Programming with Data

Session 7: Forecasting with Logistic Regressions

Dr. Wang Jiwei
Master of Professional Accounting

Binary outcomes

What are binary outcomes?

- Thus far we have talked about events with continuous outcomes
  - Revenue, earnings, ROA, etc
- Binary outcomes only have two possible outcomes
  - Did something happen, *yes* or *no*?
  - Is a statement *true* or *false*?

- Financial accounting:
  - Will the company's earnings meet analysts' expectations?
  - Will the company have positive earnings?
- Managerial accounting:
  - Will we have problem with our supply chain?
  - Will our customer go bankrupt?
- Audit:
  - Is the company committing fraud?
- Taxation:
  - Is the company too aggressive in their tax positions?
- Management and strategy
  - Does the business model change?

> We can assign a probability to any of these

Binary classficiation algos

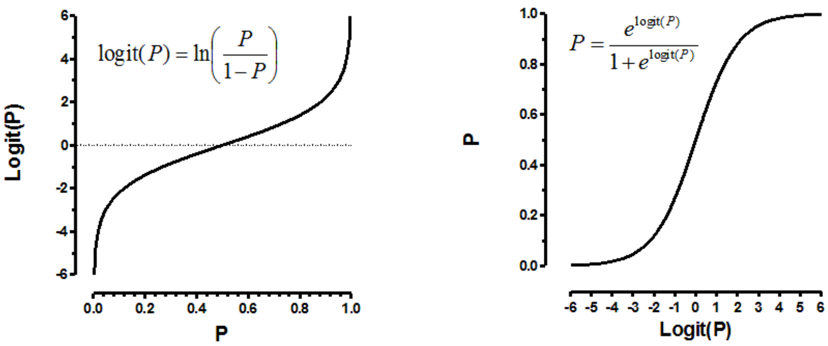
- Popular algorithms that can be used for binary classification include:
  - Logistic Regression (today's session)
  - Decision Trees (to be covered)
  - k-Nearest Neighbors
  - Support Vector Machine
  - Naive Bayes

Logistic regression

- When modeling a binary outcome, we use logistic regression
  - A.k.a. logit model
- The *logit* function is $logit(p) = \log\left(\frac{p}{1-p}\right)$
  - Also called *log odds*, see next slide

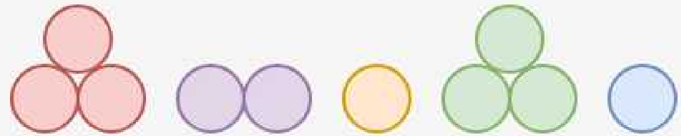
$$\log\left(\frac{\text{Prob}(y=1|X)}{1-\text{Prob}(y=1|X)}\right) = \alpha + \beta_1 x_1 + \beta_2 x_2 + \ldots + \varepsilon$$

- The **sign** of the coefficients means the same as before
  - +: *increases* the likelihood of y occurring
  - -: *decreases* the likelihood of y occurring

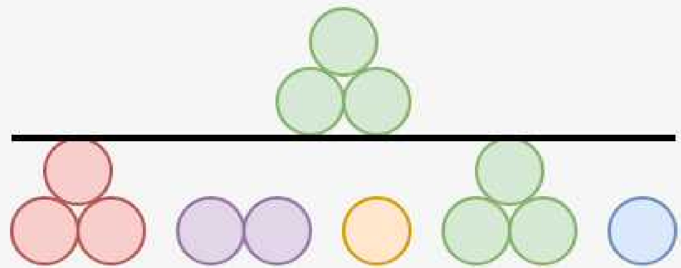


Odds vs probability

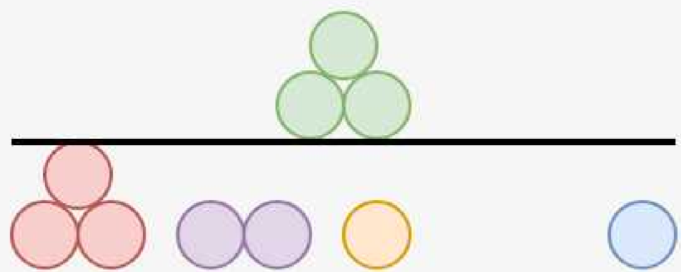
We have the following 10 objects:



The **probability** of green is: 3/10



The **odds** of green is: 3 to 7



Logistic regression interpretation

Interpreting logit values

- The level of the coefficient is different
  - The relationship isn't linear between x_i and y now
  - Instead, coefficient is in *log odds*
    - Thus, $e^{X\beta}$ gives you the *odds*, $o = \frac{p}{1-p}$
    - To get probability, p, we can calculate $p = \frac{o}{1+o}$
- Interpretation: for a one-unit increase in x_i
  - the *log odds* for $y = 1$ increase by β (same as the OLS), holding others at a fixed value
  - the *odds* for $y = 1$ increase by $(e^{\beta} - 1)$ times of baseline odds (ie, odds before the change), holding others at a fixed value
    - $log(o2) - log(o1) = \beta$
    - $log(o2/o1) = \beta$
    - $o2/o1 = e^{\beta}$
    - $o2 - o1 = (e^{\beta} - 1) * o1$
  - you need to sum all relevant log odds before converting to probability
  - Click here for a review

Implement logit regression

- The logistic model is related to our previous linear models as such:

> Both linear and logit models are under the class of General Linear Models (GLMs)

- To regress a GLM, we use the `glm()` command.

