-331 sinking of HMS Barham. I looked through the full dataframe and all details are identical. This is something to be mindful of as we start counting sums of bigger data we may not be able to look at entirely with our own eyes. We may later conduct some deliberate searches for duplicates. In the meantime, let's look for interesting findings.

The attacks on HMS Barham and HMS Royal Oak seem especially devastating. Not only are the tonnages large, we can see high numbers of killed personnel and fairly high lethality for such large vessels. Somewhat interestingly, or not, the largest ships sunk were British. This could be more a reflection of the size of the British navy and maritime industries than anything else. Or it could show a preference or attractiveness for striking British vessels. While this is only our first top ten list, we see no standout uboats, commanders or preferred attack day period as yet. But this table allows us to start building our understanding of the uboat threat.

Top 10 by most personnel killed

We'll now do something very similar, though will rank by the most personnel killed.

```
kable(uboat.target.df %>%
  select(name, nationality, tonnage, commander, ship_name, dead, lethality, day_period) %>%
  arrange(desc(dead)) %>%
  head(10))
```

name	nationality	tonnage	commander	ship_name	dead	lethality	day_period
U-331	British	31100	Hans-Diedrich Freiherr von Tiesenhausen	HMS Barham (04)	862	0.66	sunset
U-331	British	31100	Hans-Diedrich Freiherr von Tiesenhausen	HMS Barham (04)	862	0.66	sunset
U-177	British	6796	Robert Gysae	Nova Scotia	858	0.82	sunrise
U-47	British	29150	Gunther Prien	HMS Royal Oak (08)	833	0.69	small_hours
U-486	Belgian	11509	Gerhard Meyer	Leopoldville	819	0.34	sunset
U-47	British	15501	Gunther Prien	Arandora Star	805	0.48	sunrise
U-559	British	3059	Hans Heidtmann	Shuntien	700	Inf	sunset
U-223	American	5649	Karl-Jurg Wachter	Dorchester	675	0.75	small_hours
U-156	British	19695	Werner Hartenstein	Laconia	658	0.24	evening
U-515	British	18713	Werner Henke	Ceramic	654	1.00	nil

The attack on HMS Barham actually resulted in the most personnel killed as well. We also start to see some targets from other nationalities here. These attacks resulted in significant loss of life. This simple table, and the one before it, does show the significant effect uboats were having in the battlespace around Europe during WWII.

For the first time, we see one uboat and commander feature twice in the top ten. U-47 and Commander Gunther Prien sunk HMS Royal Oak and the Arandora Star which together resulted in over 1,000 deaths. Perhaps interestingly, there does seem to be a higher proportion of attacks that occurred around sunset or sunrise compared to the other times of day.

Top 10 most lethal devastating attacks

We'll now look at the top ten most devastating attacks by the highest lethality. There are many instances of attacks in the dataset where all crew were lost so filtering by a lethality of 1 would produce a very long list. So we've kept increasing the filter for number of dead higher until we start to introduce lethality rates of lower than 1 in our list. We got up to 150 killed before we saw less than ten results with lethality of 1. Remember, lethality is simply the ratio of those killed over the total number of personnel on board. So a lethality of 1 means all personnel were killed. A lethality of 0 means no personnel were killed, all survived.

```
kable(uboat.target.df %>%
  select(name, nationality, tonnage, commander, ship_name, dead, lethality, day_period) %>%
  filter(dead > 150) %>%
  arrange(desc(lethality)) %>%
  head(10))
```

name	nationality	tonnage	commander	ship_name	dead	lethality	day_period
U-559	British	3059	Hans Heidtmann	Shuntien	700	Inf	sunset
U-22	British	1475	Karl-Heinrich Jenisch	HMS Exmouth (H 02)	190	1.00	small_hours
U-43	German	5154	Hans-Joachim Schwantke	Doggerbank	364	1.00	evening
U-96	British	14936	Heinrich Lehmann-Willenbrock	Almeda Star	360	1.00	small_hours
U-103	British	9515	Viktor Schutze	Calabria	360	1.00	evening
U-224	Canadian	3921	Hans-Karl Kosbadt	Bic Island	165	1.00	midday
U-404	British	1120	Otto von Bulow	HMS Veteran (D 72)	235	1.00	morning
U-515	British	18713	Werner Henke	Ceramic	654	1.00	nil
U-604	American	7057	Horst Holtring	Coamo	186	1.00	evening
U-454	British	1870	Burckhard Hacklander	HMS Matabele (G 26)	236	0.99	evening

The first thing that stands out here is there is a German vessel sunk by a uboat. A quick search online confirmed our suspicions that the Doggerbank was sunk by U-43 by mistake. The Doggerbank was a former UK cargo ship that was captured by the German Navy in 1941 and converted to an auxiliary minelayer and blockade runner. This is likely why it was sunk. There was perhaps some vulnerability in the uboat targeting and authorisation process that could have been exploited here.

There does seem to be far more attacks here that occurred in the evening, especially if combined with the attacks in the small hours. Perhaps attacks that happened during these times resulted in more confusion on the vessel, resulting in more delayed evacuation as well as impact the ability of rescue vessels to assist during the night. It's also possible that night allowed the uboats to use stealth to maximum effect and achieve more optimum target acquisition and launch to achieve a more effective hit on the targets.

Understanding the uboat types

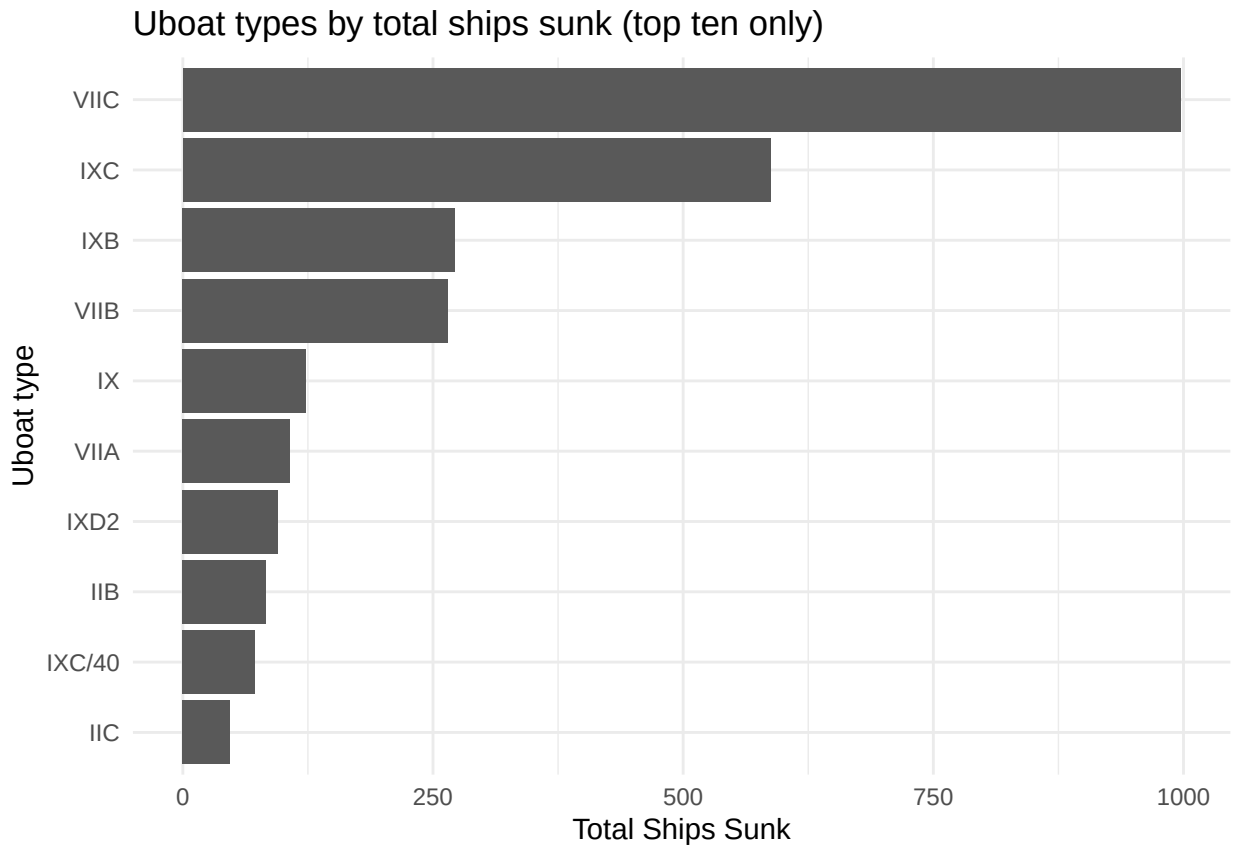
There are different uboat types as shown by the different designations. However these types are not included in the target dataset, but the base uboat dataset.

```
unique(uboat.df$type)
```

```
##  [1] "IIA"     "IIB"     "IA"      "VIIA"    "IX"      "VIIB"    "IIC"
##  [8] "IXB"     "IXC"     "VIIC"    "XB"      "IID"     "IXC/40"  "IXD2"
## [15] "IXD1"    "VIID"    "VIIC/41" "XIV"     "XVIIA"   "IXD/42"  "VIIF"
## [22] "XVIIB"   "XXIII"   "XXI"
```

However the uboat dataset does have a metric for the total ships sunk for each uboat. We could extract these types and join them with the attack dataset, but for now we'll compare the uboat types by counting the ships sunk.

```r
uboat.df %>%
  group_by(type) %>%
  summarize(sunk = sum(ships_sunk)) %>%
  arrange(desc(sunk)) %>%
  head(10) %>%
  ggplot(aes(reorder(type, sunk), sunk)) +
  coord_flip() +
  geom_bar(stat = 'identity') +
  labs(title = "Uboat types by total ships sunk (top ten only)",
       y = "Total Ships Sunk",
       x = "Uboat type") +
  theme_minimal()
```



Complementing the above barplot we can see that via wikipedia the VIIC was the German attack submarine 'workhorse' for the majority of WWII. While that may be the case, the IXC also presented a significant threat. The IXC is identified as an upgraded ocean boat used in the opening years of WWII. The IXB and VIIB were identified as 'standard' fleet ocean boat or 'standard' attack submarines, though both types sunk hundreds of vessels. Regardless, the VIIC type represented the most significant threat if we take the count of ships sunk as the key metric.

Most effective uboat commander

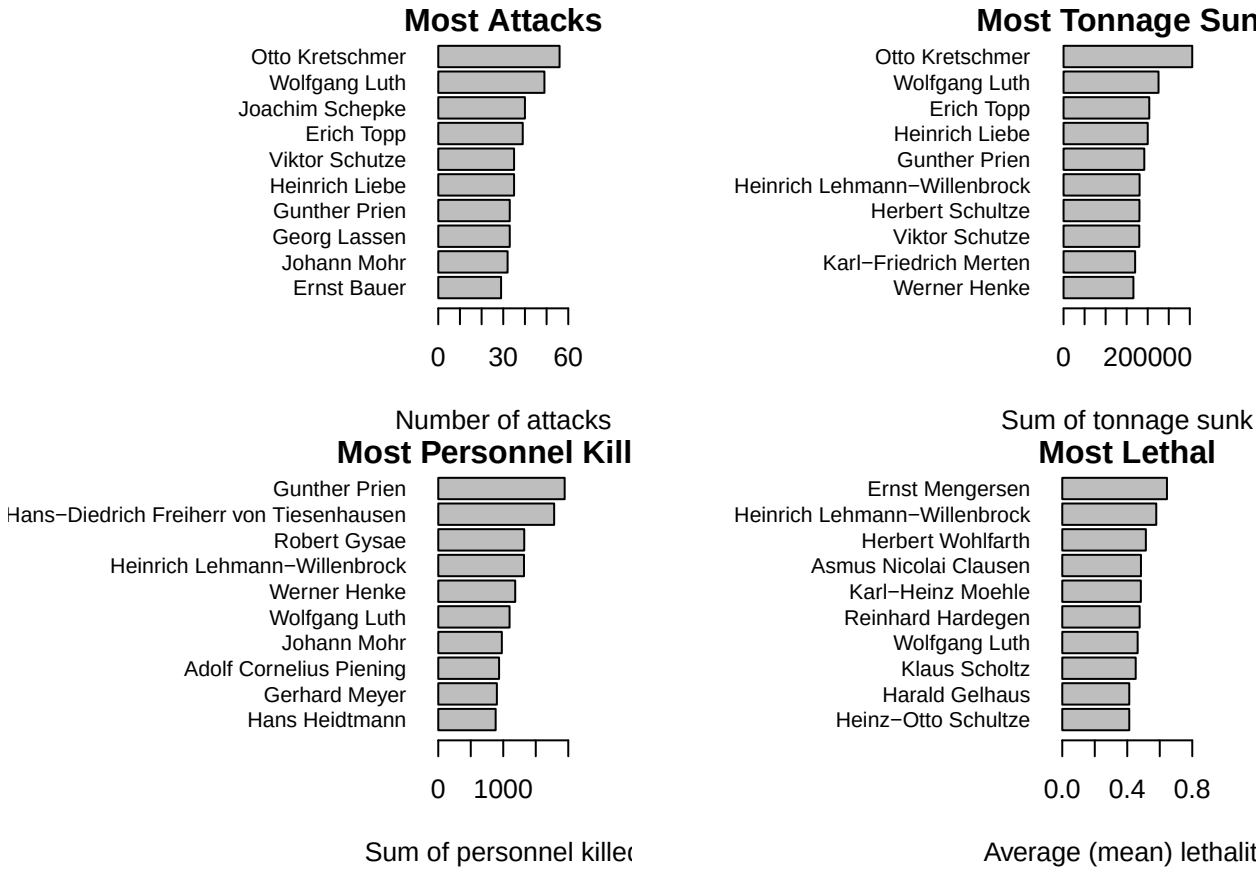
Let's now move on to analyse the uboat commanders in more detail. There are a few different ways we could look to identify the most effective uboat commanders. We could look at those who conducted the most attacks, those who sunk the most tonnage, those who killed the most personnel or those who scored

the highest on our lethality ratio. We could also build an index of sorts considering each of these. But we won't do that here. Perhaps you could try to develop an index considering each of the metrics?

