Assignment09

Zhen Liu

2022-11-30

Basic concepts in classification

```
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────── tidyverse 1.3.2 ──
## ✔ ggplot2 3.3.6      ✔ purrr   0.3.4
## ✔ tibble  3.1.8      ✔ dplyr   1.0.10
## ✔ tidyr   1.2.1      ✔ stringr 1.4.1
## ✔ readr   2.1.3      ✔ forcats 0.5.2
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
```

Q1

1. A classification rule

Given a feature vector and a set of categories, we want to assign a class label taken from the categories to the feature vecrtor according to which class it belongs to. Exp: The food was fantastic.

2. A learning algrithm

A learning algorithm is a set of instructions used in machine learning that allows a computer program to imitate the way a human gets better at characterizing some types of information. Exp: regression, classfication

3. Training data

Training data is used to teach prediction models that use machine learning algorithms how to extract features that are relevant to specific business goals. Exp: for supervised ml, the training dataset is labeled.

4. Feature vector feature vectors are used to represent numeric or symbolic characteristics, called features, of an object in a mathematical, easily analyzable way Exp: [x1,x2,x3….]

5. Label

In machine learning, data labeling is the process of identifying raw data (images, text files, videos, etc.) and adding one or more meaningful and informative labels to provide context so that a machine learning model can learn from it Exp: sales = [x1,x2,x3….]

6. Expected vector

The expected value of a random vector (or matrix) is a vector (or matrix) whose elements are the expected values of the individual random variables that are the elements of the random vector.

7. Train error

Training error is the error that you get when you run the trained model back on the training data.

Exp: mse

8. The train test split

The train-test split procedure is used to estimate the performance of machine learning algorithms when they are used to make predictions on data not used to train the model.

A chi-squared test of population variance

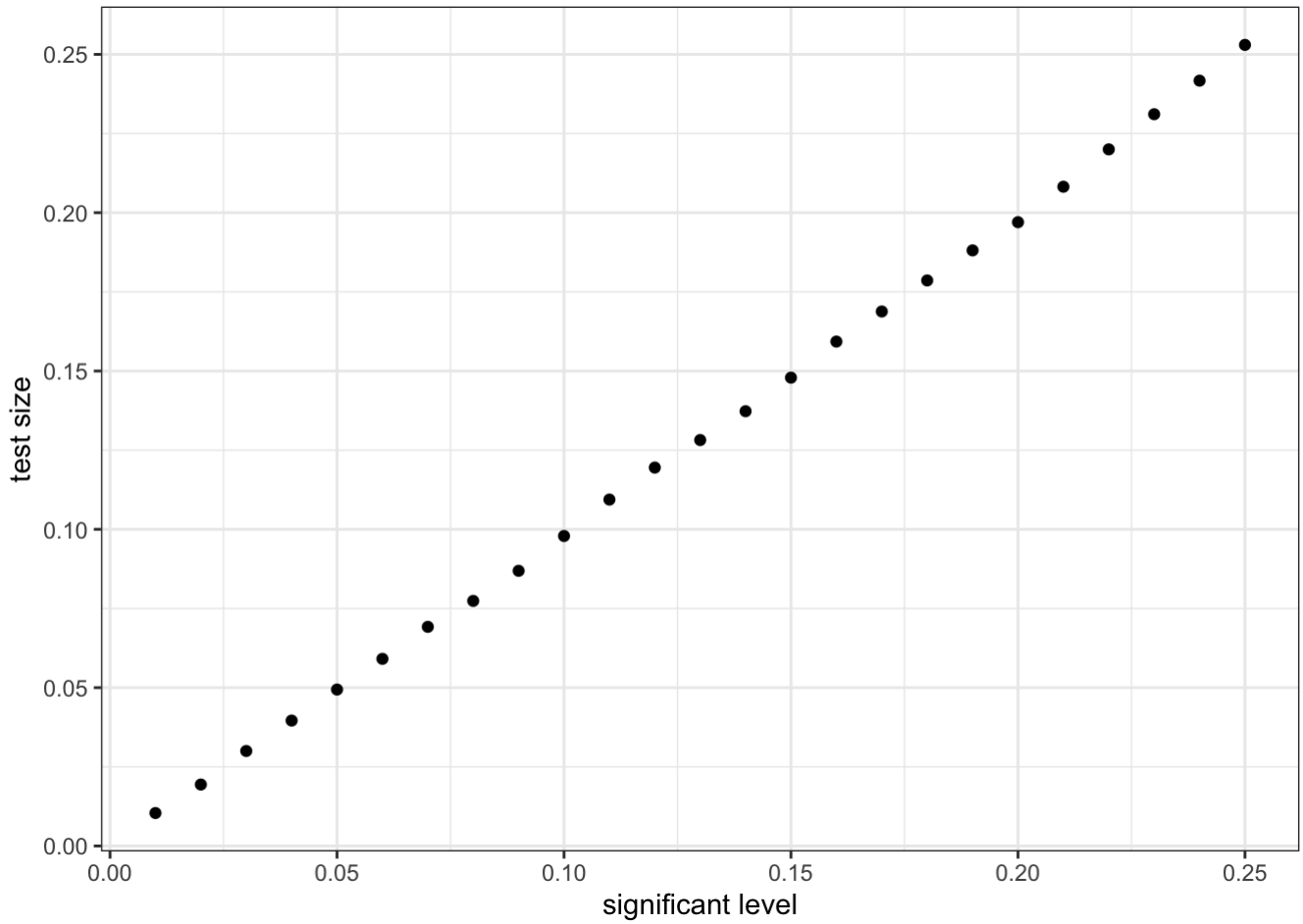
Q1

```
chi_square_test_one_sample_var<-function(sample,sigma_square_null){

  sample<-sample[!is.na(sample)]

  n<-length(sample)

  chi_squared_statistic<-(n-1)*var(sample)/sigma_square_null

  p_value<-2*min(pchisq(chi_squared_statistic,df=n-1),
                 1-pchisq(chi_squared_statistic,df=n-1))

  return(p_value)

}
```

