Statistical Modeling of Clinical Data

Anoop Mayampurath | amayampurath@peds.bsd.uchicago.edu
Julie Johnson | jjohnso3@bsd.uchicago.edu

February 1, 2018

Agenda

- Objective
- Constructing your study
- Composition of clinical data
- How to perform descriptive analyses?
- How to perform complex analyses?

This workshop is about analyzing clinical data.

Types of healthcare data

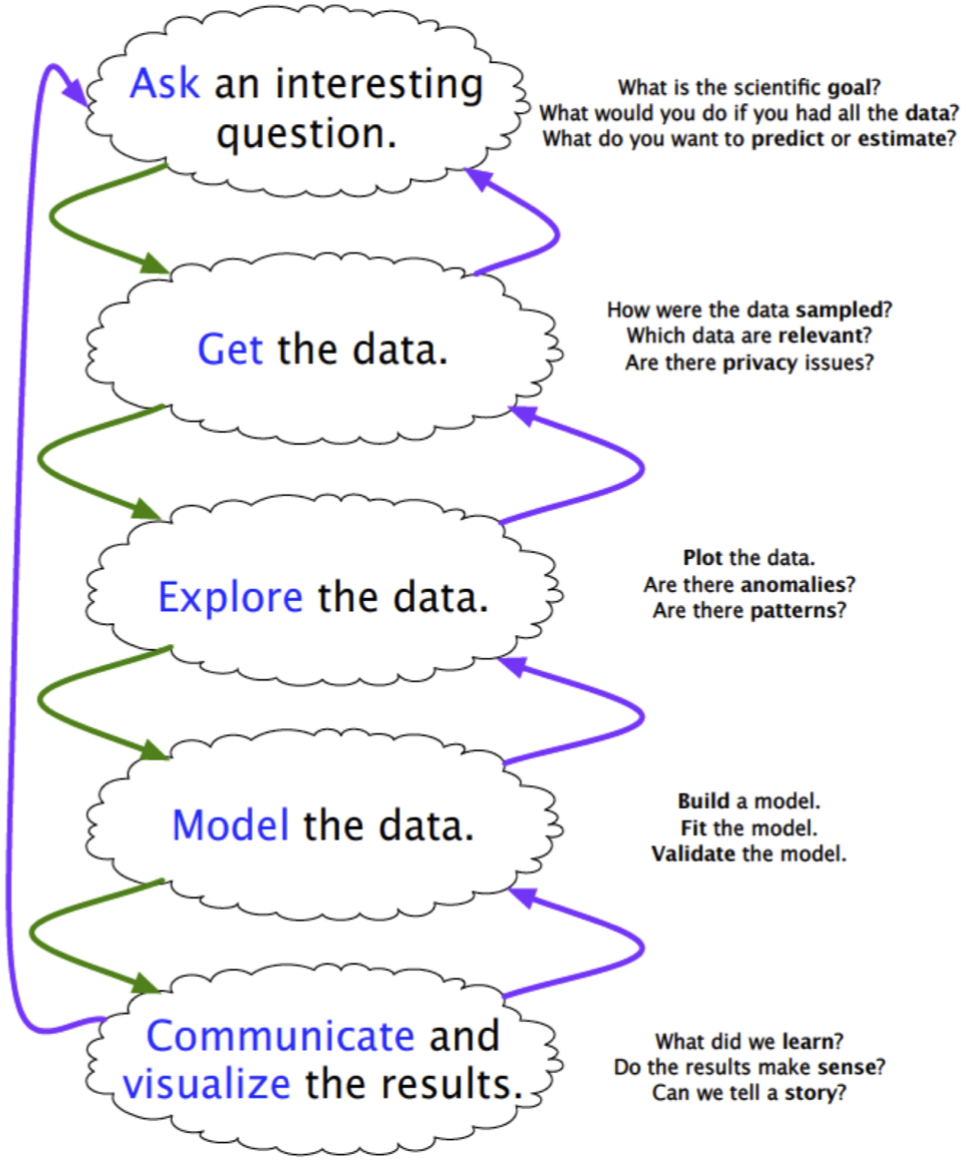
- Claims data: patient demographics, diagnosis codes, dates of service, cost of service, etc.
- **EHR data: everything above plus vitals, labs, meds, interventions, reports, and notes**.
- Socioeconomic data: average income, crime, access to healthy food, pharmacies
- Self-reported data: personalized data, wearable technology

General tips

- Don’t be scared of messy data
- Understand probability
- Learn how to do data processing
- Learn how to do data modeling
- Interpretation over blind application

I think there are three main steps in a data science project: you *collect* data (and questions), *analyze* it (using visualization and models), then *communicate* the results. It's rare to walk this process in one direction: often your analysis will reveal that you need new or different data, or when presenting results you'll discover a flaw in your model.