We could also plot all of these figures in one graphic and compare the various results and look for the commanders who feature in more than one metric. Note that for the lethality ratio, because it would be fairly easy to score highly with one attack that may have killed 5 out of 5 personnel, we've filtered this plot to only include commanders who conducted more than 15 attacks. This filter allows us to ensure we are only capturing skilled consistent lethality and not a one-off fluke.

```
# most attacks
comd_attacks = uboat.target.df %>%
  count(commander) %>%
  arrange(desc(n)) %>%
  head(10) %>%
  arrange(n)

# most tonnage
most_tonnage = uboat.target.df %>%
  group_by(commander) %>%
  filter(loss_type == "Sunk") %>%
  summarize(tonnagez = sum(tonnage)) %>%
  arrange(desc(tonnagez)) %>%
  head(10) %>%
  arrange(tonnagez)

# most kills
most_killed = uboat.target.df %>%
  group_by(commander) %>%
  summarize(killed = sum(dead)) %>%
  arrange(desc(killed)) %>%
  head(10) %>%
  arrange(killed)

# commanders with at least n attacks
five_attacks = uboat.target.df %>%
  count(commander) %>%
  filter(n > 15)

# mean of lethality
most_lethal = uboat.target.df %>%
  filter(lethality < 1.1) %>%
  group_by(commander) %>%
  summarize(lethal = mean(lethality))

# left join lethality onto at least n
five = five_attacks %>%
  left_join(most_lethal, by = "commander") %>%
  arrange(desc(lethal)) %>%
  head(10) %>%
  arrange(lethal)

#quickest = earliest_attack %>%
 # inner_join(launched, by = "name")

# start plotting
```

```
par(mfrow = c(2,2), mar = c(4,13.5,1,2))

barplot(comd_attacks$n, names.arg = comd_attacks$commander, horiz = TRUE, las=1,
        xlim = c(0,60), cex.names = 0.8,
        main = "Most Attacks",
        xlab = "Number of attacks")

barplot(most_tonnage$tonnagez, names.arg = most_tonnage$commander, horiz = TRUE, las=1,
        cex.names = 0.8,
        main = "Most Tonnage Sunk",
        xlab = "Sum of tonnage sunk")

barplot(most_killed$killed, names.arg = most_killed$commander, horiz = TRUE, las=1,
        xlim = c(0,2000), cex.names = 0.8,
        main = "Most Personnel Killed",
        xlab = "Sum of personnel killed")

barplot(five$lethal, names.arg = five$commander, horiz = TRUE, las=1, cex.names = 0.8,
        xlim = c(0,0.8),
        main = "Most Lethal",
        xlab = "Average (mean) lethality")
```



From the above we can see that Otto Kretschmer appeared to be a very effective uboat commander. Not only did he conduct the most attacks he also sunk the most tonnage, by some margin. Wolfgang Luth was second for both the number of attacks and tonnage sunk. Erich Topp was the next most effective commander by these two metrics. Gunther Prien killed the most personnel, followed by Hans-Diedrich Freiherr von

Tiesenhausen. These two killed the most personnel by some margin. Gunther Prien is noteworthy for featuring in three of the four plots.

Remember that the most lethal plot only includes commanders who conducted more than 15 attacks. Heinrich Letmann-Willenbrock featured not only the second highest for most lethal, but also fourth for the most personnel killed and sixth for most tonnage sunk. He is one of the few commanders who featured on at least three of these plots.

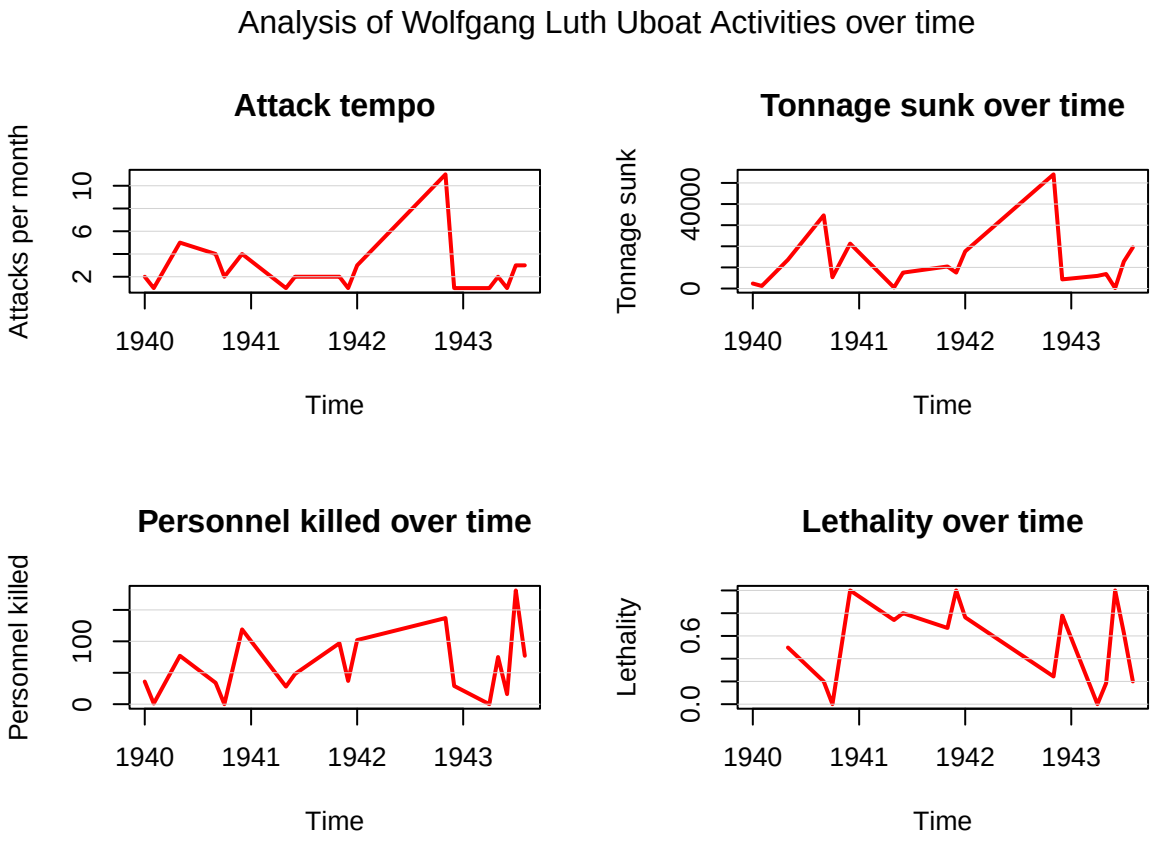
Wolfgang Luth is the standout here as he features in all four plots and as high as two in two plots, behind Otto Kretschmer.

Wolfgang Luth

Let's now take a closer look at Wolfgang Luth to see if we can better understand his *modus operandi*.

```
# count the activity of wolfgang by month
# this will give only months and n values
wolf = uboat.target.df %>%
  filter(commander == "Wolfgang Luth") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  count(month)

# wolf tonnage
wolf_tonnage = uboat.target.df %>%
  filter(commander == "Wolfgang Luth" & loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  summarize(tonnz = sum(tonnage))

# wolf killed
wolf_killed = uboat.target.df %>%
  filter(commander == "Wolfgang Luth" & loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  summarize(killed = sum(dead))

# wolf lethality
wolf_lethality = uboat.target.df %>%
  filter(commander == "Wolfgang Luth" & loss_type == "Sunk") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  summarize(lethal = mean(lethality))

# plotting
par(mfrow= c(2,2), oma=c(0,1,1,0))
#xl = c("1940", "1944")
plot(wolf, type = 'l', col = 'red', lwd = 2, main = "Attack tempo",
     xlab = "Time",
     ylab = "Attacks per month")
grid(0, NULL, lty=1, lwd=0.5)
plot(wolf_tonnage$month, wolf_tonnage$tonnz, type = 'l', col='red', lwd= 2,
     main = "Tonnage sunk over time",
     xlab = "Time",
     ylab = "Tonnage sunk")
grid(0, NULL, lty=1, lwd=0.5)
plot(wolf_killed$month, wolf_killed$killed, type = 'l', col='red', lwd= 2,
     main = "Personnel killed over time",
```

```
     xlab = "Time",
     ylab = "Personnel killed")
grid(0, NULL, lty=1, lwd=0.5)
plot(wolf_lethality$month, wolf_lethality$lethal, type = 'l', col='red', lwd= 2,
     main = "Lethality over time",
     xlab = "Time",
     ylab = "Lethality")
grid(0, NULL, lty=1, lwd=0.5)
mtext("Analysis of Wolfgang Luth Uboat Activities over time", outer = TRUE, side=3)
```



The attack tempo plot shows simply the number of attacks conducted each month. The tonnage sunk plot shows the sum of tonnage sunk for each month. The personnel killed plot also sums the personnel killed for each month. The lethality plot contains the mean average of the lethality rate for each month.

There are some significant gaps in the activities of Wolfgang Luth, likely between times being away on patrol. There is about an eleven month gap between Jan 1942 and Nov 1942. However in Nov Wolfgang conducted 10 attacks that month, all resulting in the targets being sunk. The lethality ratios for these individual attacks ranged from zero to 0.83 (note the plot for lethality is monthly mean values for lethality). The attack in 1943 where he killed the most personnel was the following:

```
kable(uboat.target.df %>%
  filter(commander == "Wolfgang Luth" & dead > 100) %>%
  select(name, loss_type, attack_date, nationality, ship_name, dead, lethality, day_period))
```

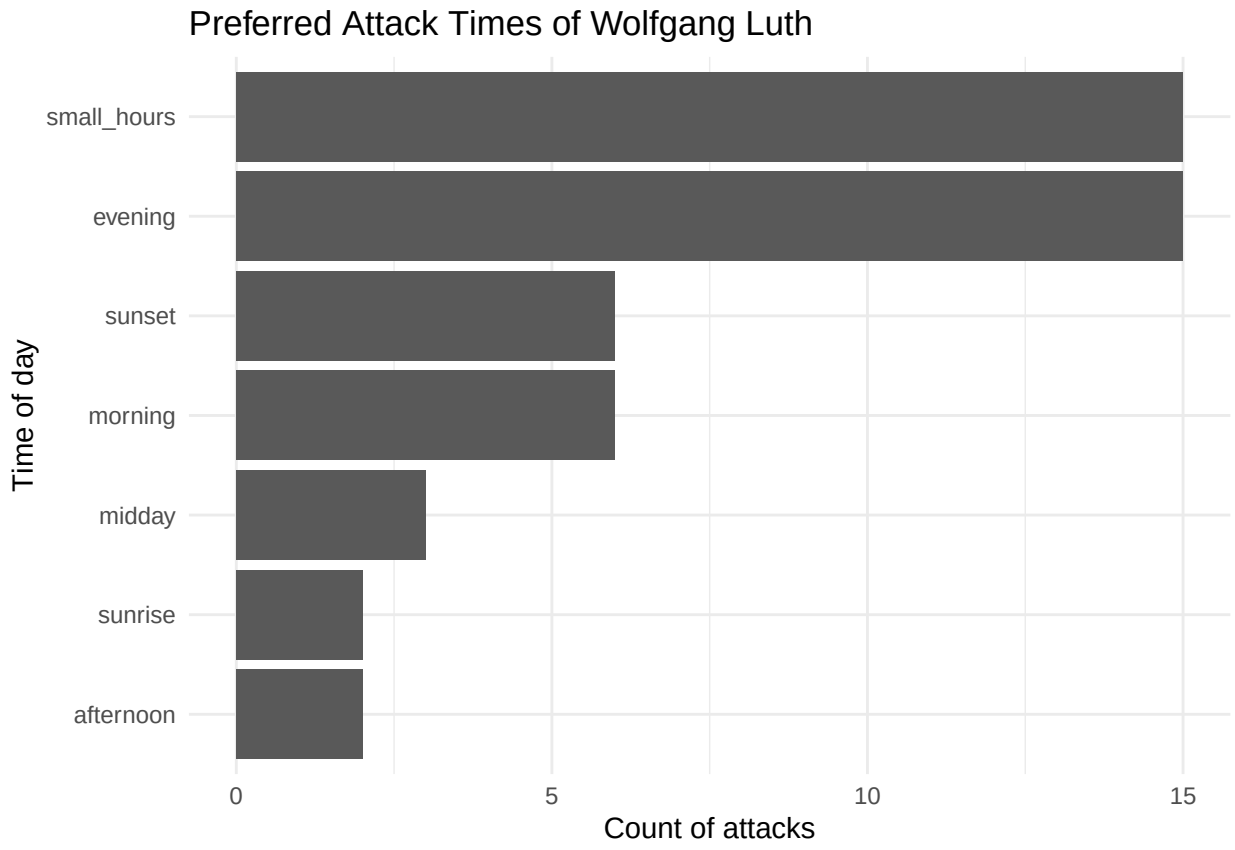
name	loss_type	attack_date	nationality	ship_name	dead	lethality	day_period
U-181	Sunk	1943-07-02	British	Hoihow	145	0.97	evening

Throughout his career, he captained the following uboats:

- U-9
- U-43
- U-138
- U-181

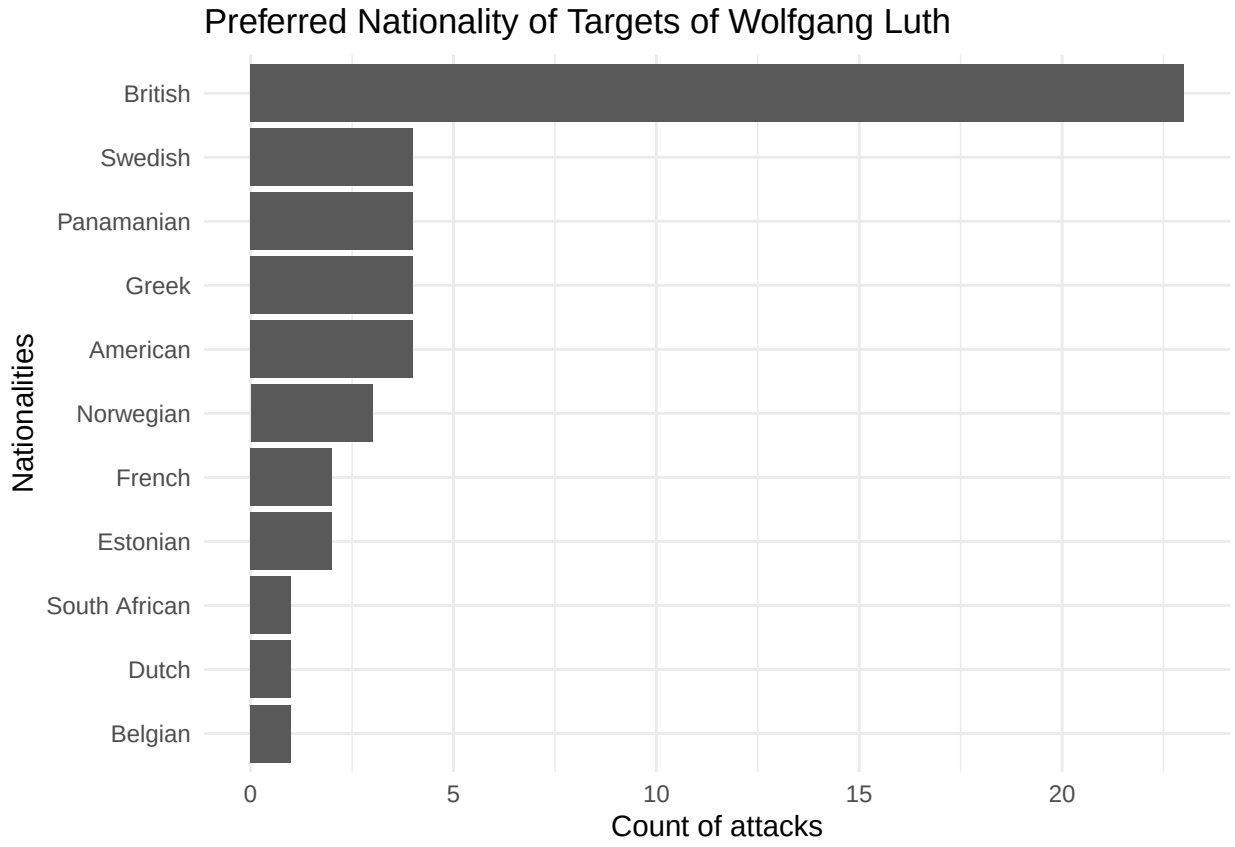
We'll now look into some of the other particulars regarding how Wolfgang Luth conducted attacks.

```
uboat.target.df %>%
  filter(commander == "Wolfgang Luth") %>%
  count(day_period) %>%
  arrange(desc(n)) %>%
  ggplot(aes(reorder(day_period, n), n)) +
  coord_flip() +
  geom_bar(stat = 'identity') +
  labs(title = "Preferred Attack Times of Wolfgang Luth",
       x = "Time of day",
       y = "Count of attacks") +
  theme_minimal()
```



Wolfgang Luth was a night hunter. Most of his attacks were either in the evening or small hours of the morning before sunrise. We need to be conscious of the ordering in the above. Is that ok?

