Code for QSS tidyverse Chapter 6: Probability

Kosuke Imai and Nora Webb Williams

First Printing

Probability

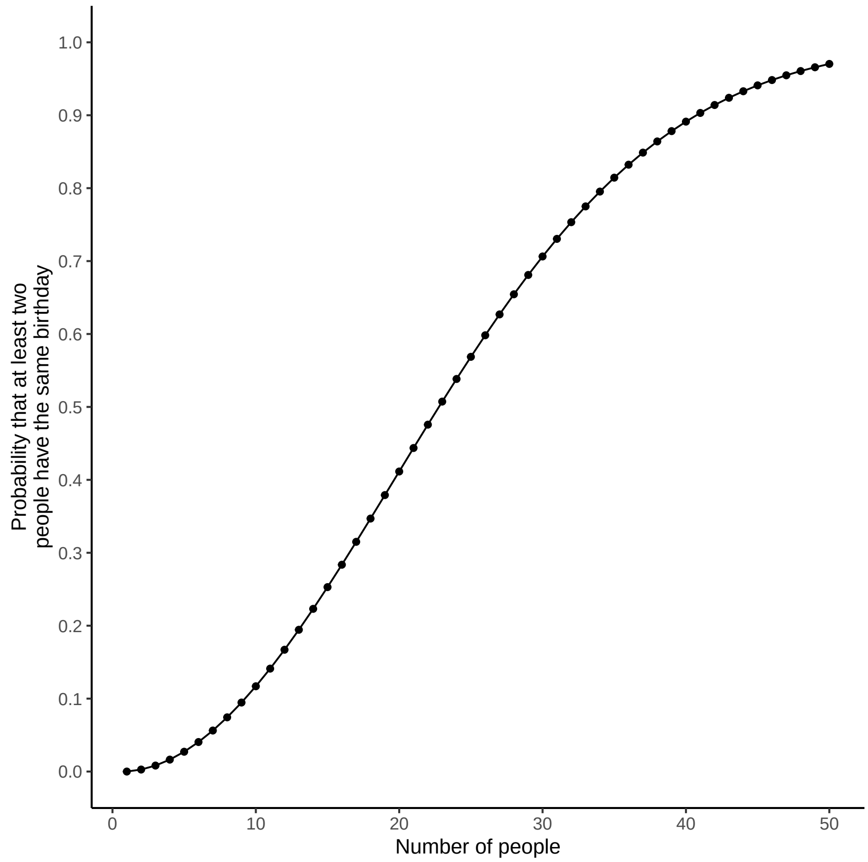
Probability

Frequentist vs. Bayesian

Definition and Axioms

Permutations

```
library(tidyverse)
## write the birthday function
birthday <- function(k) {
  logdenom <- k * log(365) + lfactorial(365 - k)
  lognumer <- lfactorial(365)
  pr <- 1 -   exp(lognumer - logdenom)
  pr
}

## create a tibble with the k and pr per k
bday <- tibble(k = 1:50, pr = birthday(k))

## plot the data
ggplot(bday, aes(x = k, y = pr)) +
  geom_line() +
  geom_point() +
  scale_y_continuous(str_c("Probability that at least two",
                           "people have the same birthday", sep = "\n"),
                     limits = c(0, 1), breaks = seq(0, 1, by = 0.1)) +
  labs(x = "Number of people")
```



Sampling with and without Replacement

```
## setting seed for replication
set.seed(4444)
k <- 23 # number of people
sims <- 1000 # number of simulations
event <- 0 # counter
for (i in 1:sims) {
    days <- sample(1:365, k, replace = TRUE)
    days.unique <- unique(days) # unique birthdays
    ## if there are duplicates, the number of unique birthdays
    ## will be less than the number of birthdays, which is `k'
    if (length(days.unique) < k) {
        event <- event + 1
    }
}
## fraction of trials where at least two bdays are the same
answer <- event / sims
answer
```

```
## [1] 0.511
```