- Hadley Wickham



- Source: CS109: Harvard

General study designs

At the beginning of a study, there are several choices available to the researcher on how to conduct the study. These include:

- Descriptive (e.g. surveys, case studies)
- Associative (e.g. observational studies of type: outcome ~ exposure)
- Predictive (e.g. risk prediction)
- Review (e.g. literature review)
- Experimental (e.g. Randomized-Controlled Trials)
- Meta-analysis (i.e. combining the results of multiple studies)

Observational Study Design

Common types of observational study designs
- Case-Control study
- Retrospective cohort study
- Prospective cohort study
- Cross-sectional study

Case-Control study

Objective is to estimate the relative risk for an outcome from a specific variable (or risk factor or exposure) using odds ratios.

The dataset is built after the outcome is identified, following which the occurence of previous exposure is determined.

Exposure is loosely defined as whether the variable of interest holds true or not.

Key things for a case-control study

- Controls must be comparable to case except without the occurence of outcome.
- Non-matched case-control study: while building control, we ignore the number as well characteristics of the case.
- Matched case-control study: while building control, we take into account some characteristic of the case (e.g gender). Ways to match are 1:n case-control matching, distribution-based matching, and propensity matching.
- Essentially, when matching, you are minimizing the effect of confounders.
- You can't measure incidence rates (i.e. rate of outcome) in case-control studies.

Cohort Study

The idea is that you recruit subjects purely based on exposure status,i.e. none of them have developed an outcome. Then, you follow them in time until some of them develop an outcome. In other words, these are longitudical studies.

- Prospective cohort study: Identify the study population at the beginning. Determine exposure status. Follow them through time.
- Retrospective cohort study: Historical record is collected for all exposed/non-exposed subjects. Determine current outcome status.
- Association is measured in terms of relative risk or using survival analysis

Differences

- Case-control study does not use an entire cohort. As a result, you cannot measure outcome rates accurately. Retrospective cohorts use the entire cohort.

- Sample size for case-control is dependent of rates of exposure, not outcome. The reverse is true for cohort studies, i.e., sample size is based on rates of outcome, not exposure.

How to choose?

- Cohort studies provide the best information about causality.
- In cohort studies (both prospective and retrospective), you can also measure associations with different outcomes for the same exposure.
- Prospective cohort studies while robust and controllable, are expensive to conduct, with a potential danger of patients dropping off from the study.
- Generally speaking, cohort studies work well with rare exposures. It does not work for rare outcomes, the sample size has to be very high for finding proper risk for incidence.

- Case control studies are simpler to conduct. They are quick and inexpensive and good for studying outbreaks.
- Case-control studies are more prone to bias and are less capable at showing a causal relationship.
- Case-control studies work well with rare outcomes, since you choose the outcome yourself.

Structure of clinical data

- Discrete
  - *demographics, vitals, cultures, etc*
- Narrative
  - *admission notes, progress notes, discharge summaries, etc.*
- Images
  - *Xray, MRI, etc.*

Analysis

- Identify outcome, exposure, and potential confounders.
- Perform an unadjusted analysis (Table 1 and Figure 1).
- Perform a fully-adjusted analysis.

Dataset

We have fake patient data (downloaded from EMRBots.org) to illustrate an example of clinical workflow.

- patient_demo.txt
  - *[Patient_ID] - a unique ID representing a patient.*
  - *[PatientGender] - Male/Female.*
  - *[PatientDateOfBirth] - Date Of Birth.*
  - *[PatientRace] - African American, Asian, White, Unknown.*

- patient_encounter.txt
  - *[Patient_ID] - a unique ID representing a patient.*
  - *[Encounter_ID] - an admission ID for the patient.*
  - *[AdmissionStartDate] - start date of encounter.*
  - *[AdmissionEndDate] - end date of encounter.*
- patient_diagnosis.txt
  - *[Patient_ID] - a unique ID representing a patient.*
  - *[Encounter_ID] - an admission ID for the patient.*
  - *[PrimaryDiagnosisCode] - ICD10 code for admission's primary diagnosis.*
  - *[PrimaryDiagnosisDescription] - admission's primary diagnosis description.*

- patients_labs.Rdata
  - *[PatientID] - a unique ID representing a patient.*
  - *[Encounter_ID] - an admission ID for the patient.*
  - *[LabName] - lab's name*
  - *[LabValue] - lab's value*
  - *[LabUnits] - lab's units.*
  - *[LabDateTime] - date.*

Study goal: Identify the risk factors for malignant neoplasm*

We aim to explore the association between diagnosis of malignant neoplasm and certain lab values. The lab values we will look at are: “CBC: WHITE BLOOD CELL COUNT”, “CBC: RED BLOOD CELL COUNT”, “CBC: HEMOGLOBIN”, “CBC: HEMATOCRIT”, “CBC: PLATELET COUNT”, “CBC: ABSOLUTE NEUTROPHILS”, “METABOLIC: ALBUMIN”, “METABOLIC: CALCIUM”, “METABOLIC: SODIUM”, “METABOLIC: POTASSIUM”, “METABOLIC: BILI TOTAL”, “URINALYSIS: PH”.

We will also look at the assocation between patient characteristics and diagnosis of the disease.

Task 1: Getting the outcome.

There are a few options that generally used to determine patient outcome.