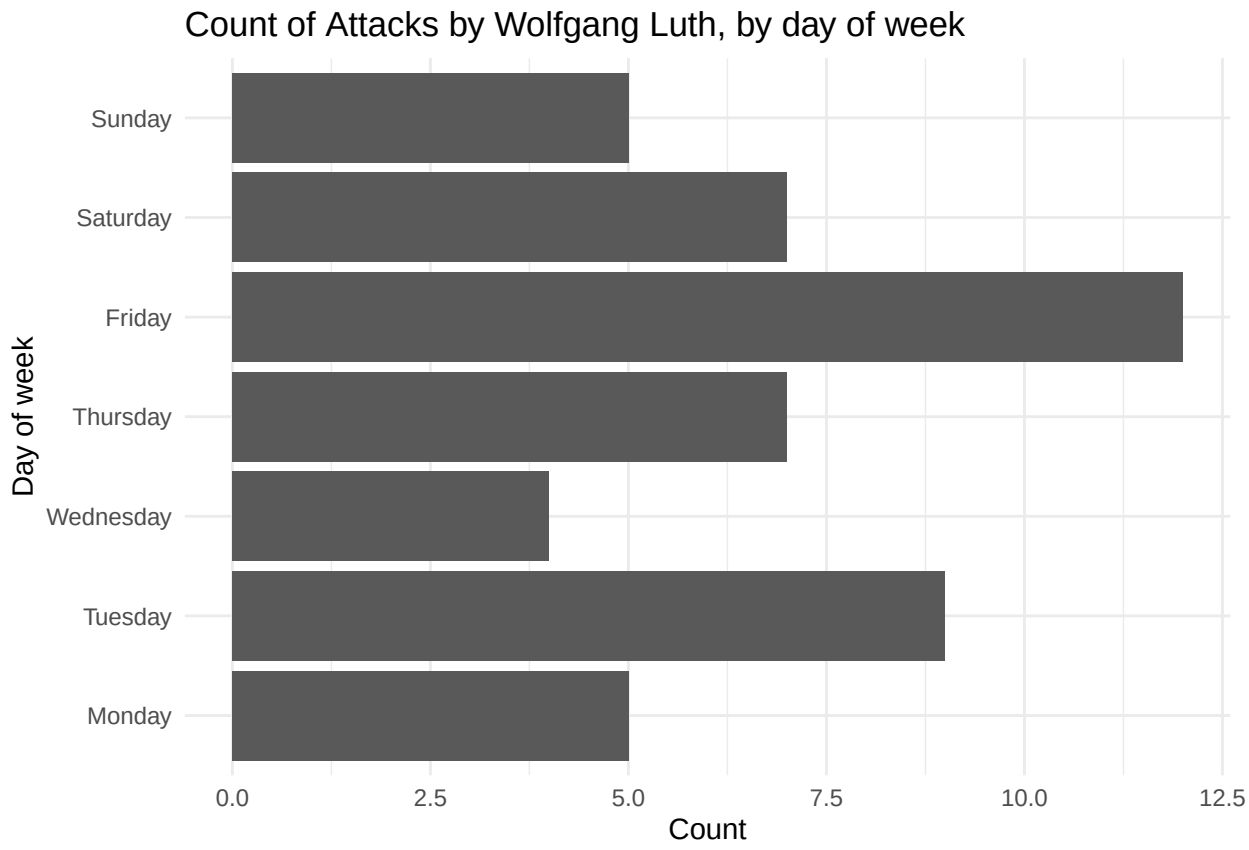
```
uboat.target.df %>%
  filter(commander == "Wolfgang Luth") %>%
  count(nationality) %>%
  arrange(desc(n)) %>%
  ggplot(aes(reorder(nationality, n), n)) +
  coord_flip() +
  geom_bar(stat = 'identity') +
  labs(title = "Preferred Nationality of Targets of Wolfgang Luth",
       x = "Nationalities",
       y = "Count of attacks") +
  theme_minimal()
```



Wolfgang struck mostly British vessels. This could be more due to the sheer number of British vessels in his AO. It might actually be interesting to measure the proportion of targets which British from all of the data then compare with individual commanders. But for now, we'll leave that.

```
uboat.target.df %>%
  filter(commander == "Wolfgang Luth") %>%
  count(weekday) %>%
  mutate(weekday = factor(weekday, levels = c("Monday", "Tuesday", "Wednesday", "Thursday",
                                              "Friday", "Saturday", "Sunday"))) %>%
  ggplot(aes(weekday, n)) +
  coord_flip() +
```

```
  geom_bar(stat = 'identity') +
  labs(title = "Count of Attacks by Wolfgang Luth, by day of week",
       x = "Day of week",
       y = "Count") +
  theme_minimal()
```



There may not be any significance to the plot above, but Wolfgang Luth conducted the most attacks on Fridays, and the least attacks on Wednesdays. This is quite different to the overall Uboat attacks by day of the week. So while it may justify additional qualitative research, it may also be a coincidence.

In a possibly fitting conclusion to the Wolfgang Luth story, he was accidentally shot and killed by a German soldier standing guard who asked Luth for the password three times without response - so the guard fired and killed him. More on Luth can be found at the wikipedia page.

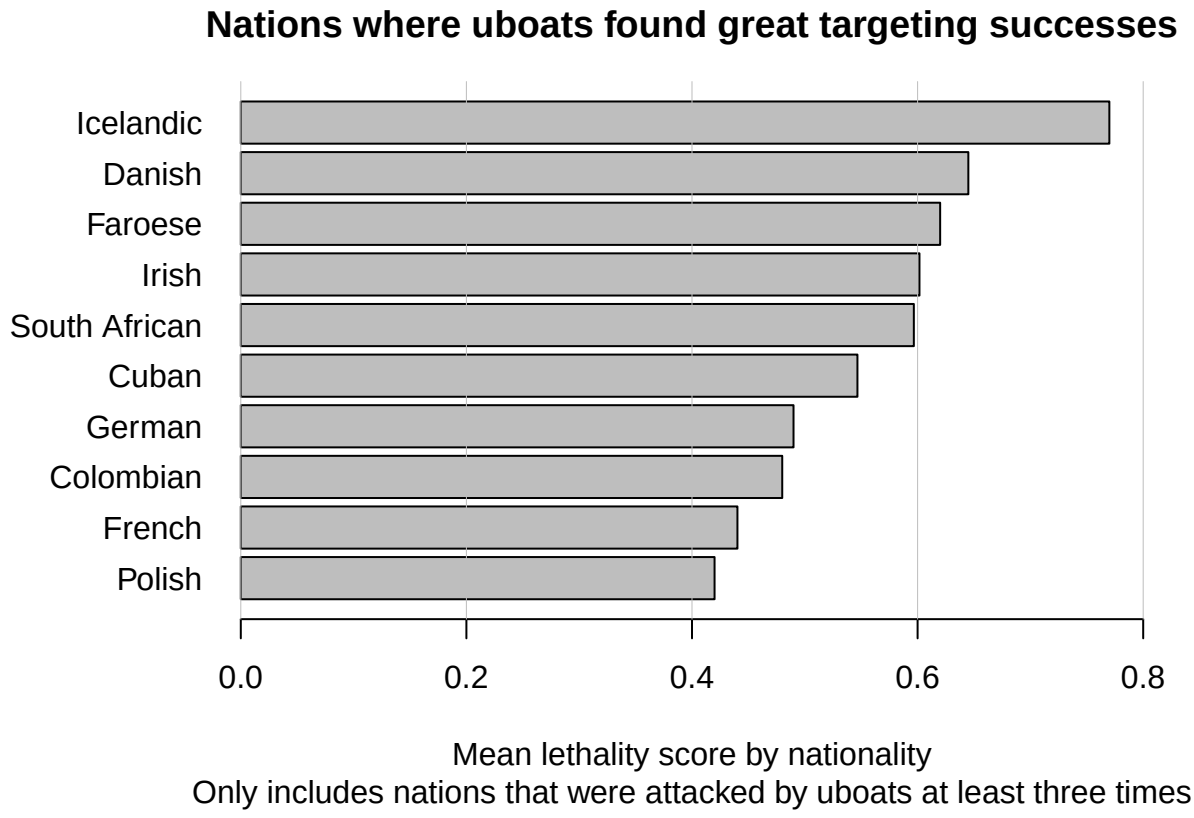
Analysing lethality in some more depth

We'll now quickly look at lethality in some other ways to understand the uboat threat in more detail.

Nationalities targeted

We'll start by attempting to understand what targets uboats had the most success against. We'll remove the attacks where a vessel from a nation was only struck by a uboat one time. For example, one Greendlandic vessel was attacked by a uboat and this one attack resulted in a lethality ratio of 1 as all personnel were killed. A one-off attack doesn't indicate any particular vulnerabilities to uboats so we'll only include measures where a nations vessel was attacked by a uboat at least three times.

```r
# create data of interest, which is when nation was attacked more than twice
three_attacks = uboat.target.df %>%
  count(nationality) %>%
  filter(n > 2)

# create actual data we want to plot
most_lethal_nation = uboat.target.df %>%
  filter(lethality < 1.1) %>%
  group_by(nationality) %>%
  summarize(lethal = mean(lethality))

# combine both, joining onto data of interest
three = three_attacks %>%
  left_join(most_lethal_nation, by = "nationality") %>%
  arrange(desc(lethal)) %>%
  head(10) %>%
  arrange(lethal)

# plot
par(mar = c(5.5,7,3,2))
barplot(three$lethal, names.arg = three$nationality, horiz = TRUE, las = 1, xlim = c(0,0.8),
        main= "Nations where uboats found great targeting successes",
        xlab = "Mean lethality score by nationality",
        sub = "Only includes nations that were attacked by uboats at least three times")
grid(NULL, 0, lty = 1, lwd = 0.5, col = 'grey')
```



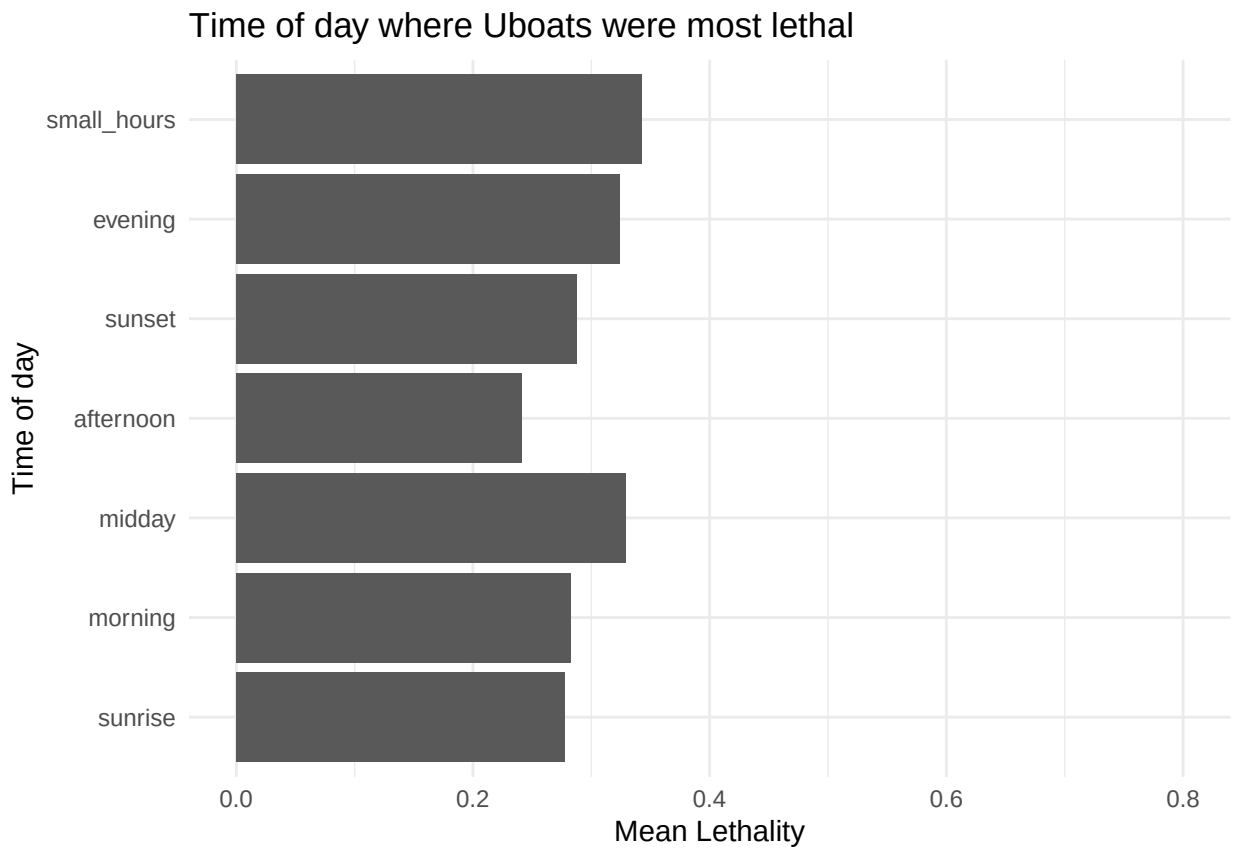
In the above plot, we can see that Icelandic vessels appeared especially vulnerable to uboat attacks as attacks tended to result in large proportions of their personnel being killed. It is unclear why this may be but the data would indicate additional research would be warranted. It is possible that the water temperature could be a factor, or remoteness. This is where perhaps some geo-spatial visualisations may provide additional insights.

Note also that German vessels tended to be vulnerable to uboat attacks... There were actually four occasions where uboats accidentally sunk other German vessels. We've already mentioned the Doggerbank being one of these. Only one of these attacks resulted in no personnel being killed, which was the Steinbek. Each case saw a different commander in charge. Research into these may identify opportunities that may have been exploitable.