- To run a logit regression:

```
mod <- glm(y ~ x1 + x2 + x3 + ..., data = df, family = binomial)

summary(mod)
```

> `family = binomial` is what sets the model to be a logit

- In fact, the `lm()` we have been using is actually `glm()` when you specify the option `family = gaussian`

Example logit regression

> Do holidays increase the likelihood that a department more than doubles its store's average weekly sales across departments?

```r
# Create the binary variable from Walmart sales data
df$double <- ifelse(df$Weekly_Sales > df$store_avg * 2, 1, 0)
model1 <- glm(double ~ IsHoliday, data = df, family = binomial)
tidy(model1)
```

```
## # A tibble: 2 x 5
##   term          estimate std.error statistic  p.value
##   <chr>            <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)     -3.45    0.00924    -373.  0
## 2 IsHolidayTRUE    0.539   0.0278       19.4 1.09e-83
```

> Holidays increase the odds... but by how much?

- There are two way to interpret this:
  1. Coefficient by coefficient
  2. In total

Interpretting specific coefficients

$$logodds(Double\ sales) = -3.45 + 0.54 IsHoliday$$

- Interpreting specific coefficients is easiest done manually
- Odds for the $IsHoliday$ coefficient are `exp(0.54) = 1.72`
  - This means that having a holiday modifies the baseline (i.e., non-Holiday) odds by 1.72 to 1
    - Where 1 to 1 is considered no change (`exp(0) = 1`)
  - Baseline `exp(-3.45)` is 0.032 to 1

```
# Automating the above:
exp(coef(model1))
```

```
##   (Intercept) IsHolidayTRUE
##    0.03184725    1.71367497
```

Interpretting in total

- It is important to note that log odds are additive
  - So, calculate a new log odd by plugging in values for variables and adding it all up
    - Holiday: $-3.45 + 0.54 * 1 = -2.91$
    - No holiday: $-3.45 + 0.54 * 0 = -3.45$
- Then calculate odds and log odds like before
  - With holiday: `exp(-2.91) = 0.055`
  - Without holiday: `exp(-3.45) = 0.032`
  - Ratio of holiday to without: 1.72!
    - This is the individual log odds for holiday

> We need to specify values to calculate log odds in total

Converting to probabilities

- We can calculate a probability at any given point using the log odds

$$Probability = \frac{odds}{odds + 1}$$

- Probability of double sales...
  - With a holiday: `0.055 / (0.055 + 1) = 0.052`
  - Without a holiday: `0.032 / (0.032 + 1) = 0.031`

> These are easier to interpret, but require specifying values

Using predict() to simplify it

- `predict()` can calculate log odds and probabilities for us with minimal effort
  - Specify `type = "response"` to get probabilities

```
IsHoliday <- c(FALSE, TRUE)
test_data <- as.data.frame(IsHoliday)
predict(model1, test_data)  # log odds if no type = "response"
```

```
##         1         2
## -3.446804 -2.908164
```

```
predict(model1, test_data, type = "response")  #probabilities
```

```
##          1          2
## 0.03086431 0.05175146
```

- Here, we see the baseline probability is 3.1%
- The probability of doubling sales on a holiday is higher, at 5.2%

R practice: Logit

- A continuation of last session's practices answering:
  - Is Walmart more likely to see a year over year decrease in quarterly revenue during a recession?
- Practice using `mutate()` and `glm()`
- Do exercises 1 and 2 in the practice file
  - R Practice

What about more complex model?

```
model2 <- glm(double ~ IsHoliday + Temperature + Fuel_Price,
              data = df, family = binomial)
tidy(model2)
```

```
## # A tibble: 4 x 5
##   term          estimate std.error statistic   p.value
##   <chr>            <dbl>     <dbl>     <dbl>     <dbl>
## 1 (Intercept)    -1.78    0.0673       -26.4 1.93e-153
## 2 IsHolidayTRUE   0.370   0.0284        13.0 8.80e- 39
## 3 Temperature    -0.0108  0.000470     -23.0 1.70e-117
## 4 Fuel_Price     -0.309   0.0196       -15.8 6.20e- 56
```

```
# Odds
exp(coef(model2))
```

```
##   (Intercept) IsHolidayTRUE   Temperature    Fuel_Price
##     0.1692308     1.4483570     0.9892316     0.7340376
```

> We need to specify values for all inputs to determine probabilities, ie, the impact of each input depends on the values of the others!

Probabilities

```r
# Average probability in September
hday_sep    <- mean(predict(model2, filter(df, IsHoliday, month == 9),
                            type = "response"))
no_hday_sep <- mean(predict(model2, filter(df, !IsHoliday, month == 9),
                            type = "response"))
# Average probability in December
hday_dec    <- mean(predict(model2, filter(df, IsHoliday, month == 12),
                            type="response"))
no_hday_dec <- mean(predict(model2, filter(df, !IsHoliday, month == 12),
                            type="response"))

html_df(data.frame(Month=c(9, 9, 12, 12), IsHoliday=c(FALSE, TRUE, FALSE, TRUE),
                   Probability=c(no_hday_sep, hday_sep, no_hday_dec, hday_dec)))
```

Month	IsHoliday	Probability
9	FALSE	0.0266789
9	TRUE	0.0374761
12	FALSE	0.0398377
12	TRUE	0.0586483