- ICD9/10
- Medications/Interventions
- Other data sources

Let's set up our R environment

```
rm(list = ls())
library(plyr)
library(dplyr)
library(lubridate)
```

Let's read in our dataset into a data frame

```
d.dx <- read.csv("~/Google Drive/teaching/2018_CRI_Seminar/data/patient_diagnosis.csv")
names(d.dx)
```

```
## [1] "Patient_ID"                "Encounter_ID"
## [3] "PrimaryDiagnosisCode"      "PrimaryDiagnosisDescription"
```

Most pre-processing can be done using the following commands

- mutate(): To create new variables based on some operation of old variables.
- filter(): To subset a set of rows based on values of a variable.
- select(): To select variables. Also used to remove variables.
- merge(): To combine data frames using common variables.

KEY point : All four use data frames as both input (the first argument) and output.

Using filter()

What if we wanted to look identify patients diagnosed with malignant neoplasm?

Let's choose that subset using the filter() command.

KEY: The filter() function is used to "subset" data, i.e. selecting rows according to a particular condition. In this example, we want to subset d.dx by selecting patients (i.e. the rows) who had malignant neoplasm (i.e. the condition).

The general syntax is

```
data_frame_new <- filter(data_frame_old,
    condition)
```

```r
# get all patients with malignant neoplasm
d.mp <- d.dx %>%
  filter(grepl('Malignant neoplasm', PrimaryDiagnosisDescription))
head(d.mp)
```

```
##   Patient_ID Encounter_ID PrimaryDiagnosisCode
## 1     101009     101009_2                C34.1
## 2     101009     101009_3                  C67
## 3     101407     101407_2                  C33
## 4     101407     101407_5                  C47
## 5     101407     101407_1                C63.0
## 6     105700     105700_1                C14.8
##                                        PrimaryDiagnosisDescription
## 1                    Malignant neoplasm of upper lobe, bronchus or lung
## 2                                      Malignant neoplasm of bladder
## 3                                      Malignant neoplasm of trachea
## 4    Malignant neoplasm of peripheral nerves and autonomic nervous system
## 5                                     Malignant neoplasm of epididymis
## 6 Malignant neoplasm of overlapping sites of lip, oral cavity and pharynx
```

Using pipe

A good way to do combine successive operations using data frames is to use the %>% symbol. Why? Instead of writing multiple lines, you can achieve the same result using single line through the pipe ("%>%") operator.

The general syntax is: output = data_frame %>%

operation_1 %>%
operation_2 %>%
operation_3..

Using select()

If you want to choose columns into another data frame, you can use the select function.

KEY: The select() function is used to choose (or remove) columns of choice. Once again, select() (like filter()) works with data frames. The general syntax is -

```
data_frame_new <- select(data_frame_old,
      c(col1, col2, etc))
```

```r
d.mp_ids <- d.mp %>%
  select(Patient_ID, Encounter_ID) %>% unique()
head(d.mp_ids)
```

```
##   Patient_ID Encounter_ID
## 1     101009     101009_2
## 2     101009     101009_3
## 3     101407     101407_2
## 4     101407     101407_5
## 5     101407     101407_1
## 6     105700     105700_1
```

Note : If you put a - in front of the variable, (i.e. say -c(Patient_ID)), you will REMOVE/DE-SELECT these columns.

There is a difference between patients and patient admissions.

```
cat("Number of admissions with malignant neoplasm",
    d.mp_ids %>% select(Encounter_ID) %>% unique() %>% nrow(), "\n")
```

```
## Number of admissions with malignant neoplasm 4375
```

```
cat("Number of patients with malignant neoplasm",
    d.mp_ids %>% select(Patient_ID) %>% unique() %>% nrow(), "\n")
```

```
## Number of patients with malignant neoplasm 3589
```

```
d.no_mp_ids <- d.dx %>%
  filter(!(Encounter_ID %in% unlist(d.mp_ids$Encounter_ID))) %>%
  select(Patient_ID, Encounter_ID) %>% unique()
cat("Number of admissions without malignant neoplasm",
    d.no_mp_ids %>% select(Encounter_ID) %>% unique() %>% nrow(), "\n")
```

```
## Number of admissions without malignant neoplasm 31768
```

```
d.mp_ids$outcome <- 1
d.no_mp_ids$outcome <- 0
d.mp_outcome <- rbind(d.mp_ids, d.no_mp_ids)
write.csv(d.mp_outcome,
          "~/Google Drive/teaching/2018_CRI_Seminar/results/outcome/mp_outcome.csv",
          row.names = FALSE)
```

Task 2: Put together a descriptive analysis.

Let's begin by putting together clinical characteristics (e.g. age, gender, race, LOS) for each admission.

We will need

- encounter ids of interest
- Age: date of birth and admission start date
- LOS: admissions start and end dates.