Time of day and lethality

We'll conduct a similar analysis but seeing how the time of day of uboat attacks may impact their lethality.

```
uboat.target.df %>%
  group_by(day_period) %>%
  filter(lethality < 1.1 & day_period != "nil") %>%
  summarize(lets = mean(lethality)) %>%
  mutate(day_period = factor(day_period, levels = c("sunrise", "morning", "midday",
                             "afternoon", "sunset", "evening", "small_hours"))) %>%
  ggplot(aes(day_period, lets)) +
  coord_flip() +
  geom_bar(stat = 'identity') +
  ylim(0, 0.8) +
  labs(title = "Time of day where Uboats were most lethal",
       x = "Time of day",
       y = "Mean Lethality") +
  theme_minimal()
```



In the above plot we can see that the small hours of the day were when uboats were most lethal. We have already seen that the small hours were the preferred attack times of uboats generally and also by Wolfgang Luth. Interestingly, uboat attacks around midday tended to be the second most lethal time of attack for uboats. This is somewhat surprising as we didn't see much preference to attack at that time. This could possibly due to uboats only attacking at this more vulnerable time when they had a great targeting opportunity that was worth the risk. There could also be other reasons. This could justify conducting more research into understanding this.

Vulnerabilities of uboats

We've looked at the targeting and operations conducted by uboats. We'll now see if we can better understand their vulnerabilities.

Firstly, we'll have a look at what the final fates were of all uboats.

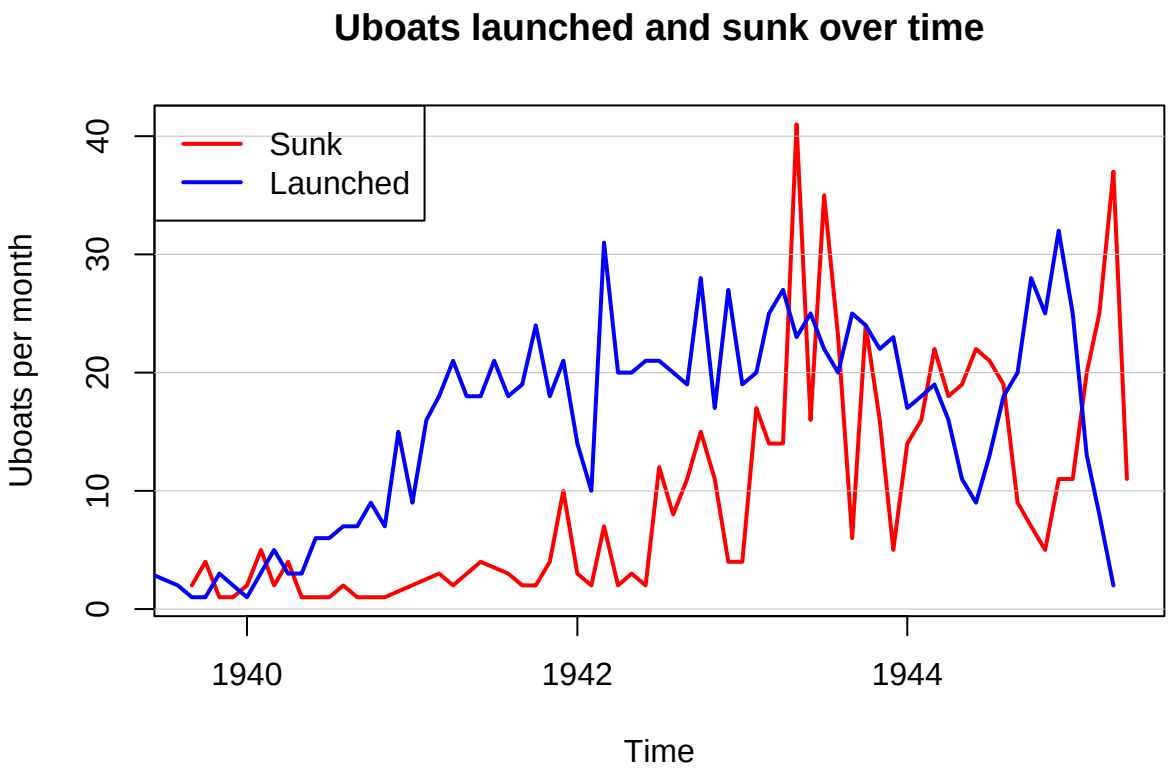
```
kable(uboat.df %>%
  count(fate_type) %>%
  arrange(desc(n)))
```

fate_type	n
Sunk	640
Scuttled	218
Surrendered	154
Decommissioned	55

fate_type	n
Missing	46
Given	16
Damaged	12
Captured	5
Grounded	4
Destroyed	3

From the above we can see that a larger proportion of them were sunk. This is perhaps a higher proportion than would be expected. We'll now compare the numbers of uboats being sunk with the number of uboats being launched into service. This may interesting.

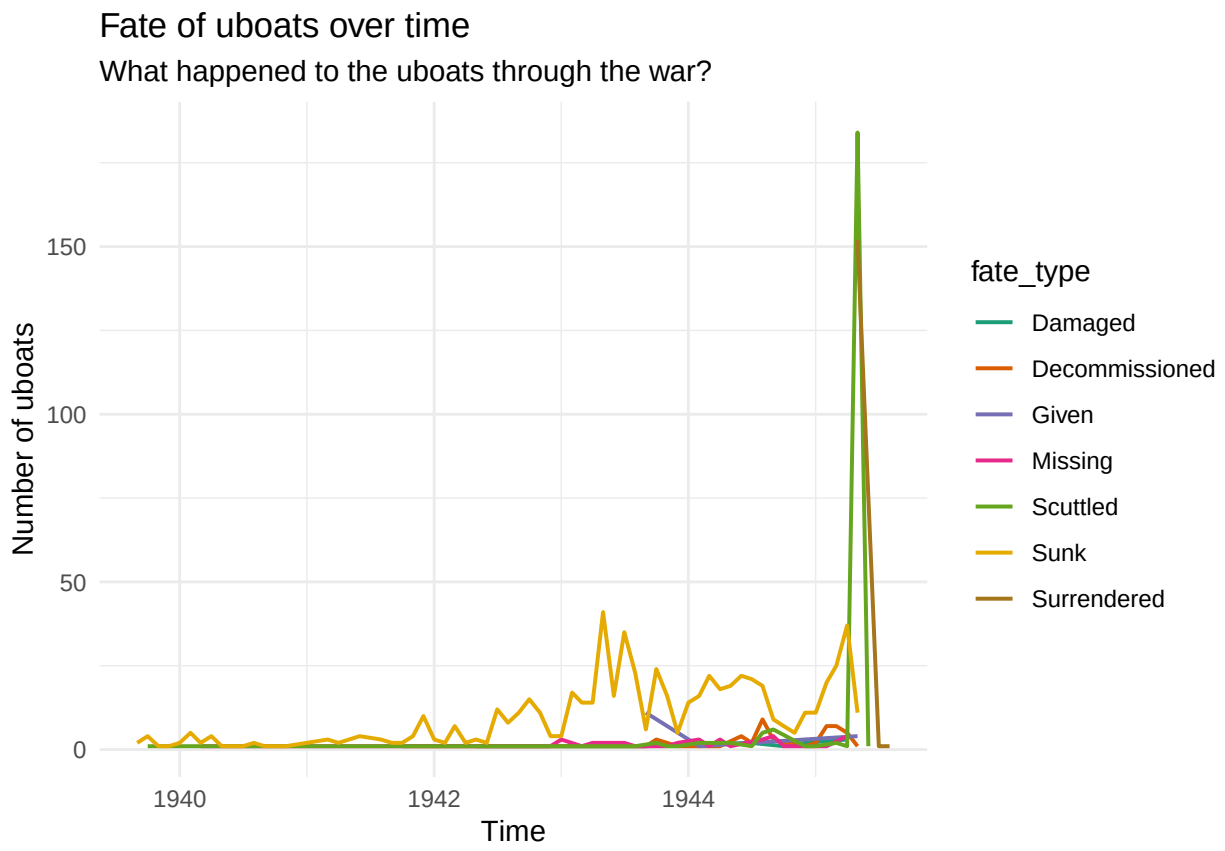
```
# uboats sunk
sunk_time = uboat.df %>%
  filter(fate_type == "Sunk") %>%
  group_by(month=floor_date(fate, "month")) %>%
  summarize(count = n())
  #summarize(uboats_sunk = count(fate_type == "Sunk"))

# uboats launched
launched_time = uboat.df %>%
  group_by(month=floor_date(launched, "month")) %>%
  summarize(count = n())

# plotting
plot(sunk_time$month, sunk_time$count, type = 'l', col = 'red', lwd = 2,
     main = "Uboats launched and sunk over time",
     xlab = "Time",
     ylab = "Uboats per month")
grid(0, NULL, col = 'grey', lty = 1, lwd = 0.5)
lines(launched_time$month, launched_time$count, type = 'l', col = 'blue', lwd = 2)
legend("topleft", c("Sunk", "Launched"), lty = 1, lwd = 2, col = c('red', 'blue'))
```



The above plot is quite interesting. Initially, the Germans were able to field more uboats than were being sunk. However from around mid 1943, it appears as though manufacturing peaked and losses of uboats exceeded the German ability to replace them. It appears as though that 1943 period would have really hit the uboat capability quite hard. Not only was there a sudden increase in the numbers of uboats being sunk but the numbers being launched not only plateaued but dropped quite significantly for some time, into the middle of 1945. There was some reversal of those trends but that wasn't long lasting.

Let's now look at how the fate of the uboats evolved over time. We'll omit the three smallest fate types which were captured (5), grounded (4) and destroyed (3) so we can keep the plot clean and easier to understand.

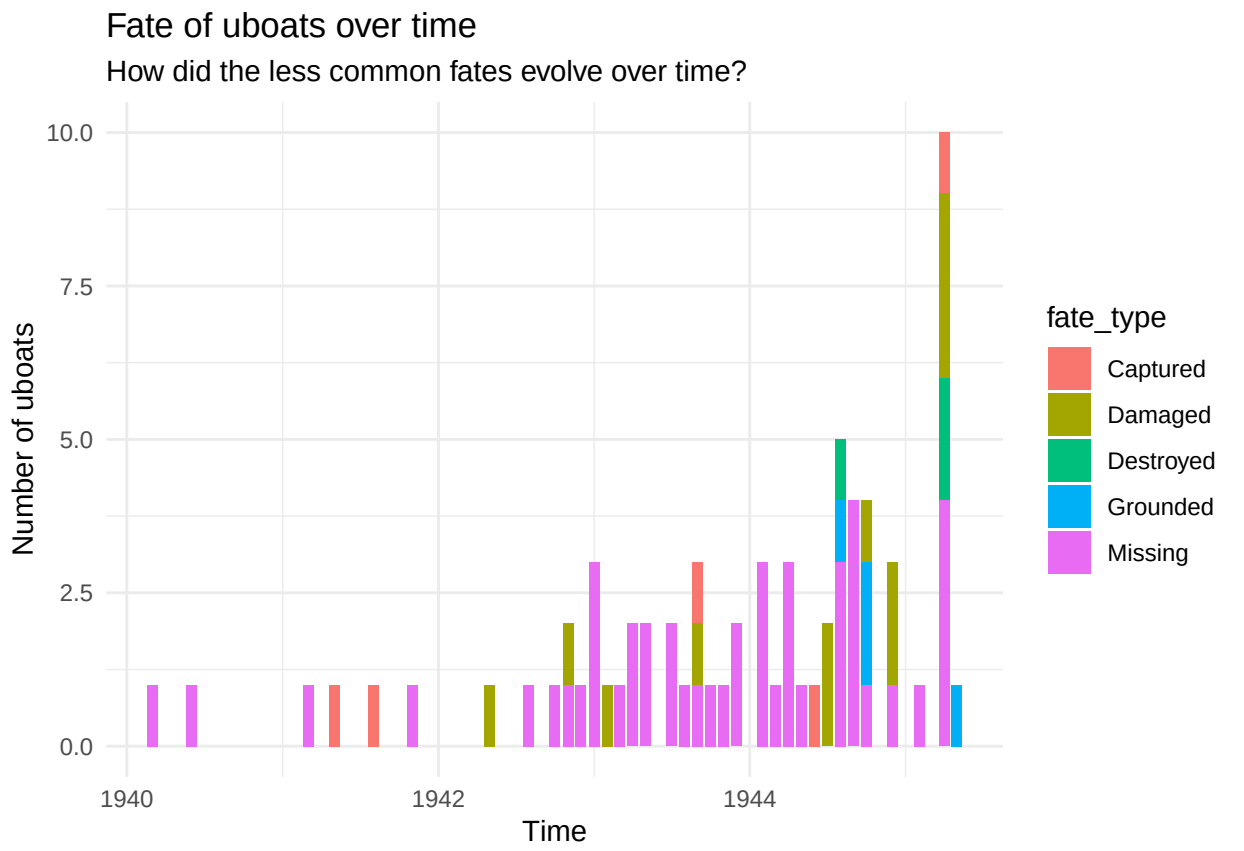
```
uboat.df %>%
  group_by(month=floor_date(fate, "month"), fate_type) %>%
  filter(fate_type != "Captured" & fate_type != "Grounded" & fate_type != "Destroyed") %>%
  summarize(num = n()) %>%
  ggplot(aes(month, num, col = fate_type))+
  geom_line(linewidth = 0.7) +
  labs(title = "Fate of uboats over time",
       subtitle = "What happened to the uboats through the war?",
       x = "Time",
       y = "Number of uboats")+
  scale_color_brewer(palette = "Dark2")+
  theme_minimal()
```




```
  #scale_colour_viridis_d()
```

So it seems our data has captured the final fate of Germany's uboats post-WWII as we see the large spikes for Scuttled and Surrendered. We can also see that numbers for Sunk dwarfed the other fate types so we might leave those fates here for now.

```
uboat.df %>%
  group_by(month=floor_date(fate, "month"), fate_type) %>%
  filter(fate_type == "Captured" | fate_type == "Grounded" | fate_type == "Destroyed"
         | fate_type == "Damaged" | fate_type == "Missing") %>%
  summarize(num = n()) %>%
  ggplot(aes(month, num, fill = fate_type))+
  geom_bar(stat = 'identity') +
  labs(title = "Fate of uboats over time",
       subtitle = "How did the less common fates evolve over time?",
       x = "Time",
       y = "Number of uboats") +
  theme_minimal()
```