Combinations

```
choose(84, 6)
```

```
## [1] 406481544
```

Conditional Probability

Conditional, Marginal, and Joint Probabilities

```
data(FLVoters, package = "qss")
## how many observations?
dim(FLVoters)
```

```
## [1] 10000     6
```

```
## what do the data look like?
glimpse(FLVoters)
```

```
## Rows: 10,000
## Columns: 6
## $ surname <chr> "PIEDRA", "LYNCH", "CHESTER", "LATHROP", "~
## $ county  <int> 115, 115, 115, 115, 115, 115, 115, 115, 1,~
## $ VTD     <int> 66, 13, 103, 80, 8, 55, 84, 48, 41, 39, 26~
## $ age     <int> 58, 51, 63, 54, 77, 49, 77, 34, 56, 60, 44~
## $ gender  <chr> "f", "m", "m", "m", "f", "m", "f", "f", "f~
## $ race    <chr> "white", "white", NA, "white", "white", "w~
```

```
## removing observations with missing values
FLVoters <- FLVoters %>%
  na.omit()
## how many observations remain?
dim(FLVoters)
```

```
## [1] 9113    6
```

```
margin_race <-
  FLVoters %>%
  count(race) %>%
  mutate(prop = n / sum(n))
margin_race
```

```
##       race    n        prop
## 1    asian  175 0.019203336
## 2    black 1194 0.131021617
## 3 hispanic 1192 0.130802151
## 4   native   29 0.003182267
## 5    other  310 0.034017338
## 6    white 6213 0.681773291
```

```r
margin_gender <-
  FLVoters %>%
  count(gender) %>%
  mutate(prop = n / sum(n))
margin_gender
```

```
##   gender    n      prop
## 1      f 4883 0.5358279
## 2      m 4230 0.4641721
```

```r
## Conditional probability, among women
margin_race_f <- FLVoters %>%
  filter(gender == "f") %>%
  count(race) %>%
  mutate(prop = n / sum(n))
margin_race_f
```

```
##       race    n        prop
## 1    asian   83 0.016997747
## 2    black  678 0.138849068
## 3 hispanic  666 0.136391563
## 4   native   17 0.003481466
## 5    other  158 0.032357157
## 6    white 3281 0.671922998
```

```r
## Conditional probability, among men
margin_race_m <- FLVoters %>%
  filter(gender == "m") %>%
  count(race) %>%
  mutate(prop = n / sum(n))
margin_race_m
```

```
##       race    n        prop
## 1    asian   92 0.021749409
## 2    black  516 0.121985816
## 3 hispanic  526 0.124349882
## 4   native   12 0.002836879
## 5    other  152 0.035933806
## 6    white 2932 0.693144208
```

```r
joint_p <-
  FLVoters %>%
  count(gender, race) %>%
  mutate(prop = n / sum(n))
joint_p
```

```
##    gender     race    n        prop
## 1       f    asian   83 0.009107868
## 2       f    black  678 0.074399210
## 3       f hispanic  666 0.073082410
## 4       f   native   17 0.001865467
```

```
## 5        f    other  158 0.017337869
## 6        f    white 3281 0.360035115
## 7        m    asian   92 0.010095468
## 8        m    black  516 0.056622408
## 9        m hispanic  526 0.057719741
## 10       m   native   12 0.001316800
## 11       m    other  152 0.016679469
## 12       m    white 2932 0.321738176
```

```
## gender to columns, with proportion as value
joint_p_wider <- joint_p %>%
  select(-n) %>%
  pivot_wider(names_from = gender,
              values_from = prop) %>%
  mutate(total_prop = f + m)
```

```
## race to columns, with proportion as value
joint_p_wider <- joint_p %>%
  select(-n) %>%
  pivot_wider(names_from = race,
              values_from = prop) %>%
  mutate(total_prop = rowSums(across(where(is.numeric))))

joint_p_wider
```

```
## # A tibble: 2 x 8
##   gender   asian  black hispanic  native  other white
##   <chr>    <dbl>  <dbl>    <dbl>   <dbl>  <dbl> <dbl>
## 1 f      0.00911 0.0744   0.0731 0.00187 0.0173 0.360
## 2 m      0.0101  0.0566   0.0577 0.00132 0.0167 0.322
## # ... with 1 more variable: total_prop <dbl>
```

```
## alternative
joint_p %>%
  group_by(gender) %>%
  summarize(prop = sum(prop))
```

```
## # A tibble: 2 x 2
##   gender  prop
##   <chr>  <dbl>
## 1 f      0.536
## 2 m      0.464
```

```
## adding the age_group variable
FLVoters <- FLVoters %>%
  mutate(age_group = cut(age, breaks = c(0, 20, 40, 60, Inf),
                         right = TRUE,
                         labels = c("<= 20", "20-40", "40-60", "> 60")))
```

```
## joint probability table
joint3 <-
  FLVoters %>%
```

```
  count(race, age_group, gender) %>%
  mutate(prop = n / sum(n))

head(joint3)
```

```
##    race age_group gender  n         prop
## 1 asian     <= 20      f  1 0.0001097333
## 2 asian     <= 20      m  2 0.0002194667
## 3 asian     20-40      f 24 0.0026336004
## 4 asian     20-40      m 26 0.0028530670
## 5 asian     40-60      f 38 0.0041698672
## 6 asian     40-60      m 47 0.0051574674
```

```
## calculate marginal probability of age groups
margin_age <-
  FLVoters %>%
  count(age_group) %>%
  mutate(margin_age = n / sum(n)) %>%
  select(-n)
margin_age
```

```
##   age_group margin_age
## 1     <= 20 0.01766707
## 2     20-40 0.27093164
## 3     40-60 0.36047405
## 4      > 60 0.35092725
```

```
## merge this with the joint probability table
## and add conditional prob
joint3 <- left_join(joint3, margin_age,
                    by = "age_group") %>%
  mutate(con_prob_age = prop / margin_age)

## conditional probability of black female given
## above 60 years old

filter(joint3, race == "black", gender == "f", age_group == "> 60") %>%
  select(race, age_group, gender, con_prob_age)
```

```
##    race age_group gender con_prob_age
## 1 black      > 60      f   0.05378361
```

```
## joint probability by age and gender
joint2 <- FLVoters %>%
  count(age_group, gender) %>%
  ungroup() %>%
  mutate(prob_age_gender = n / sum(n)) %>%
  select(-n)
joint2
```

```
##   age_group gender prob_age_gender
```

```
## 1     <= 20      f     0.009656535
## 2     <= 20      m     0.008010534
## 3     20-40      f     0.143092286
## 4     20-40      m     0.127839350
## 5     40-60      f     0.189838692
## 6     40-60      m     0.170635356
## 7      > 60      f     0.193240426
## 8      > 60      m     0.157686821
```

```
## merge to the 3 way joint probability
## calculate conditional prob of race given age and gender
joint3 <- left_join(joint3, joint2,
                    by = c("age_group", "gender")) %>%
  mutate(con_prob_race = prop / prob_age_gender)

## Conditional prob of black given female and above 60
filter(joint3, gender == "f", age_group == "> 60", race == "black") %>%
  select(con_prob_race)
```