Using merge()

merge() is used to combine two datasets based on variables (keys)

Here is a great cheat-sheet for understanding merge() in a greater detail:
http://stat545.com/bit001_dplyr-cheatsheet.html

```
d.enc_info <- read.table("~/Google Drive/teaching/2018_CRI_Seminar/data/patient_encounter.txt",
                         sep = "\t", header = TRUE)
d.mp_outcome <- read.csv("~/Google Drive/teaching/2018_CRI_Seminar/results/outcome/mp_outcome.csv")
d.demo <- read.table("~/Google Drive/teaching/2018_CRI_Seminar/data/patient_demo.txt",
                     sep = "\t", header = TRUE)
```

```
d.cohort <- merge(d.mp_outcome, d.enc_info, by = c("Encounter_ID", "Patient_ID"))
d.cohort <- merge(d.cohort, d.demo, by = c("Patient_ID"))
```

Using mutate()

mutate() is used for creating new variables using a combination of existing variables.

The general syntax is: data_frame_new <- mutate(data_frame_old,
    new_column1 = do_stuff(old_column1),
    new_column2 = do_stuff(old_column2))

Handling date and time

We will use the lubridate() package for this purpose. More details and examples can be found at https://cran.r-project.org/web/packages/lubridate/vignettes/lubridate.html

```
Sys.setenv(tz = "America/Chicago")
d.cohort <- d.cohort %>%
  mutate(AdmissionStartDate = ymd_hms(AdmissionStartDate),
         AdmissionEndDate = ymd_hms(AdmissionEndDate),
         PatientDateOfBirth = ymd_hms(PatientDateOfBirth))
```

We used ymd_hms because the format in this dataset was YYYY-MM-YY HH:MM:SS.

```
head(d.enc_info)
```

```
##   Patient_ID Encounter_ID      AdmissionStartDate        AdmissionEndDate
## 1     109081     109081_2 1983-03-22 05:04:47.540 1983-03-26 04:24:25.987
## 2     109081     109081_3 1997-03-26 20:04:15.043 1997-03-30 13:08:15.633
## 3     109081     109081_5 2004-03-27 01:01:29.530 2004-04-07 14:52:36.153
## 4     109081     109081_6 2006-03-29 08:17:54.907 2006-04-16 22:58:56.287
## 5     109081     109081_4 2003-09-09 04:17:31.027 2003-09-27 08:13:34.593
## 6     109081     109081_1 1981-05-21 05:21:14.380 1981-05-26 11:13:06.313
```

If the format was dd-mm-yy, you would use dmy(). lubridate() can identify a variety of seprators between the date-time components.

Calculating age and LOS.

```
d.cohort <- d.cohort %>%
  mutate(PatientAge = interval(PatientDateOfBirth, AdmissionStartDate) / dyears(1))

d.cohort <- d.cohort %>%
  mutate(LOS = interval(AdmissionStartDate, AdmissionEndDate) / ddays(1))
```

Some take aways

In case you hadn't noticed, the dataset was in a format that was ready to analyze. Notably,

- Every variable was in a separate column with readable column names
- Every observation was in a separate row
- The data frame (generally speaking) contained variables that are consistent with a particular theme. For e.g, patient demographics is different from patient vitals
- The data frame had at least one unique identifier from which it possible to link different tables

The importance of summarizing

Really, you are looking to test the quality of your dataset

- Missing values
- Extreme values
- Consistent units
- Remove things that shouldn't be there in the first place
- NOTE : 80% of your analyses will be prepping the data. dplyr() makes it much easier to do so
- http://seananderson.ca/2014/09/13/dplyr-intro.html

Check for consistency: continous variables

For continous variables, use the quantile() function to check for outliers. The quantile() function will return the 25%, 50%, and 75% quantiles along with max and min. Use ?quantile to study it further.

```
quantile(d.cohort$PatientAge)
```

```
##       0%      25%      50%      75%     100% 
## 18.01184 25.83876 38.58597 54.39640 92.95689
```

Check for consistency: categorical variables

For categorical variables, use the summary() function for counting the number of entries corresponding to a particulay categorical level.

```
summary(d.cohort$PatientRace)
```

```
## African American            Asian          Unknown            White
##             5403             8284             4701            17755
```

```
summary(as.factor(d.cohort$outcome))
```

```
##     0     1
## 31768  4375
```

We can start constructing our Table 1.

```
##  [1] "Patient_ID"          "Encounter_ID"        "outcome"
##  [4] "AdmissionStartDate" "AdmissionEndDate"    "PatientGender"
##  [7] "PatientDateOfBirth" "PatientRace"         "PatientAge"
## [10] "LOS"
```

Variable		Patient admissions with outcome (n=4,375)	Patient admissions without outcome (n=31,768)
Age, mean(sd), yr			
Gender	Male, n (%)		
	Female, n (%)		
LOS, median (IQR)			

Comparison of continous variables

Let's compare age between the two groups

```
d.1 <- filter(d.cohort, outcome == 1)
d.0 <- filter(d.cohort, outcome == 0)
cat("Mean age, outcome 1: ", mean(d.1$PatientAge), "\n")
```

```
## Mean age, outcome 1:  42.06632
```

```
cat("Mean age, outcome 0: ", mean(d.0$PatientAge), "\n")
```

```
## Mean age, outcome 0:  41.70317
```

```
cat("SD age, outcome 1: ", sd(d.1$PatientAge), "\n")
```

```
## SD age, outcome 1:  18.17401
```

```
cat("SD age, outcome 1: ", sd(d.0$PatientAge), "\n")
```

```
## SD age, outcome 1:  18.04217
```

```r
print(t.test(d.1$PatientAge, d.0$PatientAge))
```

```
##
##  Welch Two Sample t-test
##
## data:  d.1$PatientAge and d.0$PatientAge
## t = 1.2402, df = 5627.6, p-value = 0.215
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.2108925  0.9371819
## sample estimates:
## mean of x mean of y
##  42.06632  41.70317
```