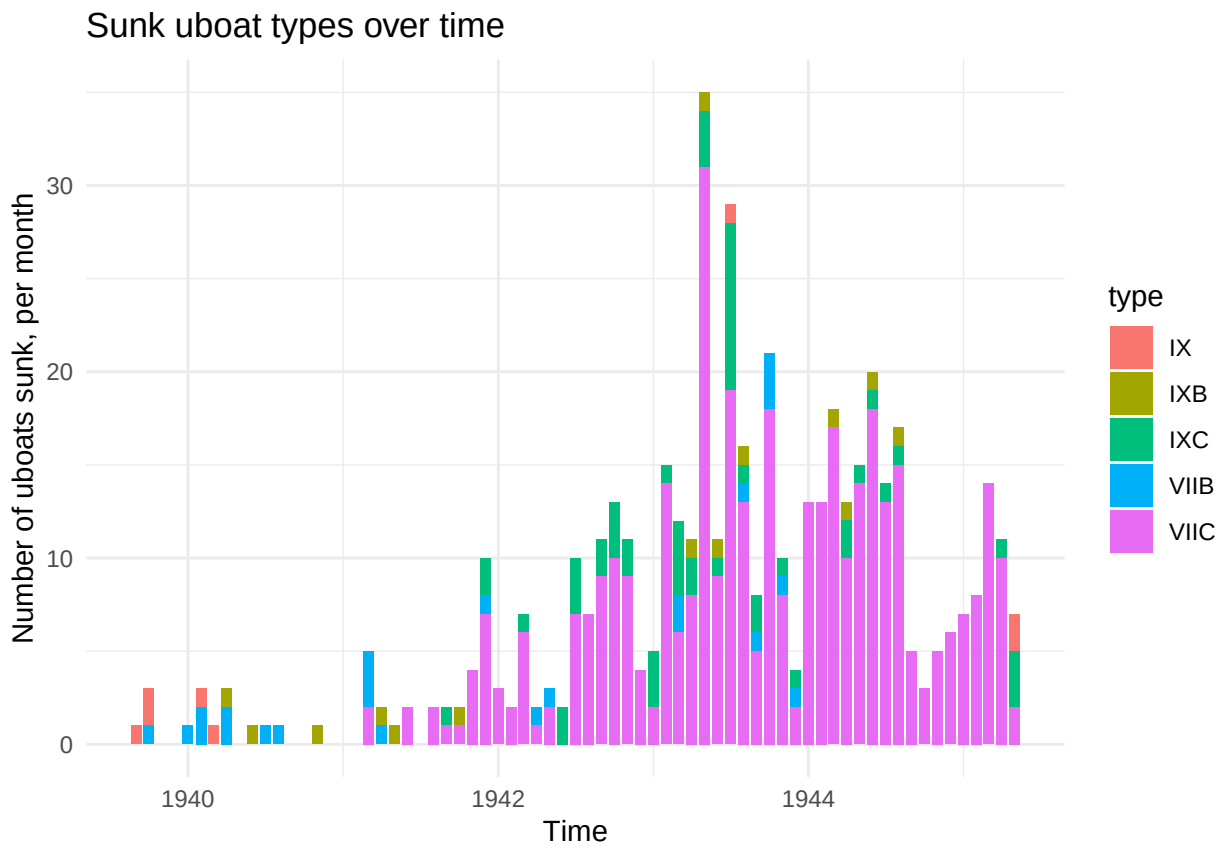



```
  #scale_color_brewer(palette = "Dark2")
```

We don't really gain any great insights from the above plot looking at less common fates. Note we've resorted to using bars instead of lines due to various entries with values of zero and a small number of categories. It is interesting that some uboats went missing and this did seem to increase from the end of 1942 but without more qualitative knowledge if this is important, there's not much else to do for now. Without much knowledge, we'd expect that many of those uboats that were missing went missing during attacks and were actually sunk. But this may be more interesting than it appears and could warrant additional research to better understand.

We'll see if there are any interesting insights to be gleaned by looking at uboats sunk over time by type. We'll select just the top five most dangerous types we identified earlier.

```
uboat.df %>%
  group_by(month=floor_date(fate, "month"), type) %>%
  filter(fate_type == "Sunk" & type == "VIIC" | type == "IXC" | type == "IXB" | type == "VIIB"
         | type == "IX") %>%
  summarize(num = n()) %>%
  ggplot(aes(month, num, fill = type))+
  geom_bar(stat = 'identity') +
  #geom_line(linewidth = 0.7) +
  labs(title = "Sunk uboat types over time",
       x = "Time",
       y = "Number of uboats sunk, per month")+
  #scale_color_brewer(palette = "Dark2")+
  theme_minimal()
```



We don't really see anything of particular interest here. We know the VIIC was the main attack workhorse and we know the number of attacks they conducted was high so we expect their numbers to be represented here. An avenue potentially worth exploring would be to compare what we know about the overall proportions of Uboat types (from earlier in this analysis) and compare those with the data in the above plot. But we won't do that here.

Understanding the most efficient uboats and crews

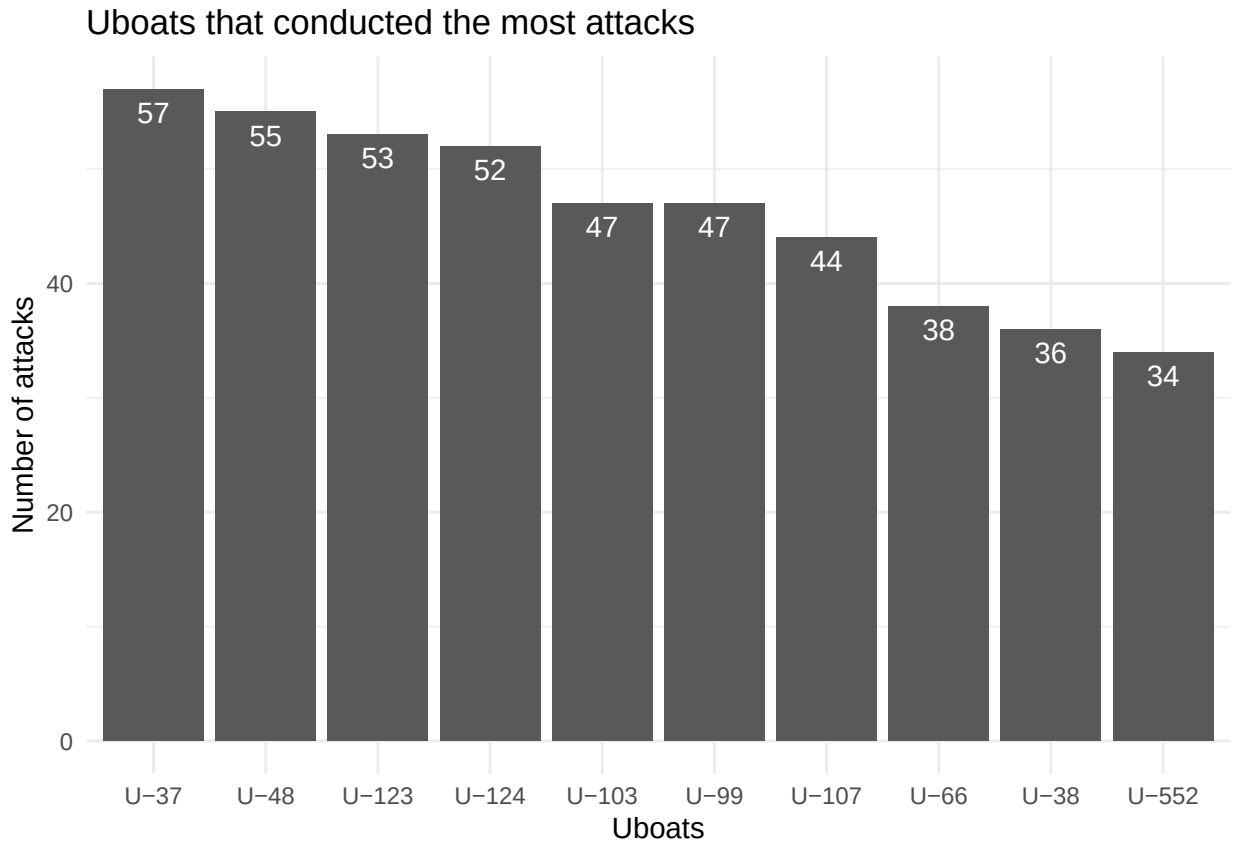
We'll now look at the most deadly uboats. These may or may not be different to insights we gleaned analysing the uboat commanders. Perhaps the effectiveness of some uboats was at least partly due to competent high performing crews. A good commander could make the most use of good crews. A good commander could be constrained by poorer performing crews. Some commanders commanded multiple uboats, they didn't only work on one uboat throughout the war.

Number of attacks

We'll start off by looking at the uboats that conducted the most attacks.

```
uboat.target.df %>%
  group_by(name) %>%
  summarise(most_attacks = n()) %>%
  arrange(desc(most_attacks)) %>%
  head(10) %>%
  ggplot(aes(reorder(name, -most_attacks), most_attacks))+
  geom_bar(stat="identity")+
```

```
  geom_text(aes(label = most_attacks), vjust=1.6, color = 'white')+
  labs(title = "Uboats that conducted the most attacks",
       x = "Uboats",
       y = "Number of attacks")+
  theme_minimal()
```



We'll now look to see who captained the top three uboats at any point and see if the same commanders who have featured previously in our analysis turn up here.

```
kable(uboat.target.df %>%
  filter(name == "U-37" | name == "U-48" | name == "U-123") %>%
  distinct(commander))
```

commander
Victor Oehrn
Werner Hartmann
Asmus Nicolai Clausen
Heinrich Bleichrodt
Herbert Schultze
Hans Rudolf Rosing
Karl-Heinz Moehle
Reinhard Hardegen
Horst von Schroeter

Here are the top ten commanders we previously identified for easy comparison.

```
kable(uboat.target.df %>%
  count(commander) %>%
  arrange(desc(n)) %>%
  head(10))
```

commander	n
Otto Kretschmer	56
Wolfgang Luth	49
Joachim Schepke	40
Erich Topp	39
Heinrich Liebe	35
Viktor Schutze	35
Georg Lassen	33
Gunther Prien	33
Johann Mohr	32
Ernst Bauer	29

Interestingly, we don't see Otto Kretschmer, Wolfgang Luth or Joachim Schepke (the top three Commanders who conducted the most attacks). Actually, none of the top 10 commanders who conducted the most attacks commanded any of the top three uboats. The actual number of attacks conducted by the top ten uboats is roughly the same as the top ten commanders. This is interesting.

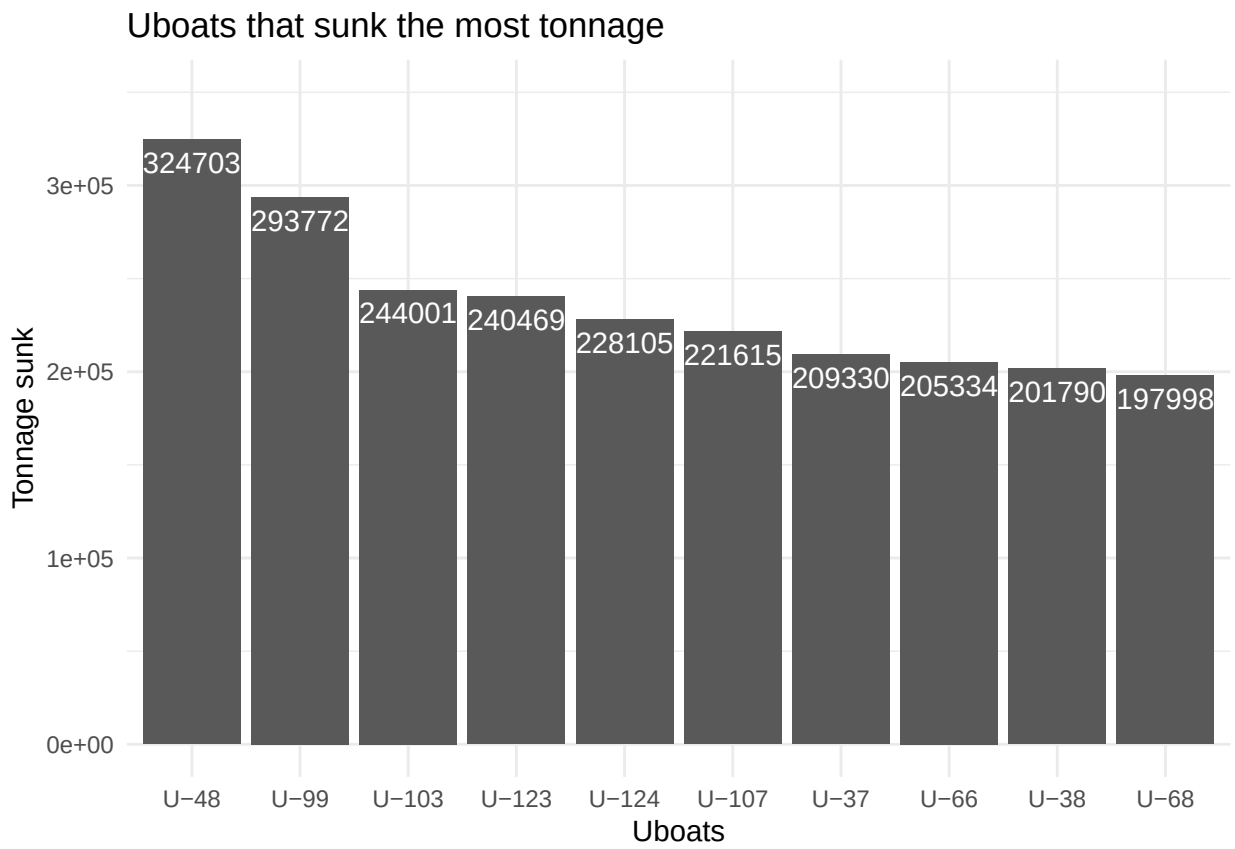
We could ask the question now, what would constitute a higher threat, a uboat (featuring commander and crew in combination) or a particular commander? What should we focus more on?

Let's analyse the uboats further to see if we can learn more.

Sum of tonnage sunk by uboats

We'll now sum the tonnage sunk by each uboat and identify the top ten.

```
uboat.target.df %>%
  filter(loss_type == "Sunk") %>%
  group_by(name) %>%
  summarise(tonage = sum(tonnage)) %>%
  arrange(desc(tonage)) %>%
  head(10) %>%
  ggplot(aes(reorder(name, -tonage), tonage))+
  geom_bar(stat="identity")+
  ylim(0,350000)+
  geom_text(aes(label = tonage), vjust=1.6, color = 'white')+
  labs(title = "Uboats that sunk the most tonnage",
       x = "Uboats",
       y = "Tonnage sunk")+
  theme_minimal()
```



We again see uboat U-48 featuring in the top three. Let's again look at who commanded the top three uboats which sunk the most tonnage.

```
kable(uboat.target.df %>%
  filter(name == "U-99" | name == "U-48" | name == "U-103") %>%
  distinct(commander))
```

commander
Otto Kretschmer
Heinrich Bleichrodt
Herbert Schultze
Hans Rudolf Rosing
Viktor Schutze
Werner Winter
Gustav-Adolf Janssen

And here are the commanders who sunk the most tonnage for easy comparison.

```
kable(uboat.target.df %>%
  group_by(commander) %>%
  filter(loss_type == "Sunk") %>%
  summarize(tonnagez = sum(tonnage)) %>%
  arrange(desc(tonnagez)) %>%
  head(10))
```

commander	tonnagez
Otto Kretschmer	305883
Wolfgang Luth	225756
Erich Topp	203786
Heinrich Liebe	200090
Gunther Prien	191919
Heinrich Lehmann-Willenbrock	180753
Herbert Schultze	180455
Viktor Schutze	180073
Karl-Friedrich Merten	170151
Werner Henke	166055

And we do see some crossover here. Otto Kretschmer did indeed command U-99 (and U-23). We also see Herbert Schultze (commanded only U-48) and Viktor Schutze (commanded U-25 and U-103) featuring in both lists here for most tonnage sunk. Note that the tonnage sunk figures are slightly higher for the uboats than for the commanders.