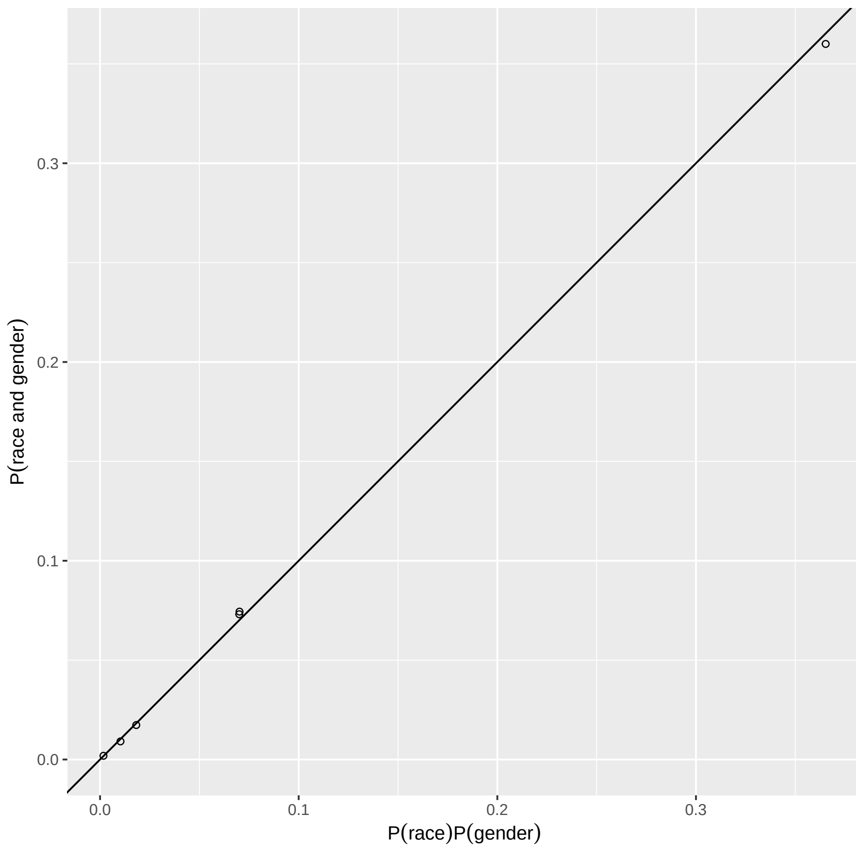
```
##   con_prob_race
## 1    0.09767178
```

Independence

```
## adjust the marginal probability data: select/rename needed variables
margin_race <- select(margin_race, race, prob_race = prop)
margin_gender <- select(margin_gender, gender, prob_gender = prop)

## combine the marginal probabilities and link to joint probabilities
race_gender_indep <- tidyr::crossing(margin_race, margin_gender) %>%
  left_join(select(joint_p, gender, race, prob_joint = prop),
            by = c("gender", "race")) %>%
  ## calculate the independent prob comparison
  mutate(prob_indp = prob_race * prob_gender)

race_gender_indep
```

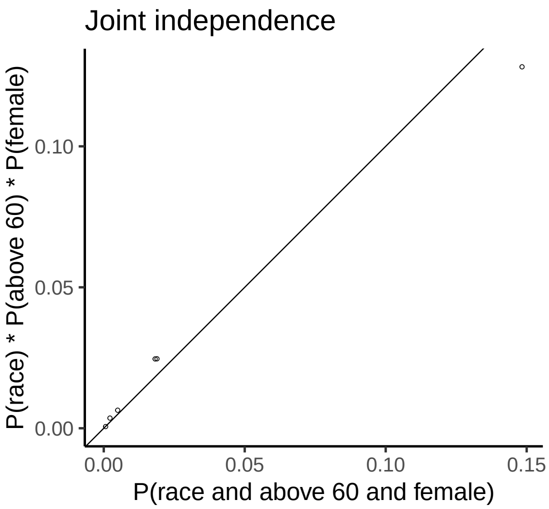
```
## # A tibble: 12 x 6
##    race    prob_race gender prob_gender prob_joint prob_indp
##    <chr>       <dbl> <chr>        <dbl>      <dbl>     <dbl>
##  1 asian     0.0192  f            0.536    0.00911   0.0103
##  2 asian     0.0192  m            0.464    0.0101    0.00891
##  3 black     0.131   f            0.536    0.0744    0.0702
##  4 black     0.131   m            0.464    0.0566    0.0608
##  5 hispan~   0.131   f            0.536    0.0731    0.0701
##  6 hispan~   0.131   m            0.464    0.0577    0.0607
##  7 native    0.00318 f            0.536    0.00187   0.00171
##  8 native    0.00318 m            0.464    0.00132   0.00148
##  9 other     0.0340  f            0.536    0.0173    0.0182
## 10 other     0.0340  m            0.464    0.0167    0.0158
## 11 white     0.682   f            0.536    0.360     0.365
## 12 white     0.682   m            0.464    0.322     0.316
```