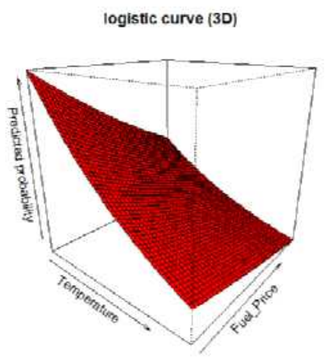
A bit easier: Marginal effects

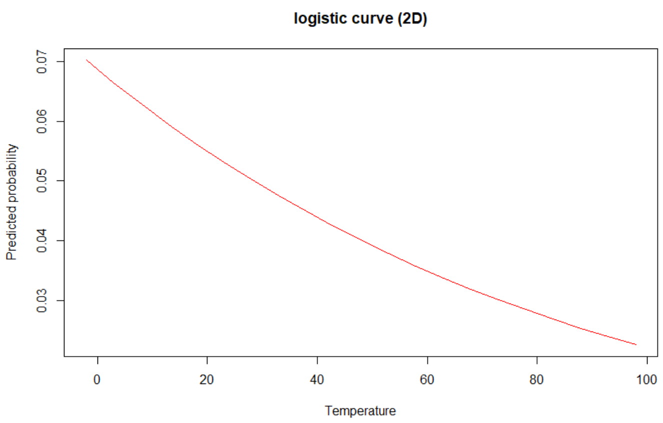
> Marginal effects tell us the change in our output for a change of 1 unit to an input

- Marginal effects are partial derivatives (slope of the tangent line) of a regression equation with respect to each variable in the model for each unit in the data
  - In OLS regression with no interactions or higher-order term (such as polynomial functions with quadratic terms x^2), the estimated coefficients are marginal effects
- Using `package:margins`, we can calculate marginal effects
- There are a few types that we could calculate:
  - An *Average Marginal Effect* tells us what the average effect of an input is across all values in our data
    - This is the default method in the package
  - We can also specify a specific value to calculate marginal effects at

Logistic 2D/3D curve

> Marginal effect means the partial derivative of any given point on the surface





Marginal effects in action

```r
# Calculate Average Marginal Effects (AME)
# It will take a while
library(margins)
m <- margins(model2)
m
```

```
##  Temperature Fuel_Price IsHoliday
##   -0.0003377  -0.009644   0.01334
```

- By default, the margins() returns the Average Marginal Effect (AME)
- A holiday increases the probability of doubling by a flat 1.33%
  - Not too bad when you consider that the probability of doubling is 3.23%
- If the temperature goes up by 1°F (0.55°C), the probability of doubling changes by -0.03%
- If the fuel price increases by 1 USD for 1 gallon of gas, the probability of doubling changes by -0.96%

`package:margins` niceties

- We can get some extra information about our marginal effects through `summary()`:

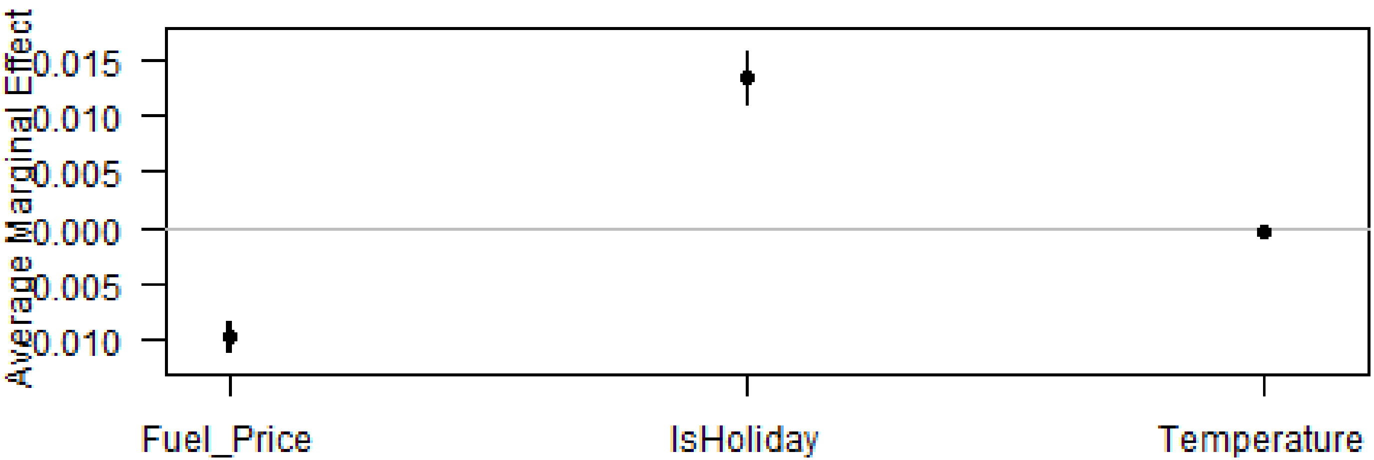
```
summary(m) %>%
  html_df()
```

factor	AME	SE	z	p	lower	upper
Fuel_Price	-0.0096438	0.0006163	-15.64800	0	-0.0108517	-0.0084359
IsHoliday	0.0133450	0.0011754	11.35372	0	0.0110413	0.0156487
Temperature	-0.0003377	0.0000149	-22.71255	0	-0.0003668	-0.0003085

- Those p-values work just like with our linear models
- We also get a confidence interval

Plotting marginal effects

```r
# Note: The `which...` part is absolutely necessary at the moment
# due to a bug in the package (mismatch of factors and AME values
# you may try to remove `which...` to see what happened
plot(m, which = summary(m)$factor)
```



