hw_2

Hengle Li

2/1/2020

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
```

```
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.2     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(broom)
library(rcfss)
library(rsample)
library(patchwork)
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(ISLR)
library(yardstick)
```

```
## For binary classification, the first factor level is assumed to be the event.
## Set the global option `yardstick.event_first` to `FALSE` to change this.
```

```
##
## Attaching package: 'yardstick'
```

```
## The following object is masked from 'package:readr':
##
##     spec
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'caret'

## The following objects are masked from 'package:yardstick':
##
##     precision, recall

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
set.seed(123)
options(digits = 3)
```

The Bayesian Classifier

generate the sample

```r
#use runif given X1 and X2 are random uniform variables
N_sample <- tibble(X1 = runif(200, -1, 1),
                   X2 = runif(200, -1, 1),
                   #use rnorm to add error in the sample as normal distribution
                   e = rnorm(200, 0, 0.5))
#calculate Y by definition
N_sample <- N_sample %>%
  mutate(log_Y = X1 + X1^2 + X2 + X2^2 + e) %>%
  mutate(Y = logit2prob(log_Y)) %>%
#transform log-odds to probability
  mutate(result = ifelse(Y > 0.5, "success", "fail"))
```

create the naive Bayesian model

```r
#set up x and y
variables <- N_sample %>%
  select(-Y, -result)
x <- variables
y <- N_sample$result
#train model
nb_model <- train(
  x = x,
  y = y,
  method = "nb"
)
```

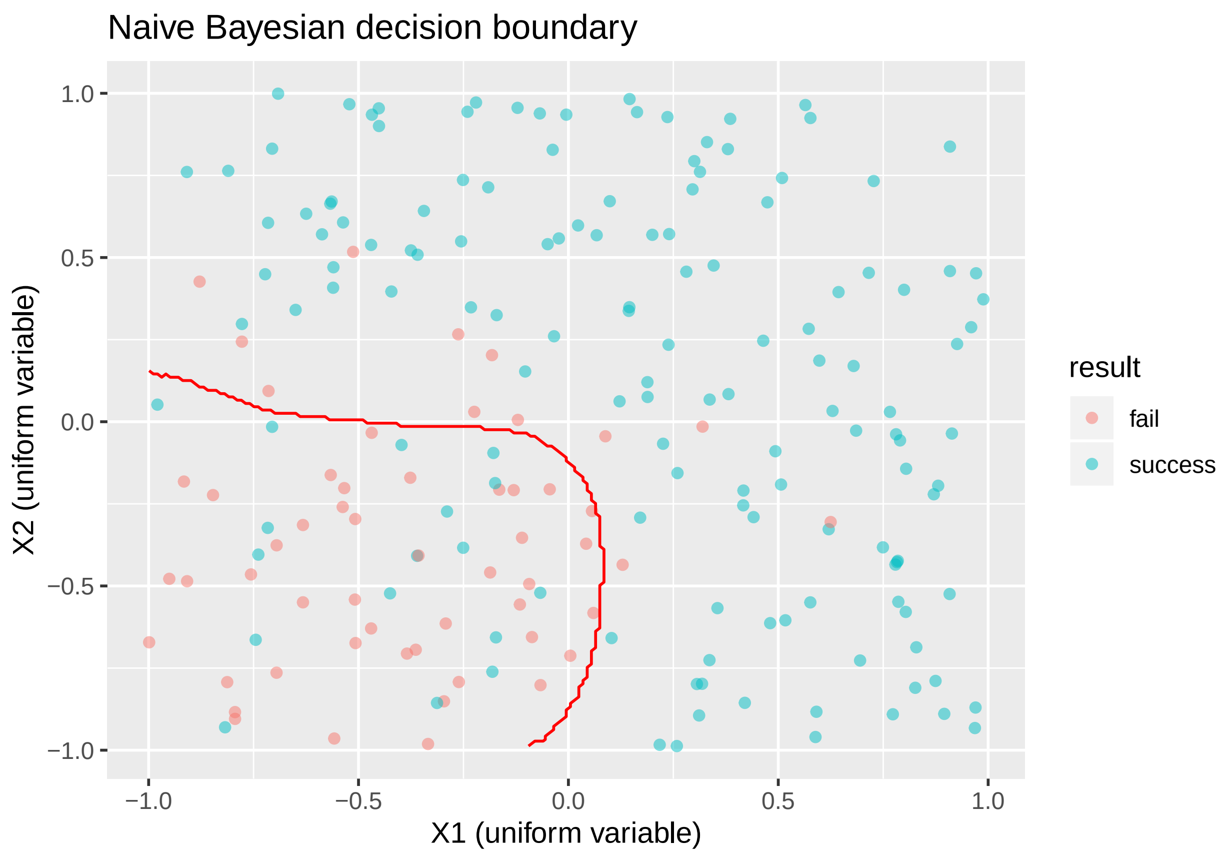
create a new set of data for testing

```r
X1 <- seq(min(N_sample$X1), max(N_sample$X1), length = 200)
X2 <- seq(min(N_sample$X2), max(N_sample$X2), length = 200)
test_set <- expand.grid(X1 = X1, X2 = X2)
```

calculate a decision boundary with the test set and plot it against the training set

```
#join the needed data first
pred_graph <- cbind(test_set, pred = predict(nb_model, test_set))

N_sample %>%
  ggplot(aes(x = X1, y = X2, color = result)) +
  geom_point(alpha = .5) +
  geom_contour(data = pred_graph,
               aes(z = as.numeric(pred)),
               color = "red",
               breaks = c(1.5)) +
  labs(title = "Naive Bayesian decision boundary",
       x = "X1 (uniform variable)",
       y = "X2 (uniform variable)")
```


Naive Bayesian decision boundary
X2 (uniform variable)
X1 (uniform variable)
result
fail
success