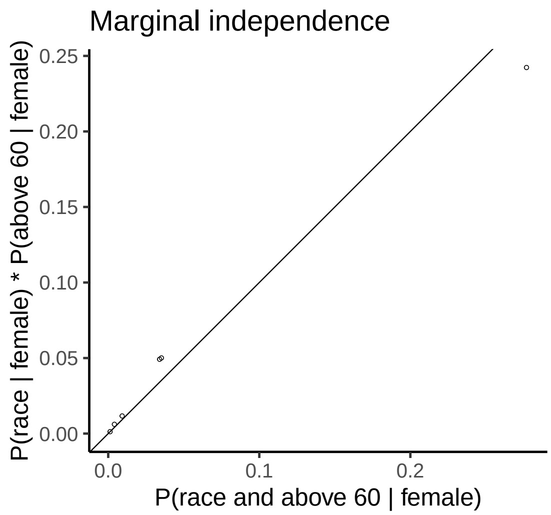
```r
## plot it, just for female
ggplot(filter(race_gender_indep, gender == "f"),
       aes(x = prob_indp, y = prob_joint)) +
  geom_abline(intercept = 0, slope = 1, color = "black", size = 0.5) +
  geom_point(shape = 1) +
  coord_fixed() +
  labs(x = expression(P("race") * P("gender")),
       y = expression(P("race and gender")))
```




```r
## adjust the marginal probability data: select/rename needed variables
margin_age <- select(margin_age, age_group, prob_age = margin_age)

## combine the marginal probabilities and link to joint probabilities
joint_indep <-
  tidyr::crossing(margin_age, margin_gender, margin_race) %>%
  mutate(indep_prob = prob_race * prob_age * prob_gender) %>%
  left_join(select(joint3, race, age_group, gender, prob_joint = prop),
            by = c("gender", "age_group", "race"))

## Plot joint versus independent prob
ggplot(filter(joint_indep, age_group == "> 60", gender == "f"),
       aes(x = prob_joint, y = indep_prob)) +
  geom_abline(intercept = 0, slope = 1, color = "black", size = 0.5) +
  geom_point(shape = 1) +
```

```r
  coord_fixed() +
  labs(x = "P(race and above 60 and female)",
       y = "P(race) * P(above 60) * P(female)",
       title = "Joint independence")

## conditional prob of race and age given gender
cond_prob_gender <- left_join(select(joint3, race, age_group, gender,
                                     joint_prob = prop),
                              margin_gender,
            by = c("gender")) %>%
  mutate(cond_prob_gender = joint_prob / prob_gender)

## conditional prob of race given gender
cond_prob_race_gender <- left_join(select(joint_p, race, gender,
                                          prob_race_gender = prop),
                                   margin_gender,
            by = "gender") %>%
  mutate(cond_prob_race_gender = prob_race_gender / prob_gender) %>%
  select(race, gender, cond_prob_race_gender)

## conditional prob of age given gender
cond_prob_age_gender <- left_join(select(joint2, age_group,
                                         gender, prob_age_gender),
                                  margin_gender,
            by = "gender") %>%
  mutate(cond_prob_age_gender = prob_age_gender / prob_gender) %>%
  select(age_group, gender, cond_prob_age_gender)

# independent probability of race and age
indep_cond_gender <- full_join(cond_prob_race_gender, cond_prob_age_gender,
            by = "gender") %>%
  mutate(indep_prob = cond_prob_race_gender * cond_prob_age_gender)

## Merge the independent probability with the conditional probability
master <- left_join(cond_prob_gender, indep_cond_gender,
                   by = c("gender", "age_group", "race"))

## plotting just for women over 60
ggplot( filter(master, age_group == "> 60", gender == "f"),
  aes(x = cond_prob_gender, y = indep_prob)) +
  geom_abline(intercept = 0, slope = 1, color = "black", size = 0.5) +
  geom_point(shape = 1) +
  coord_fixed() +
  labs(x = "P(race and above 60 | female)",
       y = "P(race | female) * P(above 60 | female)",
       title = "Marginal independence")
```






```r
## create a function that simulates the game
sims <- 1000
doors <- c("goat", "goat", "car")
result.switch <- result.noswitch <- rep(NA, sims)

for (i in 1:sims) {
    ## randomly choose the initial door
    first <- sample(1:3, size = 1)
    result.noswitch[i] <- doors[first]
    remain <- doors[-first] # remaining two doors
    ## Monty chooses one door with a goat
    if (doors[first] == "car") # two goats left
        monty <- sample(1:2, size = 1)
    else # one goat and one car left
        monty <- (1:2)[remain == "goat"]
    result.switch[i] <- remain[-monty]
}

mean(result.noswitch == "car")
```

```
## [1] 0.339
```

```r
mean(result.switch == "car")
```

```
## [1] 0.661
```