Marginal effects at a specified value

```
margins(model2, at = list(IsHoliday = c(TRUE, FALSE)),
        variables = c("Temperature", "Fuel_Price")) %>%
  summary() %>%
  html_df()
```

factor	IsHoliday	AME	SE	z	p	lower	upper
Fuel_Price	FALSE	-0.0093401	0.0005989	-15.59617	0	-0.0105139	-0.0081664
Fuel_Price	TRUE	-0.0131335	0.0008717	-15.06650	0	-0.0148420	-0.0114250
Temperature	FALSE	-0.0003271	0.0000146	-22.46024	0	-0.0003556	-0.0002985
Temperature	TRUE	-0.0004599	0.0000210	-21.92927	0	-0.0005010	-0.0004188

- specify the values through *at* argument, *variables* for the features of changes

> On a holiday, if fuel price changes by 1 unit, the probability of doubling changes by -1.31%

Marginal effects at a specified value

```r
margins(model2, at = list(Temperature = c(0, 20, 40, 60, 80)),
        variables = c("IsHoliday")) %>%
  summary() %>%
  html_df()
```

factor	Temperature	AME	SE	z	p	lower	upper
IsHoliday	0	0.0234484	0.0020168	11.62643	0	0.0194955	0.0274012
IsHoliday	20	0.0194072	0.0016710	11.61387	0	0.0161320	0.0226824
IsHoliday	40	0.0159819	0.0013885	11.51001	0	0.0132604	0.0187033
IsHoliday	60	0.0131066	0.0011592	11.30623	0	0.0108345	0.0153786
IsHoliday	80	0.0107120	0.0009732	11.00749	0	0.0088046	0.0126193

> At 0 temperature, a holiday will result in 2.34% increase of probability of doubling.

Today's Application: Shipping delays

The question

- Can we leverage global weather data to predict shipping delays?

A bit about shipping data

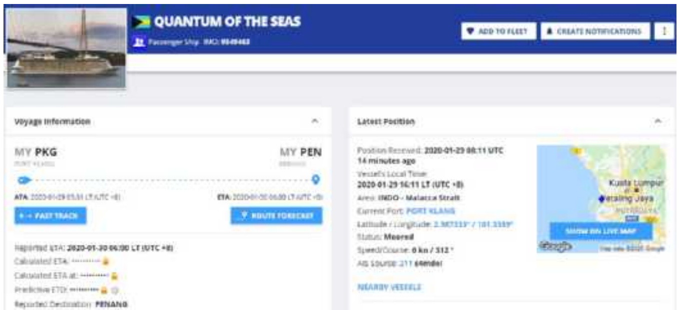
- WRDS doesn't have shipping data
- There are, however, vendors for shipping data, such as:

- They pretty much have any data you could need:
  - Over 650,000 ships tracked using ground and satellite based AIS
    - AIS: Automatic Identification System
  - Live mapping
  - Weather data
  - Fleet tracking
  - Port congestion
  - Inmarsat support for ship operators

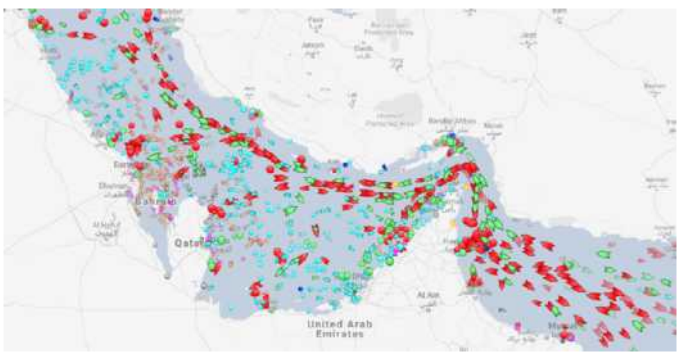
What can we see from naval data?