Q2

```
num_trials<-10000
sample_size<-100
mu<-1
sigma<-2

set.seed(0)

single_alpha_test_size_simulation_df <- data.frame(trial=seq(num_trials)) %>%
  mutate(sample=map(.x=trial,.f=~rnorm(n=sample_size,mean=mu,sd=sigma))) %>%
  mutate(p_value=map(.x=sample, .f=~chi_square_test_one_sample_var(.x,4)))

alpha_list = seq(0.01,0.25,0.01)


compute_test_size<- function(alpha){

  type_1_error = single_alpha_test_size_simulation_df$p_value<alpha

  return(mean(type_1_error))
}
```

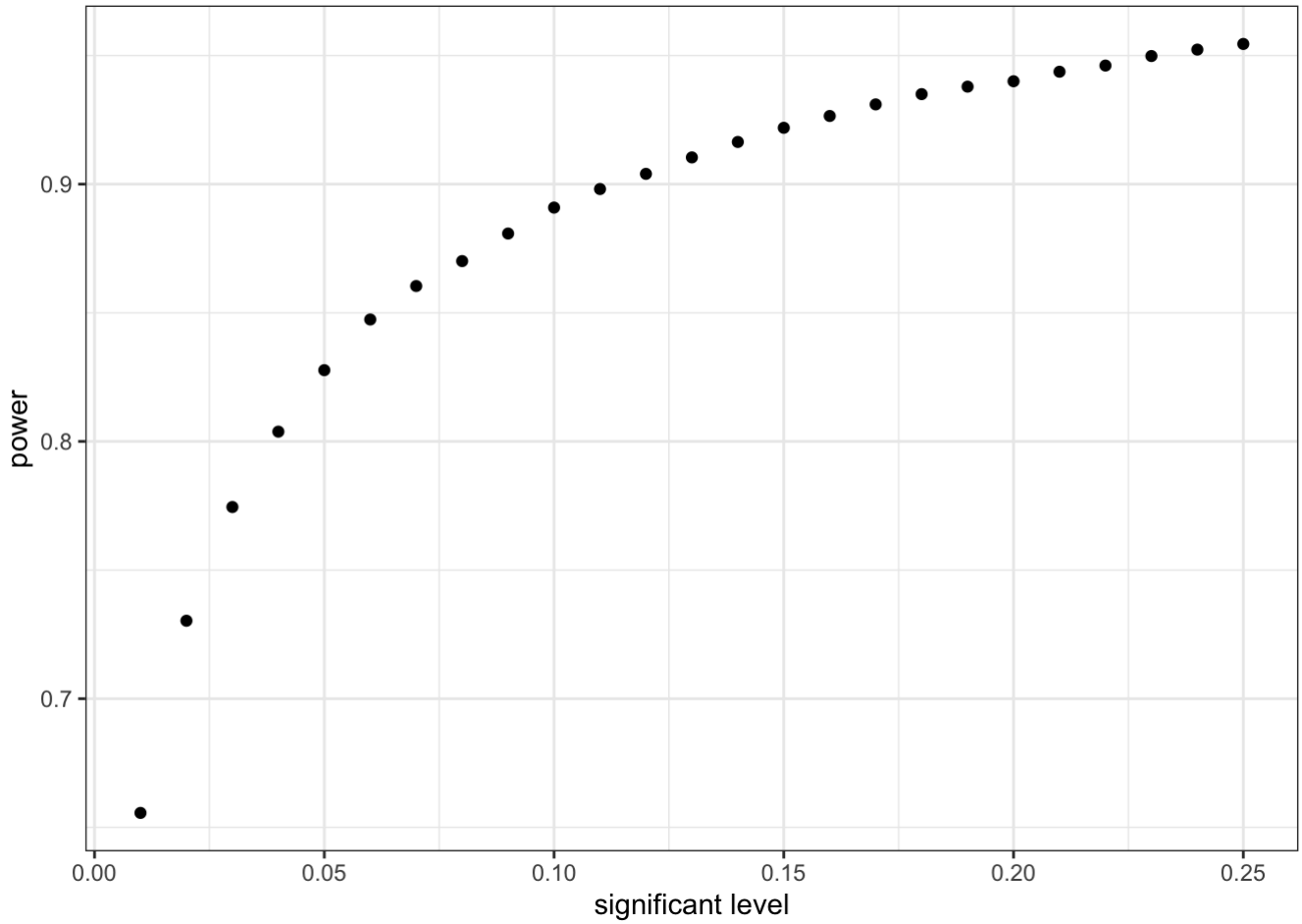
```
multiple_alpha_test_size_simulation_df<-data.frame(alpha=alpha_list)%>%
  mutate(test_size=map_dbl(alpha,compute_test_size))

multiple_alpha_test_size_simulation_df%>%
  ggplot(aes(x=alpha,y=test_size))+ geom_point()+xlab('significant level')+ylab('test
size')+theme_bw()
```



Q4

```
num_trials<-10000
sample_size<-100
mu<-1
sigma<-2.45
sigma_0<-2


set.seed(0)

single_alpha_power_df <- data.frame(trial=seq(num_trials)) %>%
  mutate(sample=map(.x=trial,.f=~rnorm(n=sample_size,mean=mu,sd=sigma))) %>%
  mutate(p_value=map(.x=sample, .f=~chi_square_test_one_sample_var(.x,4)))

alpha_list = seq(0.01,0.25,0.01)


compute_power<- function(alpha){