Bayes' Rule

Predicting Race Using Surname and Residence Location

```r
data("cnames", package = "qss")
glimpse(cnames)
```

```
## Rows: 151,671
## Columns: 7
## $ surname     <chr> "SMITH", "JOHNSON", "WILLIAMS", "BROWN~
## $ count       <int> 2376206, 1857160, 1534042, 1380145, 13~
## $ pctwhite    <dbl> 73.342666, 61.550000, 48.520000, 60.71~
## $ pctblack    <dbl> 22.217778, 33.800000, 46.720000, 34.54~
## $ pctapi      <dbl> 0.399960, 0.420000, 0.370000, 0.410041~
## $ pcthispanic <dbl> 1.559844, 1.500000, 1.600000, 1.640164~
## $ pctothers   <dbl> 2.479752, 2.730000, 2.790000, 2.690269~
```

```
## Finding the most likely race per name
most_likely_race <- cnames %>%
  select(-count) %>%
  ## reshape to longer
  pivot_longer(cols = -surname,
               names_to = "race_pred",
               values_to = "highest_pct") %>%
  # remove pct prefix from variable names
  mutate(race_pred = str_replace(race_pred, "pct", "")) %>%
  ## group by surname
  group_by(surname) %>%
  ## select row per name with the largest percentage
  ## by keeping the first instance after arranging by percentage
  filter(row_number(desc(highest_pct)) == 1) %>%
  # Ungroup to avoid errors later
  ungroup()

## merging back with the original data
cnames <- full_join(cnames, most_likely_race, by = "surname")
```

```
## Size of the voter file
dim(FLVoters)
```

```
## [1] 9113    7
```

```
## Merge with the census data
FLVotersJoin <- FLVoters %>%
  inner_join(cnames, by = "surname")
## Size after matching (smaller, some names not matched)
dim(FLVotersJoin)
```

```
## [1] 8022   15
```

```
glimpse(FLVotersJoin)
```

```
## Rows: 8,022
## Columns: 15
## $ surname     <chr> "PIEDRA", "LYNCH", "LATHROP", "HUMMEL"~
## $ county      <int> 115, 115, 115, 115, 115, 115, 1, 1, 11~
## $ VTD         <int> 66, 13, 80, 8, 55, 84, 41, 39, 26, 45,~
## $ age         <int> 58, 51, 54, 77, 49, 77, 56, 60, 44, 45~
## $ gender      <chr> "f", "m", "m", "f", "m", "f", "f", "m"~
```

```
## $ race        <chr> "white", "white", "white", "white", "w~
## $ age_group   <fct> 40-60, 40-60, 40-60, > 60, 40-60, > 60~
## $ count       <int> 3518, 114448, 7936, 14878, 812, 9355, ~
## $ pctwhite    <dbl> 6.71000, 84.22000, 93.39066, 97.21000,~
## $ pctblack    <dbl> 1.190000, 11.230000, 1.779822, 0.13000~
## $ pctapi      <dbl> 0.430000, 0.430000, 0.779922, 0.470000~
## $ pcthispanic <dbl> 91.390000, 1.680000, 1.879812, 1.22000~
## $ pctothers   <dbl> 0.280000, 2.440000, 2.169783, 0.970000~
## $ race_pred   <chr> "hispanic", "white", "white", "white",~
## $ highest_pct <dbl> 91.39000, 84.22000, 93.39066, 97.21000~
```

```
## which values for race and race_pred?
unique(FLVotersJoin$race)
```

```
## [1] "white"    "other"    "black"    "asian"    "hispanic"
## [6] "native"
```

```
unique(FLVotersJoin$race_pred)
```

```
## [1] "hispanic" "white"    "black"    "others"   "api"
```

```
## Recoding so the fields match
FLVotersJoin <- FLVotersJoin %>%
  mutate(race = recode(race, "native" = "other"),
         race_pred = recode(race_pred,
                            "api" = "asian",
                            "others" = "other"))

## check that the recoding worked
unique(FLVotersJoin$race)
```

```
## [1] "white"    "other"    "black"    "asian"    "hispanic"
```

```
unique(FLVotersJoin$race_pred)
```

```
## [1] "hispanic" "white"    "black"    "other"    "asian"
```

```
race_tp <- FLVotersJoin %>%
  group_by(race) %>%
  summarize(tp = mean(race == race_pred)) %>%
  arrange(desc(tp))

race_tp
```

```
## # A tibble: 5 x 2
##   race          tp
##   <chr>      <dbl>
## 1 white    0.950
## 2 hispanic 0.847
## 3 asian    0.564
## 4 black    0.160
## 5 other    0.00361
```

```r
race_fp <- FLVotersJoin %>%
  group_by(race_pred) %>%
  summarize(fp = mean(race != race_pred)) %>%
  arrange(desc(fp))

race_fp
```

```
## # A tibble: 5 x 2
##   race_pred    fp
##   <chr>     <dbl>
## 1 other     0.857
## 2 asian     0.342
## 3 black     0.329
## 4 hispanic  0.227
## 5 white     0.197
```

```r
data("FLCensus", package = "qss")

## recode the race variable
FLCensus <- FLCensus %>%
    rename("asian" = "api",
           "other" = "others")

## probability of race in Florida
race_prop <- FLCensus %>%
  select(total.pop, white, black, asian, hispanic, other) %>%
  pivot_longer(cols = - total.pop,
               names_to = "race",
               values_to = "pct") %>%
  group_by(race) %>%
  summarize(race_prop = weighted.mean(pct, w = total.pop)) %>%
  arrange(desc(race_prop))
race_prop
```

```
## # A tibble: 5 x 2
##   race     race_prop
##   <chr>        <dbl>
## 1 white       0.579
## 2 hispanic    0.225
## 3 black       0.152
## 4 asian       0.0242
## 5 other       0.0206
```

```r
## merge the race probability with the existing data
FLVotersJoin <- left_join(FLVotersJoin, race_prop, by = "race")

## Calculate prob of surname given race
## total number of individuals
total.count <- sum(cnames$count)

## have to reshape the names data to longer format, similar to above
cnames_reshape <- cnames %>%
```

```r
  ## drop unneeded columns
  select(-race_pred, -highest_pct) %>%
  ## reshape to longer
  pivot_longer(cols = starts_with("pct"),
               names_to = "race",
               values_to = "pct") %>%
  ## recode to match race names, and go to proportion
  mutate(race = str_replace(race, "pct", ""),
         race = recode(race, "api" = "asian",
                             "others" = "other"),
         race_surname = pct/100) %>%
  select(-pct) %>%
  ## merge the statewide race proportions, P(race)
  left_join(race_prop, by = "race") %>%
  ## calculate the quantity of interest
  ## P(surname | race) = P(race | surname) * P(surname) / P(race)
  mutate(prob_surname_race = race_surname * (count/total.count) / race_prop) %>%
  rename("posib_race" = race)
```

```r
## reshape the FL census data so we have race as a variable, then pct of pop
merge_temp <- FLCensus %>%
  pivot_longer(cols = c(white, black, hispanic, asian, other),
               names_to = "pop_race",
               values_to = "pop_pct") %>%
  inner_join(FLVoters, by = c("county", "VTD")) %>%
  inner_join(cnames_reshape, by = c("surname", "pop_race" = "posib_race")) %>%
  mutate(race_residence = prob_surname_race * pop_pct)

## then calculate the summation
name_residence <- merge_temp %>%
  group_by(surname, county, VTD) %>%
  summarize(name_residence = sum(race_residence))

## now we have all quantities of interest, can merge all together
## and calculate the predicted race
FLVoters_full <- merge_temp %>%
  inner_join(name_residence, by = c("surname", "county", "VTD")) %>%
  mutate(pred_race = prob_surname_race * pop_pct / name_residence) %>%
  select(surname, pop_race, race, pred_race, county, VTD)
```

```r
## now filter to save the highest predicted race,
## and see if it matches actual race
FL_updated <- FLVoters_full %>%
  ungroup() %>%
  group_by(surname, county, VTD) %>%
  ## select row per name with the largest percentage
  ## by keeping the first instance after arranging by percentage
  filter(row_number(desc(pred_race)) == 1) %>%
  # Ungroup to avoid errors later
  ungroup()

## calculate the new true positive rate
race_tp_new <- FL_updated %>%
```

```r
  group_by(race) %>%
  summarize(tp = mean(race == pop_race)) %>%
  arrange(desc(tp))

race_tp_new
```

```
## # A tibble: 6 x 2
##   race          tp
##   <chr>      <dbl>
## 1 white    0.942
## 2 hispanic 0.857
## 3 black    0.628
## 4 asian    0.604
## 5 other    0.00797
## 6 native   0
```

```r
## proportion of blacks among those with surname "White"
filter(cnames, surname == "WHITE") %>%
  select(pctblack) %>%
  pull()
```

```
## [1] 27.38
```

```r
## Predicted probability of being black given residence location
filter(FLVoters_full, surname == "WHITE", pop_race == "black") %>%
  select(pred_race) %>%
  summary()
```

```
##    pred_race
##  Min.   :0.004588
##  1st Qu.:0.072232
##  Median :0.159496
##  Mean   :0.250711
##  3rd Qu.:0.293640
##  Max.   :0.981864
```

```r
## the new false positive rate
race_fp_new <- FL_updated %>%
  group_by(pop_race) %>%
  summarize(fp = mean(race != pop_race)) %>%
  arrange(desc(fp))

race_fp_new
```

```
## # A tibble: 5 x 2
##   pop_race    fp
##   <chr>    <dbl>
## 1 other    0.778
## 2 asian    0.333
## 3 black    0.220
## 4 hispanic 0.212
## 5 white    0.122
```