SMU
SINGAPORE MANAGEMENT UNIVERSITY
School of Accountancy

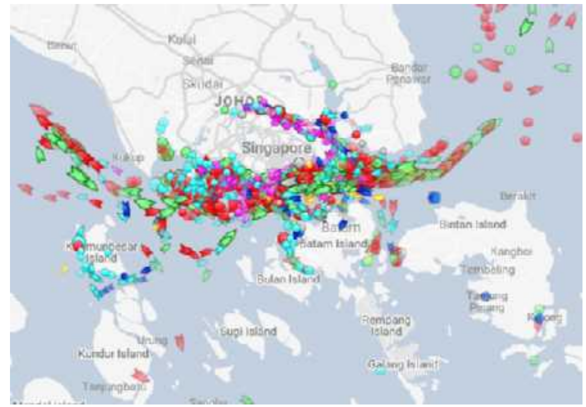
Cruise: Quantum of the Seas



Oil tankers in the Persian gulf

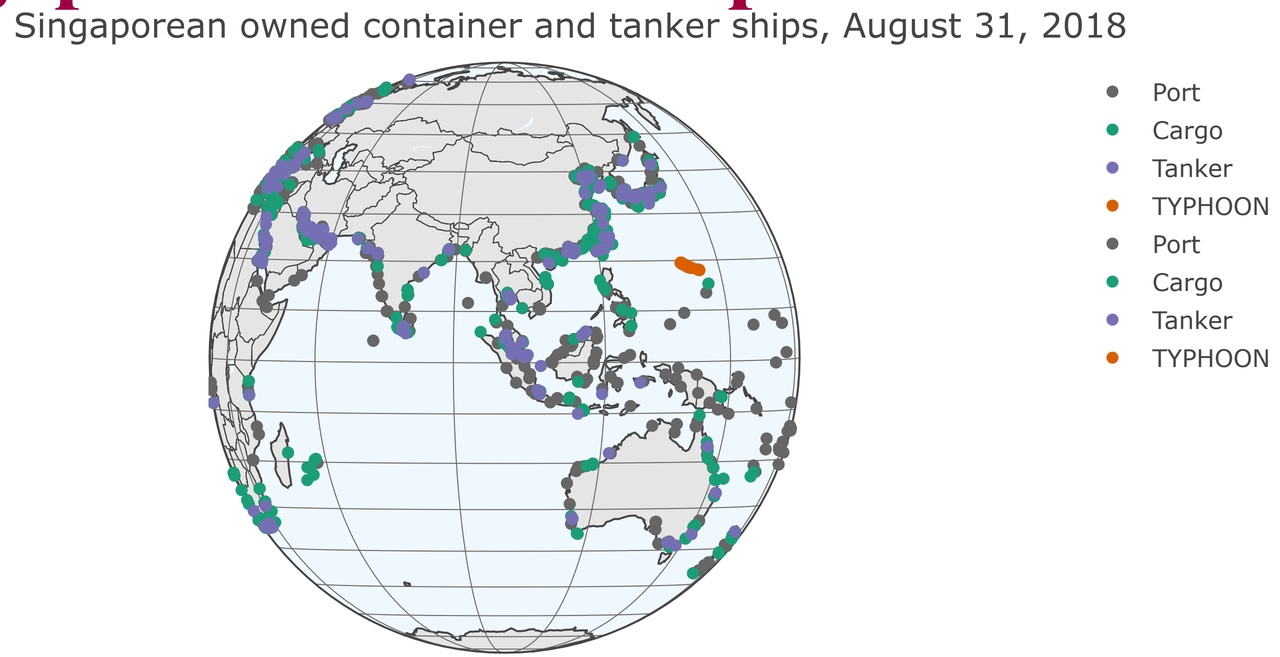


Busiest port for transshipment (Singapore)



Busiest ports by containers and tons (Shanghai & Ningbo-Zhoushan, China)

Singaporean owned ships



Code for last slide's map

```r
library(plotly)  # for plotting
library(RColorBrewer)  # for colors
# plot with boats, ports, and typhoons
# Note: geo is defined in the code file -- it controls layout
palette = brewer.pal(8, "Dark2")[c(1, 8, 3, 2)]
p <- plot_geo(colors = palette) %>%
  add_markers(data = df_ports, x = ~port_lon, y = ~port_lat, color = "Port") %>%
  add_markers(data = df_Aug31, x = ~lon, y = ~lat, color = ~ship_type,
              text = ~paste('Ship name', shipname)) %>%
  add_markers(data = typhoon_Aug31, x = ~lon, y = ~lat, color="TYPHOON",
              text = ~paste("Name", typhoon_name)) %>%
   layout(showlegend = TRUE, geo = geo,
    title = 'Singaporean owned container and tanker ships, August 31, 2018')
p
```

- `plot_geo()` is from `package:plotly`
- `add_markers()` adds points to the map
- `layout()` adjusts the layout
- Within geo, a list, the following makes the map a globe
  - `projection = list(type = "orthographic")`

What might matter for shipping?

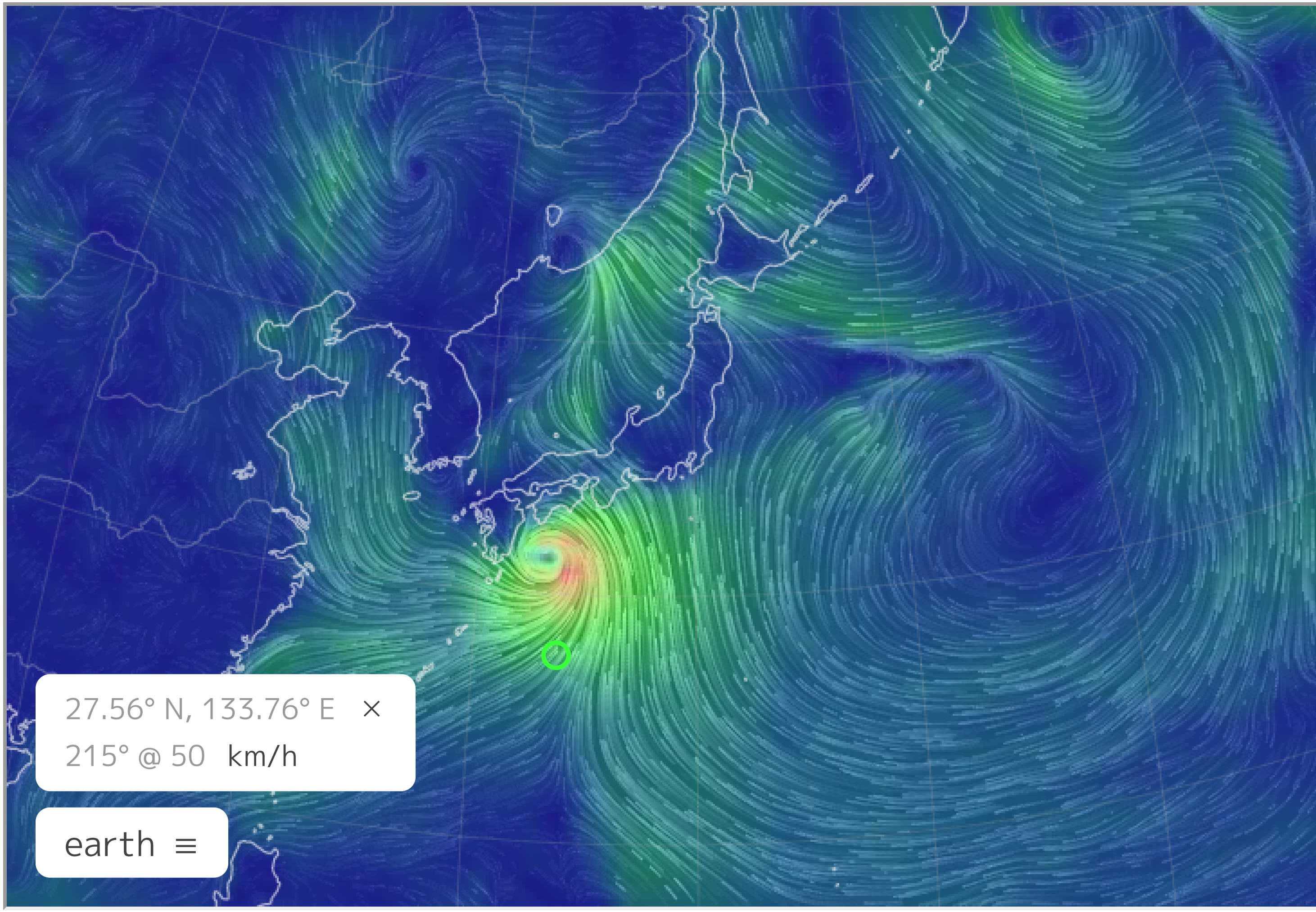
> What observable events or data might provide insight as to whether a naval shipment will be delayed or not?