Statistical inference has three important concepts

- the null hypothesis
- the alternate hypothesis
- the p-value

For comparing means between two populations

- null: there is no difference in average age between groups
- alternate: there is difference in average age between groups
- the smaller the p-value, the more confident in rejecting the null hypothesis

Step-wise logic

- Assume null is true
- If the data fails to contradict null beyond a reasonable doubt, null is not rejected
- Don’t assume null is true if we don’t reject it (tricky!)
- If not rejected, null is simply a possible explanation for data behavior
- Only when the data contradicts null strongly is the null rejected and the alternative accepted

Comparision of categorical variables

Let's compare the gender variable with respect to our outcome.

First let's build a 2x2 table.

```
gender.table <- with(d.cohort, table(outcome, PatientGender))
gender.table
```

```
##         PatientGender
## outcome Female  Male
##       0  16584 15184
##       1   2292  2083
```

Chi-squared testing Null: Outcome is not associated with gender

Alternative : Outcome is associated with gender

P-value: the smaller the p-value, the more confident in rejecting the null hypothesis

Looking at the result below, we can say that no association was observed between outcome and gender.

```
chisq.test(gender.table)
```

```
##
##  Pearson's Chi-squared test with Yates' continuity correction
##
## data:  gender.table
## X-squared = 0.045645, df = 1, p-value = 0.8308
```

Comparing medians

For variables that are not distributed normally (e.g. length of stay, which is skewed), we compare the median and the inter-quantile range (IQR). In R, we use median() and quantile() to get these values. Statiscal comparison between groups is done by the Mood test (in R this is mood.test()). Note that comparing means in skewed distributions is also done using non-parametric tests such as Wilcoxon rank sum test (wilcox.test() in R). For further details, see https://www.r-bloggers.com/example-2014-6-comparing-medians-and-the-wilcoxon-rank-sum-test/.

Table 1

We now have everything we need to create table 1.

```
##  [1] "Patient_ID"          "Encounter_ID"        "outcome"
##  [4] "AdmissionStartDate" "AdmissionEndDate"    "PatientGender"
##  [7] "PatientDateOfBirth" "PatientRace"         "PatientAge"
## [10] "LOS"
```

Variable		Patient admissions with outcome (n=4,375)	Patient admissions without outcome (n=31,768)
Age, mean(sd), yr		42 (18)	42(18)
Gender	Male, n (%)	2,083 (48)	15,184 (48)
	Female, n (%)	2,292 (52)	16,584 (52)
LOS, median (IQR), days		11 (6,15)	11(6, 16)

Task 3: Compiling the variables of interest.

We create a feature matrix that has the most-recent lab values associated with the encounter along with the outcome. This has already been compiled and given.

```
# this will load up a data frame called d.enc_labs
load("~/Google Drive/teaching/2018_CRI_Seminar/results/features/mp_most_recent_labs.RData")

# Merge with outcome to get a feature matrix
d.features <- merge(d.cohort, d.enc_labs, by = c("Encounter_ID"))
```

```r
# some cleaning
d.features <- d.features %>%
  select(-c(Encounter_ID, Patient_ID, AdmissionStartDate, AdmissionEndDate, PatientDateOfBirth,
            LabDateTime))
names(d.features)
```

```
##  [1] "outcome"                       "PatientGender"
##  [3] "PatientRace"                   "PatientAge"
##  [5] "LOS"                           "CBC: ABSOLUTE NEUTROPHILS"
##  [7] "CBC: HEMATOCRIT"               "CBC: HEMOGLOBIN"
##  [9] "CBC: PLATELET COUNT"           "CBC: RED BLOOD CELL COUNT"
## [11] "CBC: WHITE BLOOD CELL COUNT"   "METABOLIC: ALBUMIN"
## [13] "METABOLIC: BILI TOTAL"         "METABOLIC: CALCIUM"
## [15] "METABOLIC: POTASSIUM"          "METABOLIC: SODIUM"
## [17] "URINALYSIS: PH"
```

```r
# save work
save(list = c("d.features"),
     file = "~/Google Drive/teaching/2018_CRI_Seminar/results/features/mp_study_features.RData")
```

Task 4: Regression

In association studies, we want to understand the relationship between and exposure and outcome. We do this sequentially:

- outcome ~ exposure (called as unadjusted analysis)
- outcome ~ exposure + confounders (called as adjusted analysis)

The idea is to see if the relationship persists after adjustment of confounders.

The choice of linear or logistic regression depends on the outcome.