Random Variables and Probability Distributions

Random Variables

Bernoulli and Uniform Distributions

```r
## uniform PDF: x = 0.5, interval = [0, 1]
dunif(0.5, min = 0, max = 1)
```

```
## [1] 1
```

```r
## uniform CDF: x = 1, interval = [-2, 2]
punif(1, min = -2, max = 2)
```

```
## [1] 0.75
```

```r
sims <- 1000
p <- 0.5 # success probabilities
x <- runif(sims, min = 0, max = 1) # uniform [0, 1]
head(x)
```

```
## [1] 0.9133371 0.6122063 0.5716837 0.8005061 0.6264327
## [6] 0.4131128
```

```r
y <- as.integer(x <= p) # Bernoulli; turn TRUE/FALSE to 1/0
head(y)
```

```
## [1] 0 0 0 0 0 1
```

```r
mean(y) # close to success probability p, proportion of 1's vs. 0's
```

```
## [1] 0.497
```

Binomial Distribution

```r
## PMF when x = 2, n = 3, p = 0.5
dbinom(2, size = 3, prob = 0.5)
```

```
## [1] 0.375
```

```r
## CDF when x = 1, n = 3, p = 0.5
pbinom(1, size = 3, prob = 0.5)
```

```
## [1] 0.5
```

```r
## number of voters who turn out
voters <- c(1000, 10000, 100000)
dbinom(voters / 2, size = voters, prob = 0.5)
```

```
## [1] 0.025225018 0.007978646 0.002523126
```

Normal Distribution

```r
## plus minus one standard deviation from the mean
pnorm(1) - pnorm(-1)
```

```
## [1] 0.6826895
```

```r
## plus minus two standard deviations from the mean
pnorm(2) - pnorm(-2)
```

```
## [1] 0.9544997
```

```r
mu <- 5
sigma <- 2
## plus minus one standard deviation from the mean
pnorm(mu + sigma, mean = 5, sd = 2) - pnorm(mu - sigma, mean = 5, sd = 2)
```

```
## [1] 0.6826895
```

```r
## plus minus two standard deviations from the mean
pnorm(mu + 2*sigma, mean = 5, sd = 2) - pnorm(mu - 2*sigma, mean = 5, sd = 2)
```