1. Typhoons

Typhoon Jebi

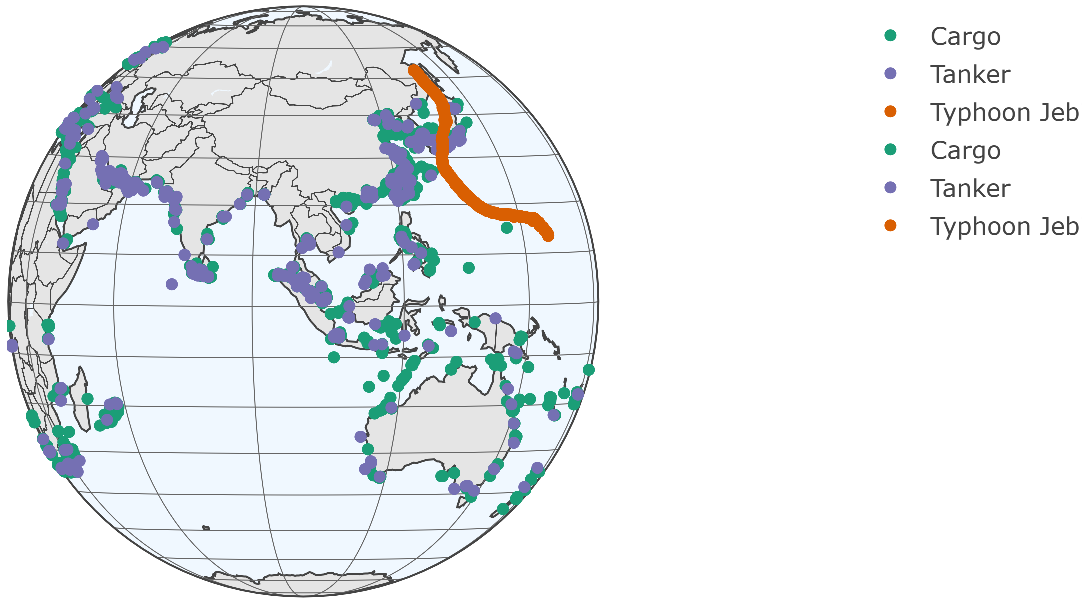
- click to play on youtube

Typhoon Jebi



Typhoons in the data

Singaporean container/tanker ships, September 4, 2018, evening



Code for last slide's map

```
# plot with boats and typhoons
palette = brewer.pal(8, "Dark2")[c(1, 3, 2)]
p <- plot_geo(colors = palette) %>%
  add_markers(data = df_all[df_all$frame == 14,], x = ~lon, y = ~lat,
              color = ~ship_type, text = ~paste('Ship name', shipname)) %>%
  add_markers(data = typhoon_Jebi, x = ~lon,
              y = ~lat, color = "Typhoon Jebi",
              text = ~paste("Name", typhoon_name, "</br>Time: ", date)) %>%
   layout(showlegend = TRUE, geo = geo,
          title = 'Singaporean ships, September 4, 2018, evening')
p
```

- This map is made the same way as the first map

R Practice on mapping

- Practice interactive mapping using typhoon data
  - 1 map using `package:plotly`
- Bonus practice
  - 1 map using `package:leaflet`
- There is another interactive mapping `package:sf` but its installation is not friendly on a Mac
- Do exercises 3 and 4 in the practice file
  - R Practice

Predicting delays due to typhoons

Data

- If the ship will report a delay of at least 3 hours some time in the next 12-24 hours
- What we have:
  - Ship location
  - Typhoon location
  - Typhoon wind speed

> We need to calculate distance between ships and typhoons

Distance for geo

- There are a number of formulas for this
  - *Haversine* for a simple calculation
  - *Vincenty's formulae* for a complex, incredibly accurate calculation
    - Accurate within **0.5mm**
- Use `distVincentyEllipsoid()` from `package:geosphere` to get a reasonably quick and accurate calculation
  - Calculates distance between two sets of points, x and y, structured as matrices
  - Matrices must have longitude in the first column and latitude in the second column
  - Provides distance in meters by default