- if outcome is continous, perform linear regression
- if outcome is binary, perform logistic regression

Linear Regression

The OLS mode for linear regression takes the form:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_p X_p$$

We know that
- Y is the response/outcome
- all X_i are predictors/variables/features
- β's are parameters/model coefficients/weights and are estimated using least squares

β_0 is the intercept, which is Y when all continous predictors are 0 and all categorical predictors are set to reference.

For every unit increase in X_i, the response Y changes by β_i.

For example, consider modeling price of car (in $1000) against years from purchase (in years)

$$car\_price = \beta_0 + \beta_1 years\_from\_purchase$$

Let $\beta_0 = 10$ and $\beta_1 = -0.95$.

The interpretation is

- at year of purchase the car price was $10,000.
- for every additional year, the car price will go down by $950.

Logistic Regression

The predictor Y is binary (i.e. 0 and 1). Consider a simple model with response Y and a single predictor X.

In logistic regression, we look at the conditional probability of Y being 1 given X.

$$P(Y = 1|X)$$

Odds

$$Odds = \frac{Probability\ of\ event}{1 - Probability\ of\ event}$$

Odds and probability are not the same.

Given a scenario where the mortality rate for an admitting patient in 20%, what are the odds that a patient will die?

Prob(Death) = 0.2

Odds(Death) = 0.2/0.8 = 1/4 = 0.25

For every patient who dies, there are four patients who will survive.

Consider the odds for Y being 1 for a single variable model.

$$Odds = \frac{P(Y = 1|X)}{1 - P(Y = 1|X)}$$

In logistic regression,

$$log(Odds) = \beta_0 + \beta_1 X$$

For multiple predictors,

$$log\frac{P(Y=1|X)}{1-P(Y=1|X)} = \beta_0 + \beta_1 X_1 + \beta_2 X_2 \ldots \beta_p X_p$$

If $\beta_j > 0$, then $exp(\beta_j) > 1$, and the odds increase.

If $\beta_j < 0$, then $exp(\beta_j) < 1$, and the odds decrease.

The p-value of β_i will indicate the signficance of that coefficient.

In order to correctly interpret the model, you have to look at both the p-value and the OR

Suppose we are interested in patient mortality from trauma patients who suffered out-of-hospital cardiac arrests.

What is the response?

What is the predictor?

Hypothetical example

Suppose we are interested in patient mortality from trauma patients who suffered out-of-hospital cardiac arrests.

What is the response?

- Y is patient dying in the hospital (1 = Yes, 0 = No)

What is the predictor?

- X whether a trauma patient had a out-of-hospital cardiac arrests. (1 = Yes, 0 = No)

Hypothetical example

Suppose we are interested in patient mortality from trauma patients who suffered out-of-hospital cardiac arrests.

We perform a logistic regression, and we get β_1 = 0.18, p-value < 0.001

This means that
- (a) the log-odds of death increases by 0.18 when a patient comes in with out-of-hospital cardiac arrest

Hypothetical example

Suppose we are interested in in-hospital patient mortality from incoming trauma where patients suffer out-of-hospital cardiac arrests.

We perform a logistic regression, and we get β_1 = 0.18, p-value < 0.001,

Odds ratio = exp(0.18) = 1.20 (95%CI: 1.14, 1.30)

This means that
- (a) the log-odds of death increases by 0.18 when a patient comes in with out-of-hospital cardiac arrest
- (b) the odds-ratio increases by 1.2 when a patient comes in with out-hosptial cardiac arrest.

Hypothetical example

Suppose we are interested in in-hospital patient mortality from incoming trauma where patients suffer out-of-hospital cardiac arrests.

We perform a logistic regression, and we get β_1 = 0.18, p <0.001

Odds ratio = exp(0.18) = 1.20 (95%CI: 1.14, 1.30)

This means that for an incoming trauma that is from a out-of-hospital cardiac arrest, likelihood of patient dying in the hospital increases by 20%.

Hypothetical example

Suppose we are interested in in-hospital patient mortality from incoming trauma where patients suffer out-of-hospital cardiac arrests.

We perform a logistic regression, and we get β_1 = 0.18, and the p-value is not signficant (i.e. > 0.001)

Odds ratio = exp(0.18) = 1.20 (95% CI: 0.89, 1.70)

This means that no significant associations can be drawn from this study.