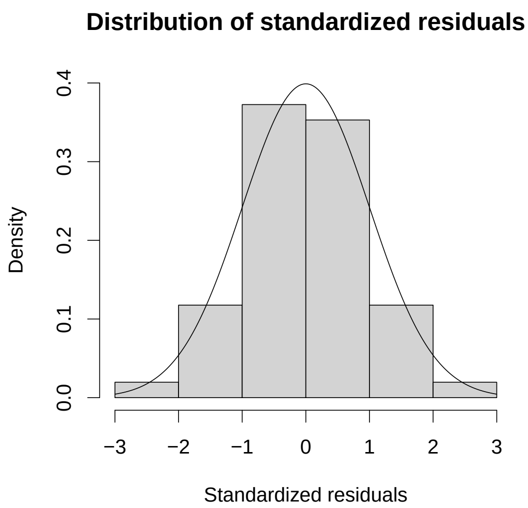
```
## [1] 0.9544997
```

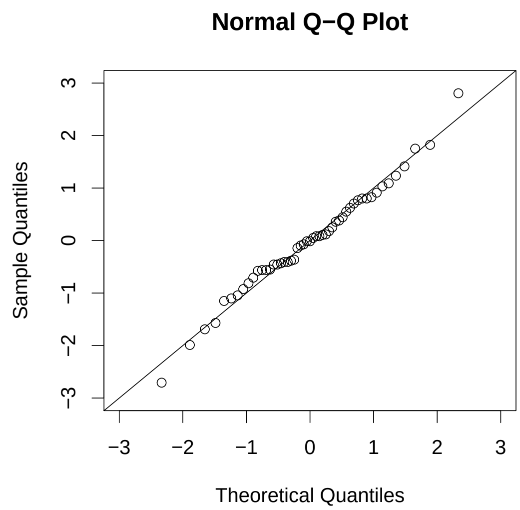
```r
## see the page reference above
## `Obama2012.z' is Obama's 2012 standardized vote share
## `Obama2008.z' is Obama's 2008 standardized vote share
fit1
```

```
##
## Call:
## lm(formula = Obama2012.z ~ -1 + Obama2008.z, data = pres)
##
## Coefficients:
## Obama2008.z
##      0.9834
```

```r
par(cex = 1.5)
e <- resid(fit1)
## z-score of residuals
e.zscore <- scale(e)
## alternatively we can divide residuals by their standard deviation
```

```r
e.zscore <- e / sd(e)
hist(e.zscore, freq = FALSE, ylim = c(0, 0.4),
     xlab = "Standardized residuals",
     main = "Distribution of standardized residuals")
x <- seq(from = -3, to = 3, by = 0.01)
lines(x, dnorm(x)) # overlay the normal density
qqnorm(e.zscore, xlim = c(-3, 3), ylim = c(-3, 3)) # quantile-quantile plot
abline(0, 1) # 45 degree line
```






```r
e.sd <- sd(e)
e.sd
```

```
## [1] 0.1812239
```

```r
CA.2008 <- filter(pres, state == "CA") %>%
  select(Obama2008.z) %>%
  pull()
CA.2008
```

```
## [1] 0.8720631
```

```r
CA.mean2012 <- coef(fit1) * CA.2008
CA.mean2012
```

```
## Obama2008.z
##   0.8576233
```

```r
## area to the right; greater than CA.2008
pnorm(CA.2008, mean = CA.mean2012, sd = e.sd, lower.tail = FALSE)
```

```
## [1] 0.4682463
```

```r
TX.2008 <- filter(pres, state == "TX") %>%
  select(Obama2008.z) %>%
  pull()
TX.mean2012 <- coef(fit1) * TX.2008
TX.mean2012
```

```
## Obama2008.z
##  -0.6567543
```

```r
pnorm(TX.2008, mean = TX.mean2012, sd = e.sd, lower.tail = FALSE)
```

```
## [1] 0.5243271
```

Expectation and Variance

```r
## theoretical variance: p was set to 0.5 earlier
p * (1-p)
```

```
## [1] 0.25
```

```r
## sample variance using `y' generated earlier
var(y)
```

```
## [1] 0.2502412
```

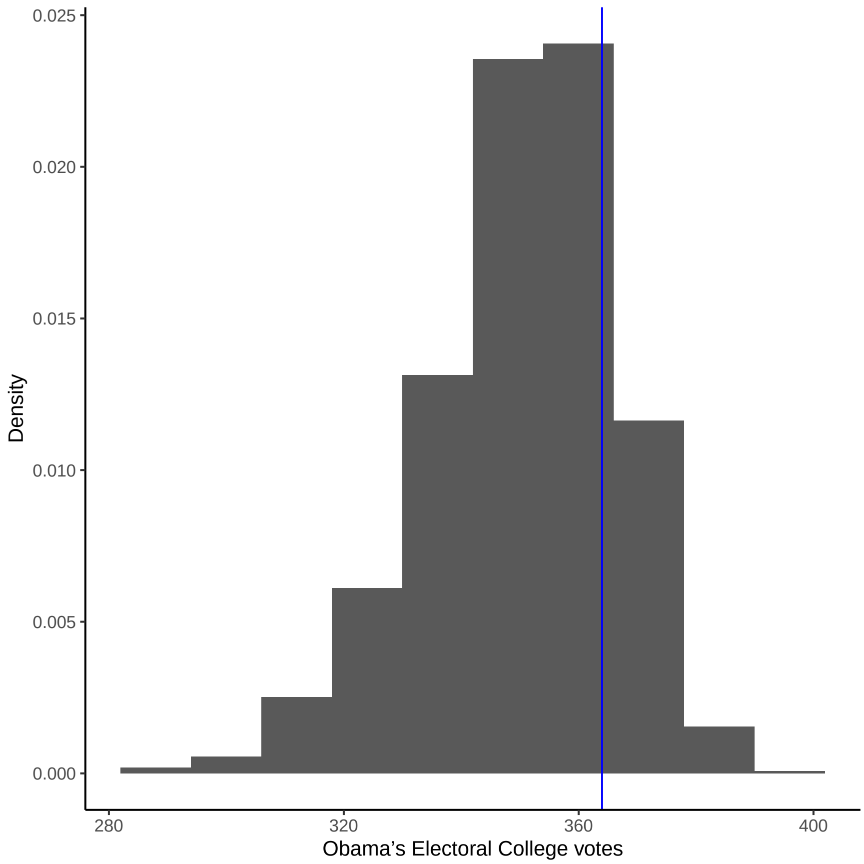
Predicting Election Outcomes with Uncertainty

```r
# load the data
data("pres08", package = "qss")
# Add variable for Obama vote share
pres08 <- pres08 %>%
  mutate(p = Obama / (Obama + McCain))
```

```r
sim_election <- function(.id, df, n_draws = 1000) {
  # For each state randomly sample
  mutate(df, draws = rbinom(n(), n_draws, p)) %>%
  filter(draws > (n_draws / 2)) %>%
  summarize(EV = sum(EV),
            .id = .id)
}
```

```r
sims <- 10000
sim_results <- map_df(seq_len(sims),
                      ~ sim_election(.x, pres08, n_draws = 1000))
```

```r
ggplot(sim_results, aes(x = EV, y = ..density..)) +
  geom_histogram(binwidth = 12) +
  geom_vline(xintercept = 364, color = "blue", size = 0.5) +
  labs(x = "Obama’s Electoral College votes", y = "Density")
```




```r
sim_results %>%
  select(EV) %>%
  summarize_all(list(mean = mean,
                     median = median,
                     var = var,
                     sd = sd))
```

```
##       mean median      var       sd
## 1 352.0262    353 274.5194 16.56863
```

```r
## probability of binomial random variable taking greater than n/2 votes
n_draws <- 1000

pres08 %>%
  mutate(pb = pbinom(n_draws / 2, size = n_draws, prob = p,
                     lower.tail  = FALSE)) %>%
  summarize(mean = sum(pb * EV))
```

```
##       mean
## 1 352.1388
```

```r
var_sd <- pres08 %>%
  mutate(pb = pbinom(n_draws / 2, size = n_draws, prob = p,
                     lower.tail  = FALSE)) %>%
  summarize(V = sum(pb * (1 - pb) * EV ^ 2),
            sd = sqrt(V))
```