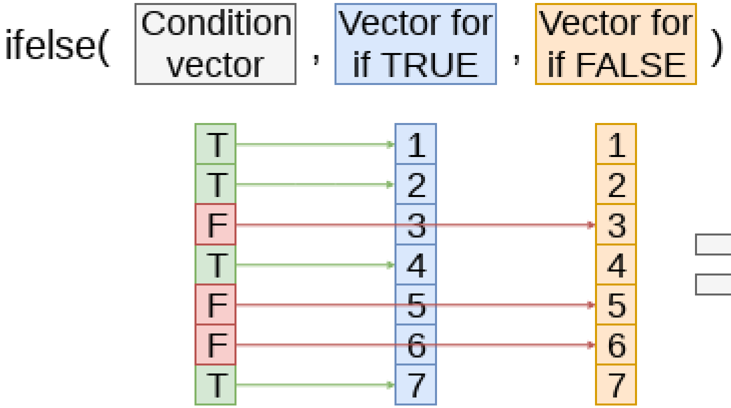
```r
library(geosphere)
x <- as.matrix(df3[ , c("lon", "lat")])  # ship location
y <- as.matrix(df3[ , c("ty_lon", "ty_lat")]) # typhoon location

df3$dist_typhoon <- distVincentyEllipsoid(x, y) / 1000 # convert to KM
```

Clean up

- Some indicators to cleanly capture how far away the typhoon is

```
df3$typhoon_500 = ifelse(df3$dist_typhoon < 500 &
                         df3$dist_typhoon >= 0, 1, 0)
df3$typhoon_1000 = ifelse(df3$dist_typhoon < 1000 &
                          df3$dist_typhoon >= 500, 1, 0)
df3$typhoon_2000 = ifelse(df3$dist_typhoon < 2000 &
                          df3$dist_typhoon >= 1000, 1, 0)
```



Do typhoons delay shipments?

```
fit1 <- glm(delayed ~ typhoon_500 + typhoon_1000 + typhoon_2000,
            data = df3, family = binomial)
summary(fit1)
```

```
##
## Call:
## glm(formula = delayed ~ typhoon_500 + typhoon_1000 + typhoon_2000,
##     family = binomial, data = df3)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.2502  -0.2261  -0.2261  -0.2261   2.7127
##
## Coefficients:
##              Estimate Std. Error  z value Pr(>|z|)
## (Intercept)  -3.65377    0.02934 -124.547   <2e-16 ***
## typhoon_500   0.14073    0.16311    0.863   0.3883
## typhoon_1000  0.20539    0.12575    1.633   0.1024
## typhoon_2000  0.16059    0.07106    2.260   0.0238 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 14329  on 59184  degrees of freedom
## Residual deviance: 14322  on 59181  degrees of freedom
##   (3866 observations deleted due to missingness)
## AIC: 14330
##
```

Interpretation of coefficients

```
odds1 <- exp(coef(fit1))
odds1
```

```
##  (Intercept)  typhoon_500 typhoon_1000 typhoon_2000
##   0.02589334   1.15111673   1.22800815   1.17420736
```

- Ships 1,000 to 2,000 km from a typhoon have a 17% increased odds of having a delay

```
m1 <- margins(fit1)
summary(m1)
```

```
##        factor    AME     SE      z      p   lower  upper
##  typhoon_1000 0.0052 0.0032 1.6322 0.1026 -0.0010 0.0115
##  typhoon_2000 0.0041 0.0018 2.2570 0.0240  0.0005 0.0076
##   typhoon_500 0.0036 0.0042 0.8626 0.3883 -0.0046 0.0117
```

- Ships 1,000 to 2,000 km from a typhoon have an extra 0.41% chance of having a delay (baseline of 2.5%, ie, all x = 0)

Interpretation of coefficients

- Alternatively, we may calculate actual probability by summing up all relevant log odds.

```
prob_odds1 <- c(exp(coef(fit1)[1]),
                exp(coef(fit1)[c(2, 3, 4)] + coef(fit1)[c(1, 1, 1)]))
probability1 <- prob_odds1 / (1 + prob_odds1)
probability1
```

```
##  (Intercept)  typhoon_500 typhoon_1000 typhoon_2000
##   0.02523980   0.02894356   0.03081733   0.02950702
```

- Ships 1,000 to 2,000 km from a typhoon have a 3% chance of having a delay (baseline of 2.5%)

- Note the calculation of odds for each scenario:

    - all typhoon_ = 0: α
    - typhoon_500 = 1: $\alpha + \beta 1$
    - typhoon_1000 = 1: $\alpha + \beta 2$
    - typhoon_2000 = 1: $\alpha + \beta 3$

What about typhoon intensity?

- Hong Kong's typhoon classification: Official source
    1. 41-62 km/h: Tropical depression
    2. 63-87 km/h: Tropical storm
    3. 88-117 km/h: Severe tropical storm
    4. 118-149 km/h: **Typhoon**
    5. 150-184 km/h: **Severe typhoon**
    6. 185+km/h: **Super typhoon**

```r
# 1 knot (nautical mile/h) = 1.852 km/h
# cut() makes a categorical variable out of a numerical variable
# using specified bins
df3$Super <- ifelse(df3$intensity_vmax * 1.852 >= 185, 1, 0)
df3$Moderate <- ifelse(df3$intensity_vmax * 1.852 >= 88 &
                         df3$intensity_vmax * 1.852 < 185, 1, 0)
df3$Weak <- ifelse(df3$intensity_vmax * 1.852 >= 41 &
                     df3$intensity_vmax * 1.852 < 88, 1, 0)
df3$HK_intensity <- cut(df3$intensity_vmax * 1.852, c(-1, 41, 62, 87, 117,
                                                       149, 999))
table(df3$HK_intensity)
```

```
##
##   (-1,41]   (41,62]   (62,87]  (87,117] (117,149] (149,999]
##      3398     12039     12615     11527      2255     21141
```