Commanders or crew?

At this point, it remains unclear how much credit we should place on the commanders and/or the crews of the uboats regarding their effectiveness in combat. We note that some of the highest rated commanders did command more than one uboat. We should then see that if the commander has a larger affect on the effectiveness of the uboat attacks, then each uboat these commanders command should all feature similar statistics. If they vary significantly, we should perhaps place more emphasis on the competency of the crew. To do this fairly, we need to be aware that some commanders may have spent more time commanding one uboat compared to another so they won't have had time to rack up the numbers in the boats they haven't spent as much time in.

We have developed a lethality ratio where that metric isn't really affected too much by time. However we'll aim to do something similar but for tonnage sunk over time. We'll then aim to compare this sort of ratio between the top uboat (U-43) then the top commander for tonnage sunk (Otto Kretschmer) and compare his performance for both U-99 and U-23.

Efficiency ratio

For the uboats themselves, we already have metrics for the time they launched to their fate and we've already calculated the total time they were operational. We could calculate a ratio for uboats. To keep things simple, we'll use the following expression.

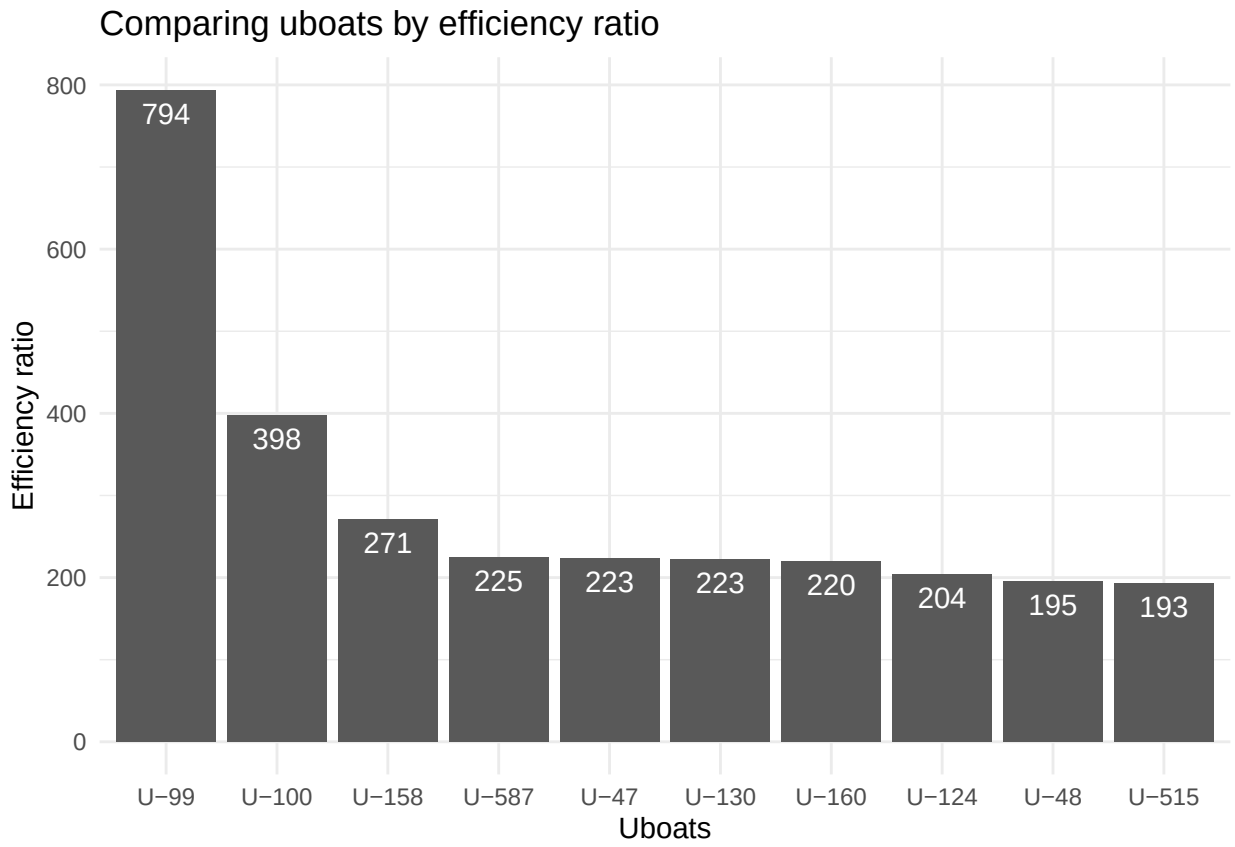
efficiency ratio = tonnage sunk / time operational (in days)

This will likely provide more meaningful metrics than purely a sum. Uboats that were able to avoid being sunk or lost in some way will likely continue to increase their number of attacks. However some of these uboats may not be particularly effective, but they endure. On the one hand, their survival and their ability to sink large tonnages over time may actually be what makes them more lethal over time. However this new metric that considers time may identify uboats that were highly effective but came to a perhaps earlier demise.

We'll now aim to identify the most efficient uboats at sinking larger tonnages.

```
efficiency_ratio = uboat.target.df %>%
  filter(loss_type == "Sunk") %>%
  group_by(name) %>%
```

```
  summarise(tonnages = sum(tonnage)) %>%
  inner_join(uboat.df, by = "name") %>%
  mutate_at('total_time', as.numeric) %>%  # converting to numeric so we can calculate
  mutate(efficiency = tonnages / total_time) %>%
  select(name, efficiency)

efficiency_ratio %>%
  arrange(desc(efficiency)) %>%
  head(10) %>%
  ggplot(aes(reorder(name, -efficiency), efficiency))+
  geom_bar(stat = 'identity')+
  geom_text(aes(label = round(efficiency)), vjust=1.6, color = 'white')+
  labs(title = "Comparing uboats by efficiency ratio",
       x = "Uboats",
       y = "Efficiency ratio")+
  theme_minimal()
```



We can clearly see here that U-99 was significantly more efficient at sinking large tonnages in a relatively shorter amount of time than other uboats, dwarfing the rest of the top ten. Let's see how many commanders U-99 had, noting we already know that Otto Kretschmer commanded it for part of his career.

```
kable(uboat.target.df %>%
  filter(name == "U-99") %>%
  distinct(commander))
```

commander
Otto Kretschmer

And we can see that only Otto Kretschmer commanded this uboat. This is interesting. We also know that Otto Kretschmer commanded U-23 so let's see how efficient that uboat was.

```
kable(efficiency_ratio %>%
  filter(name == "U-23"))
```

name	efficiency
U-23	4.289268

We can see that U-23 only had an efficiency ratio of just over 4, which is far lower than the other uboat Otto Kretschmer commanded. Let's look at who else commanded U-23, how long it was operational and how long Otto Kretschmer commanded to ensure we are getting a fair comparison.

```
kable(uboat.df %>%
  filter(name == "U-23") %>%
  select(name, total_time))
```

name	total_time
U-23	2935 days

We can see that U-23 was operational for some time.

```
kable(uboat.target.df %>%
  filter(name == "U-23") %>%
  distinct(commander))
```

commander
Otto Kretschmer
Rolf-Birger Wahlen
Rudolf Arendt

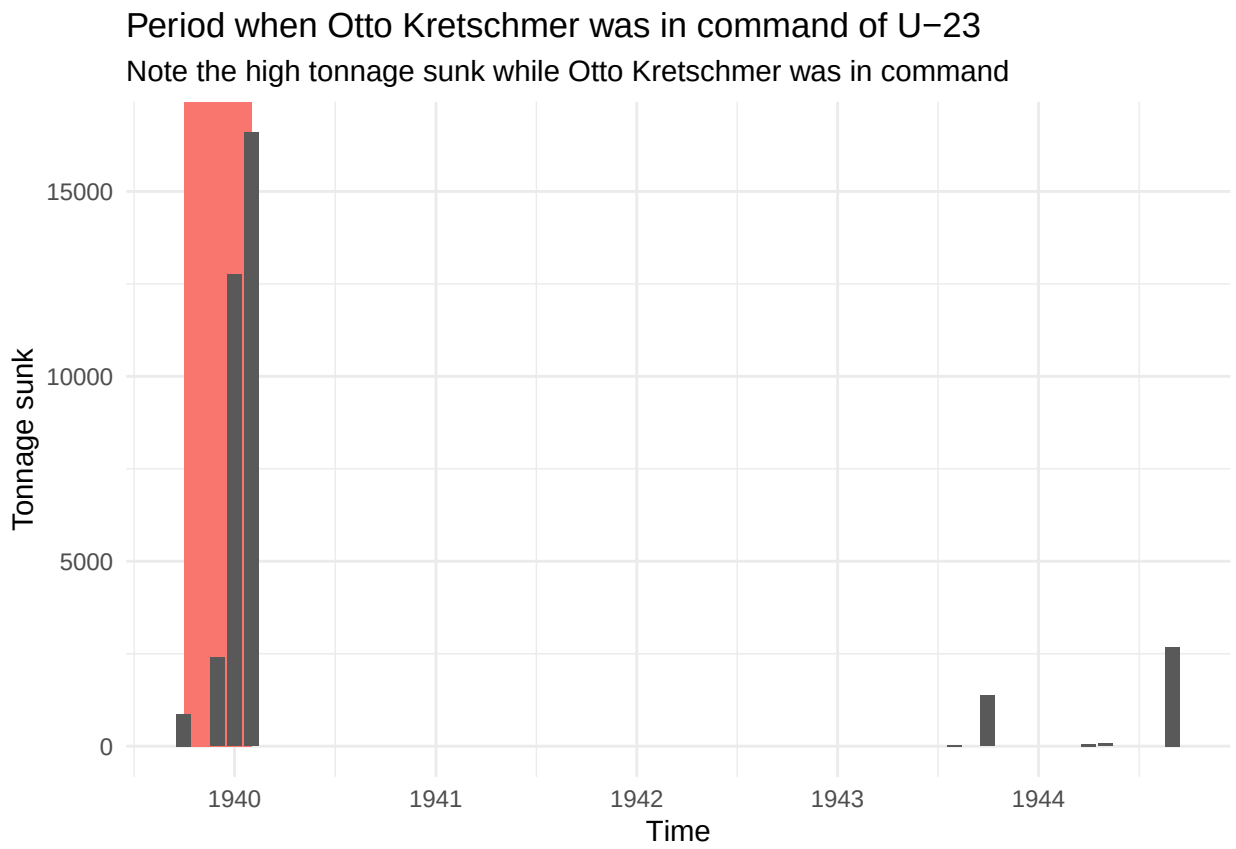
We can see that U-23 was commanded by three different commanders at different times. Let's see if we can determine the time period that Otto Kretschmer commanded U-23. This time period will not accurately represent how long a commander commanded a uboat. It will only capture the times of their first and last attack in any given uboat.

```
kable(uboat.target.df %>%
  filter(name == "U-23" & commander == "Otto Kretschmer") %>%
  mutate(early = min(attack_date)) %>%
  mutate(latest = max(attack_date)) %>%
  mutate(period = latest - early) %>%
  select(name, commander, period) %>%
  head(1))
```

name	commander	period
U-23	Otto Kretschmer	141 days

So we can see that Otto Kretschmer only commanded U-23 for 141 out of 2935 days. So in this case, it isn't a really fair comparison. This means we cannot make any strong conclusions as to the importance of a commander versus a crew at the moment. Let's see if we can see any changes to the operational effectiveness of U-23 during the time period when Otto Kretschmer was in command. We'll look at the tonnage sunk by U-23 over time. Because Otto Kretschmer was most effective at conducting attacks and sinking the most tonnage, we may see some effect if this was due to this command versus the capability of the crew of U-99.

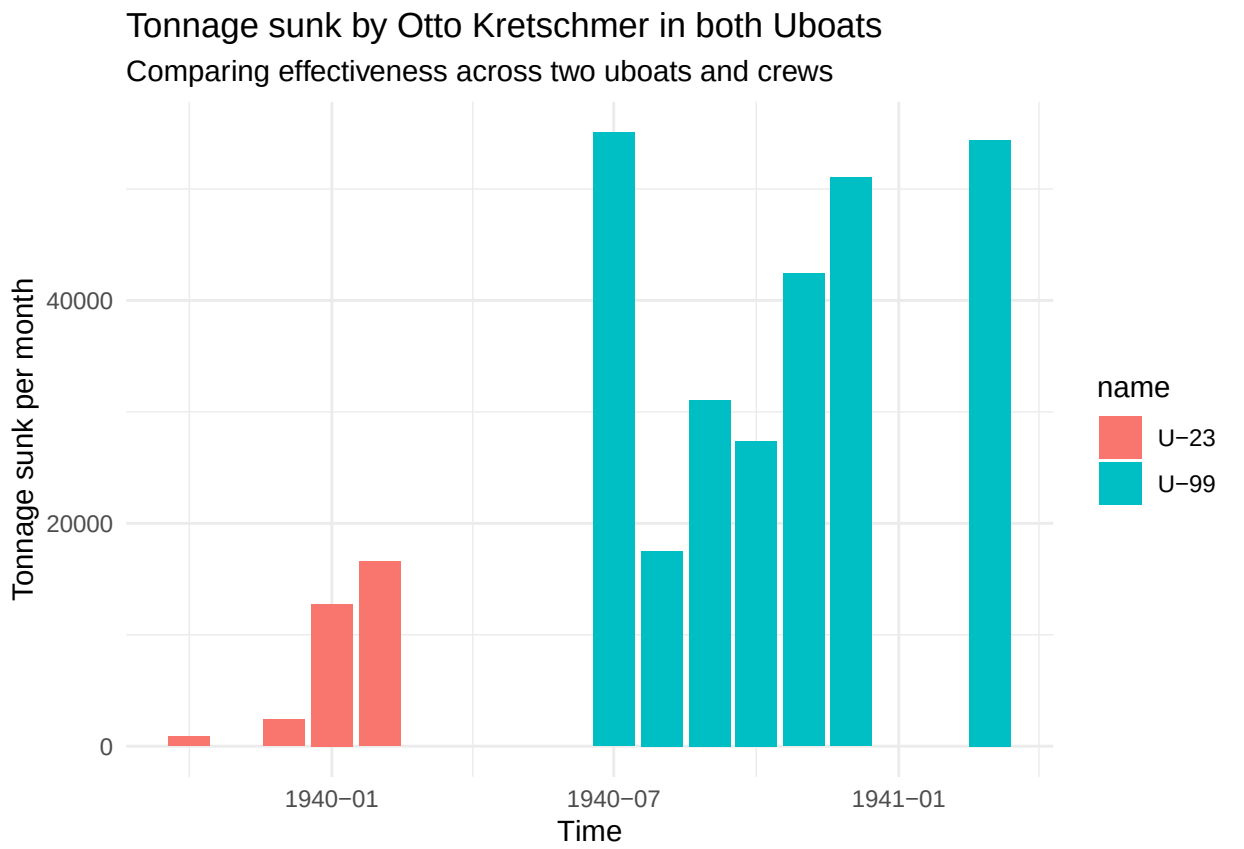
```
# we had to use month[1] instead of the saved min_otto here
min_otto = uboat.target.df %>%
  filter(name == "U-23" & commander == "Otto Kretschmer") %>%
  filter(attack_date == min(attack_date)) %>%
  select(attack_date)
# we had to use month[4] instead of the saved max_otto here
max_otto = uboat.target.df %>%
  filter(name == "U-23" & commander == "Otto Kretschmer") %>%
  filter(attack_date == max(attack_date)) %>%
  select(attack_date)

# note we had to manually select the time period for the rect, matching with above.
uboat.target.df %>%
  filter(name == "U-23") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  summarize(tonz = sum(tonnage)) %>%
  ggplot(aes(month, tonz)) +
  labs(title = "Period when Otto Kretschmer was in command of U-23",
       subtitle = "Note the high tonnage sunk while Otto Kretschmer was in command",
       x = "Time",
       y = "Tonnage sunk")+
  geom_rect(aes(xmin=month[1], xmax=month[4], ymin=0, ymax=Inf, fill='red'))+
  geom_bar(stat = 'identity') +
  #geom_line(linewidth = 0.7)+
  theme_minimal()+
  theme(legend.position = "none")
```




```
  #geom_vline(aes(xintercept = month[1], color = 'red'))+
  #geom_vline(aes(xintercept = month[5], color = 'red'))
```