Task 4: Regression

```
m1 <- glm(outcome ~ ., data = d.features, family = "binomial")
```

```r
summary(m1)
```

```
##
## Call:
## glm(formula = outcome ~ ., family = "binomial", data = d.features)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.5766  -0.5177  -0.5043  -0.4892   2.1608
##
## Coefficients:
##                                Estimate Std. Error z value Pr(>|z|)
## (Intercept)                   -1.8703611  0.4187966  -4.466 7.97e-06 ***
## PatientGenderMale             -0.0117749  0.0325666  -0.362  0.71768
## PatientRaceAsian               0.0513524  0.0547992   0.937  0.34871
## PatientRaceUnknown             0.0355191  0.0625141   0.568  0.56991
## PatientRaceWhite               0.0759927  0.0487248   1.560  0.11885
## PatientAge                     0.0009372  0.0008968   1.045  0.29600
## LOS                           -0.0047828  0.0031642  -1.512  0.13065
## `CBC: ABSOLUTE NEUTROPHILS`   -0.0001219  0.0028179  -0.043  0.96548
## `CBC: HEMATOCRIT`             -0.0003363  0.0022647  -0.148  0.88196
## `CBC: HEMOGLOBIN`             -0.0061471  0.0062494  -0.984  0.32529
## `CBC: PLATELET COUNT`          0.0001796  0.0001704   1.054  0.29183
## `CBC: RED BLOOD CELL COUNT`   -0.0241488  0.0140585  -1.718  0.08585 .
## `CBC: WHITE BLOOD CELL COUNT` -0.0036096  0.0062556  -0.577  0.56393
## `METABOLIC: ALBUMIN`          -0.0118067  0.0160323  -0.736  0.46147
## `METABOLIC: BILI TOTAL`        0.0106724  0.0465493   0.229  0.81866
## `METABOLIC: CALCIUM`           0.0023245  0.0112481   0.207  0.83628
## `METABOLIC: POTASSIUM`         0.0062358  0.0187237   0.333  0.73910
## `METABOLIC: SODIUM`           -0.0017143  0.0018733  -0.915  0.36013
## `URINALYSIS: PH`               0.0490157  0.0187973   2.608  0.00912 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 26266  on 35591  degrees of freedom
## Residual deviance: 26246  on 35573  degrees of freedom
##   (551 observations deleted due to missingness)
## AIC: 26284
##
## Number of Fisher Scoring iterations: 4
```

```
##                         variable      p.value        OR     OR_2.5
## 1                    (Intercept) 7.968199e-06 0.1540680 0.06777298
## 2    `CBC: ABSOLUTE NEUTROPHILS` 9.654819e-01 0.9998781 0.99437070
## 3              `CBC: HEMATOCRIT` 8.819557e-01 0.9996638 0.99523621
## 4              `CBC: HEMOGLOBIN` 3.252950e-01 0.9938717 0.98177113
## 5          `CBC: PLATELET COUNT` 2.918346e-01 1.0001796 0.99984570
## 6    `CBC: RED BLOOD CELL COUNT` 8.584525e-02 0.9761404 0.94960526
## 7  `CBC: WHITE BLOOD CELL COUNT` 5.639268e-01 0.9963969 0.98425419
## 8           `METABOLIC: ALBUMIN` 4.614668e-01 0.9882627 0.95768775
## 9        `METABOLIC: BILI TOTAL` 8.186576e-01 1.0107296 0.92259621
## 10          `METABOLIC: CALCIUM` 8.362789e-01 1.0023272 0.98047136
## 11        `METABOLIC: POTASSIUM` 7.391031e-01 1.0062552 0.96999414
## 12           `METABOLIC: SODIUM` 3.601293e-01 0.9982872 0.99462812
## 13              `URINALYSIS: PH` 9.118144e-03 1.0502368 1.01225511
## 14                          LOS 1.306524e-01 0.9952286 0.98907451
## 15                   PatientAge 2.960048e-01 1.0009377 0.99917528
## 16            PatientGenderMale 7.176778e-01 0.9882942 0.92715513
## 17             PatientRaceAsian 3.487060e-01 1.0526938 0.94579455
## 18           PatientRaceUnknown 5.699140e-01 1.0361575 0.91655239
## 19             PatientRaceWhite 1.188467e-01 1.0789547 0.98130392
##      OR_97.5
## 1  0.3499829
## 2  1.0054159
## 3  1.0041112
## 4  1.0061201
## 5  1.0005136
## 6  1.0034084
## 7  1.0086895
## 8  1.0198090
## 9  1.1072907
## 10 1.0246719
## 11 1.0438694
## 12 1.0019592
## 13 1.0896633
## 14 1.0014195
## 15 1.0026943
## 16 1.0534104
```

```
## 17 1.1724776
## 18 1.1711326
## 19 1.1878658
```

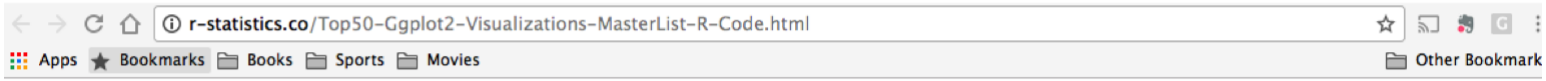
Things that could have gone wrong.

- We chose the most-recent lab for that encounter.
- We chose the lab values within the same encounter that was diagnosed with the condition.
- We chose patient admissions vs. patients.