Typhoon intensity and delays

```
fit2 <- glm(delayed ~ (typhoon_500 + typhoon_1000 + typhoon_2000) :
              (Weak + Moderate + Super), data = df3,
            family = binomial)
tidy(fit2)
```

```
## # A tibble: 10 x 5
##    term                  estimate std.error statistic p.value
##    <chr>                    <dbl>     <dbl>     <dbl>   <dbl>
##  1 (Intercept)            -3.65      0.0290 -126.      0
##  2 typhoon_500:Weak       -0.00879   0.213    -0.0413 0.967
##  3 typhoon_500:Moderate    0.715     0.251     2.86   0.00430
##  4 typhoon_500:Super      -8.91    123.       -0.0726 0.942
##  5 typhoon_1000:Weak       0.250     0.161     1.55   0.121
##  6 typhoon_1000:Moderate   0.123     0.273     0.451  0.652
##  7 typhoon_1000:Super     -0.0269    0.414    -0.0648 0.948
##  8 typhoon_2000:Weak       0.182     0.101     1.80   0.0723
##  9 typhoon_2000:Moderate   0.0253    0.134     0.189  0.850
## 10 typhoon_2000:Super      0.311     0.136     2.29   0.0217
```

> Moderate storms predict delays when within 500km
>
> Super typhoons predict delays when 1,000 to 2,000km away

Interpretation of coefficients

```
odds2 <- exp(coef(fit2))
odds2[c(1, 3, 10)]
```

```
##          (Intercept) typhoon_500:Moderate   typhoon_2000:Super
##           0.02589637           2.04505487           1.36507575
```

- Ships within 500km of a moderately strong storm have 104% higher *odds* of being delayed
- Ships 1,000 to 2,000km from a super typhoon have 36% higher *odds*

Interpretation of coefficients

```
prob_odds2 <- c(exp(coef(fit2)[1]),
                exp(coef(fit2)[c(3, 10)] + coef(fit2)[c(1, 1)]))
probability2 <- prob_odds2 / (1 + prob_odds2)
probability2
```

```
##          (Intercept) typhoon_500:Moderate   typhoon_2000:Super
##           0.02524268           0.05029586           0.03414352
```

- Ships within 500km of a moderately strong storm have a 5% chance of being delayed (baseline: 2.5%)
- Ships 1,000 to 2,000km from a super typhoon have a 3.4% chance

Marginal effects

```
m2 <- margins(fit2)
summary(m2) %>%
  html_df()
```

factor	AME	SE	z	p	lower	upper
Moderate	0.0007378	0.0006713	1.0990530	0.2717449	-0.0005779	0.0020535
Super	-0.0050241	0.0860163	-0.0584087	0.9534231	-0.1736129	0.1635647
typhoon_1000	0.0035473	0.0036186	0.9802921	0.3269420	-0.0035450	0.0106396
typhoon_2000	0.0039224	0.0017841	2.1985908	0.0279070	0.0004257	0.0074191
typhoon_500	-0.0440484	0.6803640	-0.0647424	0.9483791	-1.3775373	1.2894405
Weak	0.0009975	0.0005154	1.9353011	0.0529534	-0.0000127	0.0020077

- Delays appear to be driven mostly by 2 factors:
  1. A typhoon 1,000 to 2,000 km away from the ship
  2. Weak typhoons

Summary of Session 7

For next week

- Try to replicate the code
- Continue your Datacamp career track
- Continue with your project
  - You can start to explore models

R Coding Style Guide

Style is subjective and arbitrary but it is important to follow a generally accepted style if you want to share code with others. I suggest the The tidyverse style guide which is also adopted by Google with some modification

- Highlights of **the tidyverse style guide**:
  - *File names*: end with .R
  - *Identifiers*: variable_name, function_name, try not to use "." as it is reserved by Base R's S3 objects
  - *Line length*: 80 characters
  - *Indentation*: two spaces, no tabs (RStudio by default converts tabs to spaces and you may change under global options)
  - *Spacing*: x = 0, not x=0, no space before a comma, but always place one after a comma
  - *Curly braces {}*: first on same line, last on own line
  - *Assignment*: use `<-`, not = nor `->`
  - *Semicolon(;)*: don't use, I used once for the interest of space
  - *return()*: Use explicit returns in functions: default function return is the last evaluated expression
  - *File paths*: use relative file path "../../filename.csv" rather than absolute path "C:/mydata/filename.csv". Backslash needs \\

R packages used in this slide

This slide was prepared on 2021-09-08 from Session_7s.Rmd with R version 4.1.1 (2021-08-10) Kick Things on Windows 10 x64 build 18362 🙋.

The attached packages used in this slide are:

```
##    geosphere RColorBrewer       plotly      margins        rlang        broom
##     "1.5-10"      "1.1-2"    "4.9.4.1"     "0.3.26"     "0.4.11"      "0.7.9"
##    lubridate      forcats      stringr        dplyr        purrr        readr
##     "1.7.10"      "0.5.1"      "1.4.0"      "1.0.7"      "0.3.4"      "2.0.1"
##        tidyr       tibble      ggplot2    tidyverse   kableExtra        knitr
##      "1.1.3"      "3.1.3"      "3.3.5"      "1.3.1"      "1.3.4"       "1.33"
```