The previous analysis leading to the above plot seems to indicate that the command of Otto Kretschmer was indeed influential in the effectiveness of U-23. While this only represents one case, for at least some commanders, their impact on the overall effectiveness of uboat operations appears significant. For now, we don't have a good understanding of other factors that may have impacted the operational effectiveness of U-23 from the mid-1943 and onwards period. It's also worth noting that the main reason that U-23 has a very low efficiency ratio is likely due to a large period of apparent inactivity.

```
uboat.target.df %>%
  filter(commander == "Otto Kretschmer") %>%
  group_by(month=floor_date(dateandtime, "month"), name) %>%
  summarize(ttonz = sum(tonnage)) %>%
  ggplot(aes(month, ttonz, fill=name))+
  labs(title = "Tonnage sunk by Otto Kretschmer in both Uboats",
       subtitle = "Comparing effectiveness across two uboats and crews",
       x = "Time",
       y = "Tonnage sunk per month")+
  #geom_line(linewidth = 0.7)+
  geom_bar(stat = 'identity') +
  theme_minimal()
```



The above plot shows us that Otto Kretschmer did sink far more tonnage per month in U-99 than in U-23. It is difficult to make too strong of a conclusion from this for now. His command of U-23 was earlier in his career so he could have been 'learning'. His area of operations could have been different and have different levels of target-richness.

The above plot may also indicate that it was the crew of U-99 that was far more effective when compared to the crew of U-23, and the success of U-99 may have been at least partially due to the crew rather than just the command of Otto Kretschmer. We'd really need to conduct more qualitative analysis to increase our confidence in any conclusion or inference we make here.

While we can't make any strong conclusions right now, we have learnt a little more at least about the command of Otto Kretschmer and the operations of U-99. It's noteworthy how much tonnage U-99 with Otto Kretschmer sunk considering they were only operational for a relatively short period of time, totally a little over a year.

Let's see what fate awaited U-99.

```
kable(uboat.df %>%
  filter(name == "U-99") %>%
  select(name, total_time, ships_sunk, fate, fate_type, fate_survivors, fate_dead))
```

name	total_time	ships_sunk	fate	fate_type	fate_survivors	fate_dead
U-99	370 days	35	1941-03-17	Scuttled	40	3

U-99 was badly damaged by a destroyer soon after firing her last torpedoes sinking *Korshamn*. Due to the damage, U-99 needed to surface and was soon captured, the crew scuttled the boat and the surviving crew were captured. More can be found at the wikipedia page for U-99.

Regarding the efficiency ratios for the uboats, we'll quickly see who commanded the second most efficient uboat from the above barplot, which was U-100.

```
uboat.target.df %>%
  filter(name == "U-100") %>%
  distinct(commander)
```

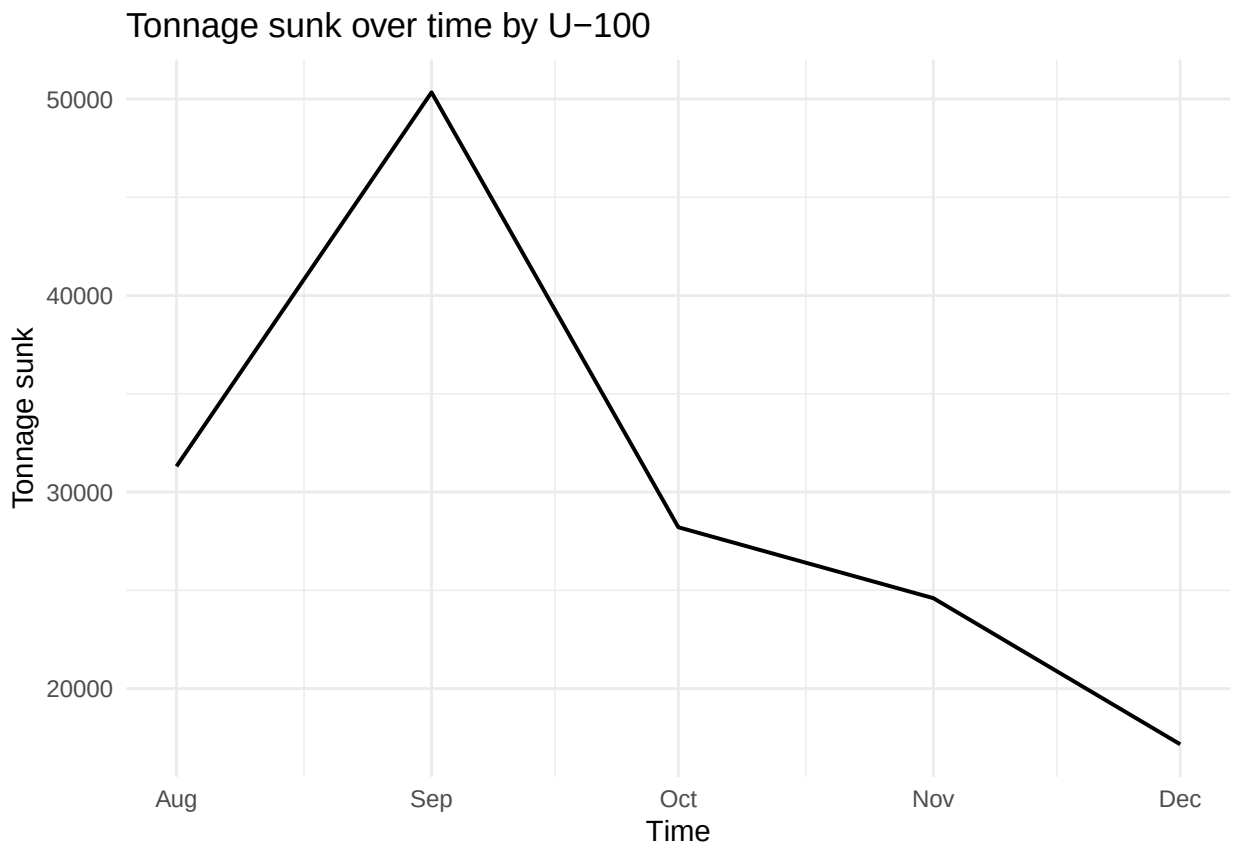
```
##         commander
## 1 Joachim Schepke
```

Joachim Schepke was the only commander for U-100 and was the third highest rated commander when it came to total tonnage sunk. So not only did he sink large amounts of tonnage, he did so in a time efficient manner.

What is perhaps of note, is that the two uboats with the highest efficiency ratios were both commanded by just one commander. This may indicate there were less changes to processes onboard and that the commanders had time to build up a solid working relationship with the crew and develop well-drilled processes. It could also just be a coincidence at this point. This could be an entirely different avenue of analysis to drill into to explore further. But we won't for now.

Let's look at the tonnage sunk by U-100 over time.

```
uboat.target.df %>%
  filter(name == "U-100") %>%
  group_by(month=floor_date(dateandtime, "month")) %>%
  summarize(tonz = sum(tonnage)) %>%
  ggplot(aes(month, tonz)) +
  labs(title = "Tonnage sunk over time by U-100",
       x = "Time",
       y = "Tonnage sunk")+
  geom_line(linewidth = 0.7)+
  #geom_bar(stat = 'identity')+
  theme_minimal()
```



What's most interesting here, is that U-100 at the command of Joachim Schepke rated so highly when it comes to tonnage sunk and this was all achieved over just a five month period. Let's see what happened to U-100.

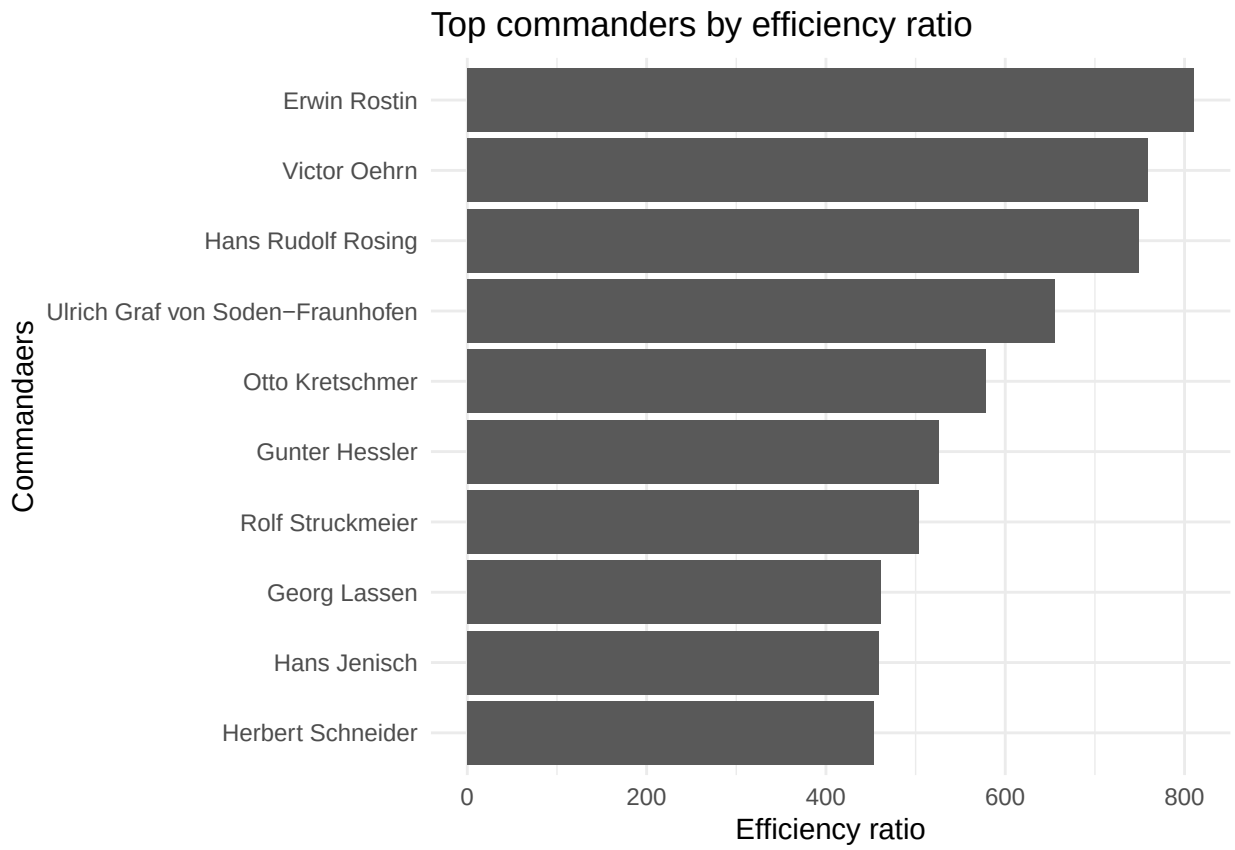
```
kable(uboat.df %>%
  filter(name == "U-100") %>%
  select(name, total_time, ships_sunk, fate, fate_type, fate_survivors, fate_dead))
```

name	total_time	ships_sunk	fate	fate_type	fate_survivors	fate_dead
U-100	341 days	25	1941-03-17	Sunk	6	38

So we can see that the efficiency ratio seems to be identifying some particularly interesting uboats. We'll now attempt to build similar metrics for the commanders.

```
comd_period = uboat.target.df %>%
  group_by(commander) %>%
  mutate(early = min(attack_date)) %>%
  mutate(latest = max(attack_date)) %>%
  mutate(period = latest - early) %>%
  filter(period > 60) %>%
  select(commander, period) %>%
  filter(period > 1) %>% # removes rows where period is 0 due to data issues
  distinct()
```

```
ten_efficient_comd = uboat.target.df %>%
  filter(loss_type == "Sunk") %>%
  group_by(commander) %>%
  summarise(tonnnages = sum(tonnage)) %>%
  inner_join(comd_period, by = "commander") %>%
  mutate_at('period', as.numeric) %>%
  mutate(efficiency = tonnnages / period) %>%
  select(commander, tonnnages, period, efficiency) %>%
  arrange(desc(efficiency)) %>%
  head(10)

ten_efficient_comd %>%
  ggplot(aes(reorder(commander, efficiency), efficiency))+
  geom_bar(stat = 'identity')+
  coord_flip()+
  labs(title = "Top commanders by efficiency ratio",
       x = "Commandaers",
       y = "Efficiency ratio")+
  theme_minimal()
```



We've added a filter where commanders needed to be active for at least 60 days. Without that, we got ten commanders who conducted attacks all within less than 8 days where they sunk around two to four vessels or so in quick succession. That's not really the sort of efficiency we are interested in exploring. We want to understand efficiency ratings where that efficiency lasts for much longer than one week.

Here we see some famililar names and some new names. Let's have a look at some of their key descriptive

statistics.

```
kable(ten_efficient_comd)
```

commander	tonnnages	period	efficiency
Erwin Rostin	101321	125	810.5680
Victor Oehrn	111648	147	759.5102
Hans Rudolf Rosing	60701	81	749.3951
Ulrich Graf von Soden-Fraunhofen	57694	88	655.6136
Otto Kretschmer	305883	529	578.2287
Gunter Hessler	122651	233	526.3991
Rolf Struckmeier	75020	149	503.4899
Georg Lassen	161068	349	461.5129
Hans Jenisch	110139	240	458.9125
Herbert Schneider	51258	113	453.6106

One thing that stands out here is that Otto Kretschner was able to build and maintain a high efficiency rating over the longest period of time compared to the other commanders in the top ten here.