Other topics

Visualization

- ggplot2 : http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html



Top 50 ggplot2 Visualizations - The Master List (With Full R Code)

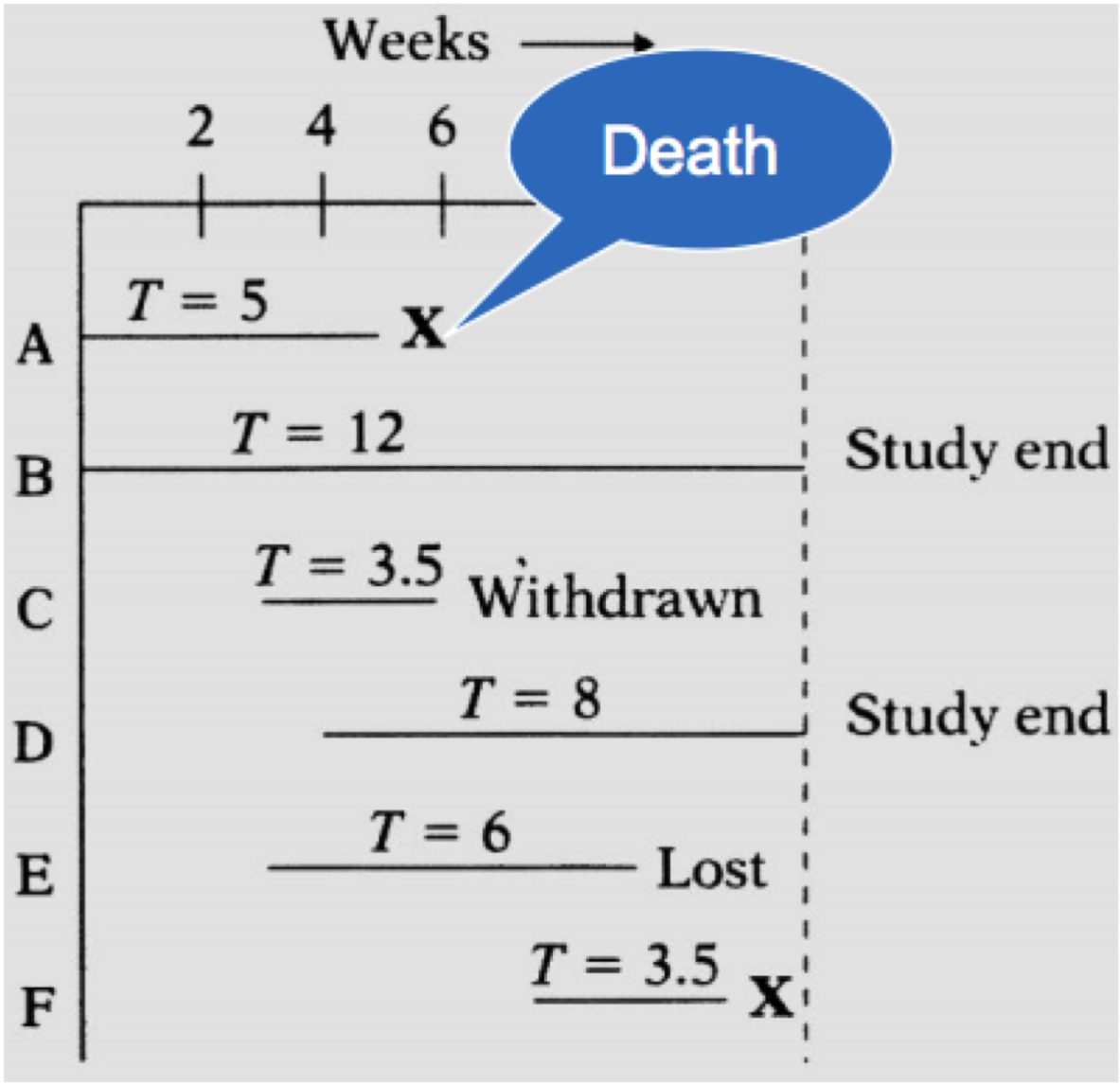
> What type of visualization to use for what sort of problem? This tutorial helps you choose the right type of chart for your specific objectives and how to implement it in R using ggplot2.

This is part 3 of a three part tutorial on ggplot2, an aesthetically pleasing (and very popular) graphics framework in R. This tutorial is primarily geared towards those having some basic knowledge of the R programming language and want to make complex and nice looking charts with R ggplot2.

- Part 1: Introduction to ggplot2, covers the basic knowledge about constructing simple ggplots and modifying the components and aesthetics.
- Part 2: Customizing the Look and Feel, is about more advanced customization like manipulating legend, annotations, multiplots with faceting and custom layouts
- Part 3: Top 50 ggplot2 Visualizations - The Master List, applies what was learnt in part 1 and 2 to construct other types of ggplots such as bar charts, boxplots etc.

Survival Analysis

- If you want to model time-to-event (such as death) on censored data.



Sun

SIGKDD 2013

- We want to model the probability that an observation can survive after a time point t.
- We calculate the hazard function, which is simply the probability that the event will occur in the next instant, given survival till time point t.
- Cox Proportional-Hazard model: estimate the effects of your variables/covariates on surivival.
- use survival() package in R

Prediction

- Sensitivity/Specificity/Type I error
- Receiver Operating characteristic (ROC), Area under the Curve (AUC)
- Training/Testing/Cross-validation
- Machine learning models
  - *Logistic Regression*
  - *Decision Trees/Random Forests*
  - *Support Vector Machines*
  - *Artifical neural network*
  - *Deep learning*