Large Sample Theorems

The Law of Large Numbers

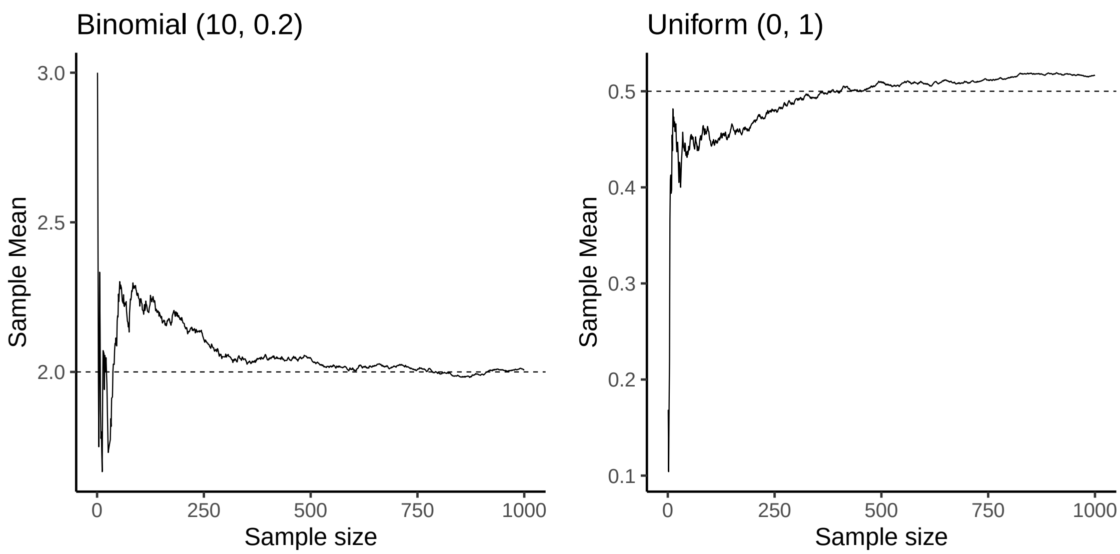
```r
sims <- 1000
p <- 0.2
size <- 10
## Putting the results into a tibble
lln_binom <- tibble(
  n = seq_len(sims),
  x = rbinom(sims, prob = p, size = size),
  mean = cumsum(x) / n,
  distrib = str_c("Binomial(", size, ", ", p, ")"))

## look at the last few rows
tail(lln_binom)
```

```
## # A tibble: 6 x 4
##       n     x  mean distrib
##   <int> <int> <dbl> <chr>
## 1   995     2  2.01 Binomial(10, 0.2)
## 2   996     2  2.01 Binomial(10, 0.2)
## 3   997     1  2.01 Binomial(10, 0.2)
## 4   998     0  2.01 Binomial(10, 0.2)
## 5   999     2  2.01 Binomial(10, 0.2)
## 6  1000     3  2.01 Binomial(10, 0.2)
```

```r
lln_unif <-
 tibble(n = seq_len(sims),
        x = runif(sims),
        mean = cumsum(x) / n,
        distrib = str_c("Uniform(0, 1)"))

tail(lln_unif)
```

```
## # A tibble: 6 x 4
##       n     x  mean distrib
##   <int> <dbl> <dbl> <chr>
## 1   995 0.353 0.516 Uniform(0, 1)
## 2   996 0.706 0.516 Uniform(0, 1)
## 3   997 0.688 0.516 Uniform(0, 1)
## 4   998 0.826 0.517 Uniform(0, 1)
## 5   999 0.247 0.516 Uniform(0, 1)
## 6  1000 0.735 0.517 Uniform(0, 1)
```

```r
## Binomial
ggplot(data = lln_binom) +
  geom_line(aes(x = n, y = mean)) +
  geom_hline(yintercept =  2, lty = "dashed") + # expectation
  labs(x = "Sample size", y = "Sample Mean",
       title = "Binomial (10, 0.2)")
## Uniform
ggplot(data = lln_unif) +
  geom_line(aes(x = n, y = mean)) +
  geom_hline(yintercept =  0.5, lty = "dashed") + # expectation
  labs(x = "Sample size", y = "Sample Mean",
       title = "Uniform (0, 1)")
```



The Central Limit Theorem

```r
sims <- 1000
n.samp <- 1000
z.binom <- z.unif <- rep(NA, sims)
for (i in 1:sims) {
    x <- rbinom(n.samp, p = 0.2, size = 10)
    z.binom[i] <- (mean(x) - 2) / sqrt(1.6 / n.samp)
    x <- runif(n.samp, min = 0, max = 1)
    z.unif[i] <- (mean(x) - 0.5) / sqrt(1 / (12 * n.samp))
}
## bind the results together
results <- tibble(z.binom = z.binom,
                  z.unif = z.unif,
                  n.samp = seq(1:n.samp))

## plot the results
## binomial
ggplot(results) +
  geom_histogram(aes(x = z.binom, y = ..density..),
```

```
                 bins = 20) +
  stat_function(fun = dnorm, color = "blue") +
  labs(x = "z-score",
       y = "Density",
       title = "Binomial (0.2, 10)")

## uniform
ggplot(results) +
  geom_histogram(aes(x = z.unif, y = ..density..),
                 bins = 20) +
  stat_function(fun = dnorm, color = "blue") +
  labs(x = "z-score",
       y = "Density",
       title = "Uniform (0, 1)")
```