Let's just take a closer look at the commander with the highest efficiency score, Erwin Rostin.

```
kable(uboat.target.df %>%
  filter(commander == "Erwin Rostin") %>%
  select(name, dateandtime, nationality, tonnage, ship_name, lethality, day_period))
```

name	dateandtime	nationality	tonnage	ship_name	lethality	day_period
U-158	1942-02-24 08:55:00	British	8032	Empire Celt	0.11	morning
U-158	1942-02-24 10:35:00	British	8146	Diloma	0.00	morning
U-158	1942-03-01 02:34:00	Norwegian	9551	Finnanger	1.00	small_hours
U-158	1942-03-11 07:58:00	American	2609	Caribsea	0.75	sunrise
U-158	1942-03-13 05:05:00	American	11641	John D. Gill	0.47	small_hours
U-158	1942-03-15 06:04:00	American	7118	Olean	0.14	sunrise
U-158	1942-03-15 07:22:00	American	6952	Ario	0.24	sunrise
U-158	1942-05-20 22:00:00	British	8113	Darina	0.11	evening
U-158	NA	Canadian	1748	Frank B. Baird	0.00	nil
U-158	1942-06-02 02:57:00	American	5686	Knoxville City	0.04	small_hours
U-158	1942-06-04 04:00:00	Norwegian	2647	Nidarnes	0.54	small_hours
U-158	1942-06-05 03:32:00	American	2572	Velma Lykes	0.47	small_hours

name	dateandtime	nationality	tonnage	ship_name	lethality	day_period
U-158	NA	Panamanian	5234	Hermis	0.02	nil
U-158	1942-06-11 10:55:00	Panamanian	13467	Sheherazade	0.02	morning
U-158	1942-06-12 07:50:00	American	8192	Cities Service Toledo	0.33	sunrise
U-158	1942-06-17 04:50:00	Panamanian	3601	San Blas	0.68	small_hours
U-158	1942-06-17 13:00:00	Norwegian	1560	Moira	0.05	afternoon
U-158	1942-06-23 08:40:00	American	5766	Major General Henry Gibbins	0.00	morning
U-158	1942-06-29 17:45:00	Latvian	3950	Everalda	0.00	sunset

Erwin Rostin only ever commanded one uboat, and that was U-158. U-158, which was also only ever commanded by Erwin Rostin, also featured the third highest efficiency ratio for uboats.

```
uboat.target.df %>%
  filter(name == "U-158") %>%
  distinct(commander)
```

```
##      commander
## 1 Erwin Rostin
```

Modelling

Predicting HVT involvement in an attack

We'll quickly walk through a very basic demonstration of a model to see if we can predict if a high value commander conducted an attack based on time of day, numbers killed, tonnage sunk, nationality of target and what weekday.

We'll demonstrate a simple model using a generalised linear model for this. We'll first need to create a binary variable where we manually list the commanders of most note resulting from the analysis we've conducted. We'll place these 0s and 1s in a variable / column called `hvt`. `hvt` will be our *response* variable. Because we will be using a variable with a value of either 0 or 1, this will be a *binomial* value.

```
uboat.target.df = uboat.target.df %>%
  mutate(hvt = case_when(
    commander == "Joachim Schepke" ~ 1,
    commander == "Werner Heidel" ~ 1,
    commander == "Wolfgang Luth" ~ 1,
    commander == "Erwin Rostin" ~ 1,
    commander == "Victor Oehrn" ~ 1,
    commander == "Otto Kretschmer" ~ 1,
    commander == "Heinrich Bleichrodt" ~ 1,
    commander == "Herbert Schultze" ~ 1,
    commander == "Viktor Schutze" ~ 1,
```

```
    commander == "Werner Winter" ~ 1,
    commander == "Gustav-Adolf Janssen" ~ 1,
    commander == "Heinrich Liebe" ~ 1,
    commander == "Gunther Prien" ~ 1,
    commander == "Heinrich Lehmann-Willenbrock" ~ 1,
    commander == "Karl-Friedrich Merten" ~ 1,
    commander == "Werner Henke" ~ 1,
    commander == "Hans-Diedrich Freiherr von Tiesenhausen" ~ 1,
    commander == "Robert Gysae" ~ 1,
    commander == "Johann Mohr" ~ 1,
    commander == "Adolf Cornelius Piening" ~ 1,
    commander == "Gerhard Meyer" ~ 1,
    commander == "Hans Heidtmann" ~ 1,
    commander == "Ulrich Graf von Soden-Fraunhofen" ~ 1,
    commander == "Rolf Struckmeier" ~ 1,
    commander == "Georg Lassen" ~ 1,
    commander == "Hans Jenisch" ~ 1,
    commander == "Herbert Schneider" ~ 1,
    TRUE ~ 0),
             .after="commander")
```

Now we'll build a model where we aim to predict if an attack was conducted by a high value commander or not. Here is what our proposed model will look like, drawing on what we think seems logical from our exploratory analysis conducted so far.

hvt ~ day_period + weekday + dead + tonnage + nationality + complement

For demonstration purposes, we'll build a binomial generalised linear model and check the model summary.

```
comd.lm = glm(hvt ~ day_period + weekday + dead + tonnage + nationality + complement,
              data = uboat.target.df, family = 'binomial')
summary(comd.lm)
```

```
##
## Call:
## glm(formula = hvt ~ day_period + weekday + dead + tonnage + nationality +
##     complement, family = "binomial", data = uboat.target.df)
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -2.763e+00  2.729e-01 -10.127  < 2e-16 ***
## day_periodevening         5.385e-01  2.059e-01   2.615 0.008934 **
## day_periodmidday          1.936e-01  2.453e-01   0.789 0.430062
## day_periodmorning         2.055e-01  2.363e-01   0.870 0.384486
## day_periodsmall_hours     4.826e-01  1.996e-01   2.418 0.015597 *
## day_periodsunrise         5.337e-01  2.357e-01   2.264 0.023580 *
## day_periodsunset          3.289e-01  2.235e-01   1.471 0.141186
## weekdayMonday            -4.034e-01  1.770e-01  -2.279 0.022639 *
## weekdaySaturday           1.760e-01  1.654e-01   1.064 0.287246
## weekdaySunday            -3.063e-01  1.738e-01  -1.762 0.077999 .
## weekdayThursday          -3.366e-02  1.716e-01  -0.196 0.844485
## weekdayTuesday           -3.175e-01  1.727e-01  -1.838 0.066076 .
## weekdayWednesday         -4.560e-01  1.768e-01  -2.579 0.009918 **
## dead                      3.306e-03  1.077e-03   3.069 0.002150 **
```

```
## tonnage                   8.115e-05  1.558e-05   5.207 1.92e-07 ***
## nationalityArgentinian    1.439e+00  1.242e+00   1.159 0.246566
## nationalityAustralian     1.983e+01  3.956e+03   0.005 0.996000
## nationalityBelgian        1.426e+00  4.036e-01   3.534 0.000410 ***
## nationalityBrazilian      1.024e-01  6.312e-01   0.162 0.871164
## nationalityBritish        7.663e-01  1.550e-01   4.944 7.66e-07 ***
## nationalityCanadian       2.078e-01  4.321e-01   0.481 0.630671
## nationalityChilean       -1.554e+01  3.956e+03  -0.004 0.996866
## nationalityChinese       -1.587e+01  3.956e+03  -0.004 0.996800
## nationalityColombian     -1.486e+01  2.257e+03  -0.007 0.994746
## nationalityCuban         -1.502e+01  1.929e+03  -0.008 0.993786
## nationalityDanish         1.986e+00  4.158e-01   4.777 1.78e-06 ***
## nationalityDominican     -1.503e+01  1.958e+03  -0.008 0.993876
## nationalityDutch          1.038e+00  2.474e-01   4.195 2.73e-05 ***
## nationalityEgyptian      -1.932e-01  7.522e-01  -0.257 0.797311
## nationalityEstonian       1.927e+00  6.544e-01   2.945 0.003233 **
## nationalityFaroese       -1.525e+01  2.754e+03  -0.006 0.995583
## nationalityFinnish        7.046e-01  6.576e-01   1.072 0.283925
## nationalityFrench         1.233e+00  3.872e-01   3.185 0.001446 **
## nationalityGerman        -1.556e+01  1.943e+03  -0.008 0.993609
## nationalityGreek          1.276e+00  2.524e-01   5.056 4.28e-07 ***
## nationalityHonduran       5.052e-02  1.067e+00   0.047 0.962247
## nationalityIcelandic     -1.501e+01  1.308e+03  -0.011 0.990847
## nationalityIndian        -1.494e+01  3.956e+03  -0.004 0.996987
## nationalityIrish          1.677e+00  8.910e-01   1.883 0.059760 .
## nationalityItalian        9.537e-01  1.169e+00   0.816 0.414491
## nationalityLatvian        1.341e+00  8.545e-01   1.569 0.116554
## nationalityLebanese      -1.501e+01  2.284e+03  -0.007 0.994756
## nationalityLithuanian    -1.545e+01  3.956e+03  -0.004 0.996884
## nationalityMexican       -1.560e+01  1.593e+03  -0.010 0.992190
## nationalityNicaraguan    -1.496e+01  2.278e+03  -0.007 0.994759
## nationalityNorwegian      9.005e-01  2.024e-01   4.448 8.66e-06 ***
## nationalityPalestinian   -1.496e+01  1.310e+03  -0.011 0.990892
## nationalityPanamanian     1.085e+00  2.948e-01   3.679 0.000234 ***
## nationalityPolish         8.707e-01  1.154e+00   0.755 0.450412
## nationalityPortuguese     6.061e-01  1.102e+00   0.550 0.582289
## nationalityRomanian      -1.550e+01  3.956e+03  -0.004 0.996873
## nationalitySouth African  1.273e+00  1.263e+00   1.008 0.313336
## nationalitySoviet        -1.518e+01  3.900e+02  -0.039 0.968950
## nationalitySpanish        1.353e+00  8.472e-01   1.597 0.110155
## nationalitySwedish        8.828e-01  3.114e-01   2.835 0.004585 **
## nationalitySyrian        -1.503e+01  1.311e+03  -0.011 0.990850
## nationalityUruguayan     -1.540e+01  3.956e+03  -0.004 0.996894
## nationalityVenezuelan    -1.480e+01  3.956e+03  -0.004 0.997016
## nationalityYugoslavian   -1.580e-01  1.059e+00  -0.149 0.881464
## complement               -1.280e-03  5.218e-04  -2.453 0.014180 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 3143.2  on 3239  degrees of freedom
## Residual deviance: 2928.2  on 3180  degrees of freedom
##   (218 observations deleted due to missingness)
```

```
## AIC: 3048.2
##
## Number of Fisher Scoring iterations: 16
```

From the above output, it seems like at least some of the variables in the model contribute to explaining whether an attack was conducted by a *hvt*. The stars to the right of the model parameters indicate they are significant in the mode. No stars is not significant and three stars is very significant.

Prediction example

What we would do in proper modelling is work our way through a range of model diagnostics as well as use some libraries to determine via a range of methods the most effective choice of variables to use to predict whether a *hvt* was involved in an attack. However, this will become quite technical so we won't do that in this example. Rather, we'll show how we can use a model once we have tested that it is a good fit for the data and is consistent with the model assumptions.

Experimenting with predictions

Just as an example in exploring the utility of such a model, we'll go ahead and run some predictions.

Remember, the main intent with this model, is we'd like to be able to predict if an attack was likely conducted by one of the highly competent commanders we've identified during our analysis.

We'll try to predict if an attack was conducted by a high value commander by using the following inputs.

- day_periodsmall_hours
- weekdayMonday
- dead = 15
- tonnage = 11000
- nationalityBritish
- complement = 30

We can make our prediction plugging in the values from above using the following code:

```
new = data.frame(day_period = "small_hours", weekday = "Monday", dead = 30,
                 tonnage = 11000, nationality = "British", complement = 30)
predict(comd.lm, new, type = 'response')
```

```
##         1
## 0.275963
```

Our model gives us 0.28, which means it is unlikely that such an attack with those input figures was conducted by a high value commander. Let's change some inputs and see how the prediction changes. We'll increase the number killed, the tonnage sunk and number of total personnel.

```
new_two = data.frame(day_period = "small_hours", weekday = "Monday", dead = 45,
                     tonnage = 41000, nationality = "British", complement = 50)
predict(comd.lm, new_two, type = 'response')
```

```
##         1
## 0.8166745
```

Our model gives us 0.82. Now our model is tending to indicate that such an attack would be much more likely to have been conducted by a hvt. We did increase the values so it makes sense that the model would increase the likelihood that the attack was conducted by a hvt.

Remember that the hvt got the value of 1 and all others got 0.

Let's now keep some values low and some high and see what happens. We'll increase those dead but lower tonnage lost.

```
new_three = data.frame(day_period = "small_hours", weekday = "Monday", dead = 65,
                       tonnage = 21000, nationality = "British", complement = 67)
predict(comd.lm, new_three, type = 'response')
```

```
##         1
## 0.4788392
```

While we have increased those killed, it's tending to indicate that the attack was less likely to be conducted by a hvt - but only just. We'll now keep the complement the same, lower the number of dead but increase the tonnage.

```
new_four = data.frame(day_period = "small_hours", weekday = "Monday", dead = 12,
                      tonnage = 51000, nationality = "British", complement = 67)
predict(comd.lm, new_four, type = 'response')
```

```
##         1
## 0.8979486
```

And while we've lowered the number of those killed, we've increased the tonnage and this more strongly indicates the attack was conducted by a hvt. So we are seeing that tonnage is strongly influencing the model's predictions.

We'll now keep the numbers the same but change some of the categorical coefficients, noting we're telling the model that the target was British. Let's now change the nationality only. We'll it to a factor that is still significant and choose a French vessel.

```
new_five = data.frame(day_period = "small_hours", weekday = "Monday", dead = 12,
                      tonnage = 51000, nationality = "French", complement = 67)
predict(comd.lm, new_five, type = 'response')
```

```
##         1
## 0.9335028
```

The model actually increased the confidence that this attack was conducted by a hvt by keeping the numbers the same but changing the factor with the new coefficient only, from British to French.

While we still have work to do, we've been able to demonstrate some potential utility in building such a model. We've used the model to run some predictions using the following inputs:

- The day period
- The weekday
- How many were killed
- The tonnage of the struck vessel
- The nationality of the vessel

- How many crew were on board

This sort of model could provide some preliminary assessment regarding the value of the uboat that conducted an attack. If used in real-time during a conflict like WWII, this could inform or drive operational decisions around how to respond, what assets to use and how quickly to respond. We could build a similar model for uboats and compare the accuracy but we won't do that for now.

Note that some of the factors are more significant than others. So depending on what the inputs are, the model accuracy will vary.

Conclusion

This concludes the example data analysis on the German Uboat attack data from WWII. This example aimed to highlight the sorts of analysis that can be done with a curated *event* or *activity* dataset. This example focused on exploratory data analysis using primarily graphical analyses. Modelling and time series forecasting is a deeper more technical analysis that can be covered at a later stage.