  reject_null<- single_alpha_power_df$p_value<alpha

  return(mean(reject_null))
}
```

```
multiple_alpha_power_df<-data.frame(alpha=alpha_list)%>%
  mutate(power=map_dbl(alpha,compute_power))

multiple_alpha_power_df%>%
  ggplot(aes(x=alpha,y=power))+ geom_point()+xlab('significant level')+ylab('power')+
theme_bw()
```



Q5

```
library(palmerpenguins)
```

```
bill_adelie<-penguins%>%
  filter(species =='Adelie')%>%
  select(bill_length_mm)
```

```
chi_square_test_one_sample_var(sample = bill_adelie,sigma_square_null=9)
```

```
## [1] 0.05196711
```

The p-value is less than significant level

The train test split

Q1

```
library(Stat2Data)
data(Hawks)
```

```
hawks_total<-Hawks%>%
  select(Weight,Wing,Tail,Hallux,Species)%>%
  filter(Species!="RT")%>%
  drop_na()%>%
  mutate(Species=as.numeric(Species=="SS"))
```

Q2

```
num_total<-hawks_total%>%nrow()
num_train<-floor(num_total*0.6)
num_test<-num_total-num_train

set.seed(1)
test_inds<-sample(seq(num_total),num_test)
train_inds<-setdiff(seq(num_total),test_inds)

hawks_train<-hawks_total%>%
  filter(row_number() %in% train_inds)

hawks_test<-hawks_total%>%
  filter(row_number() %in% test_inds)
```

Q3

```
hawks_train_x<-hawks_train%>%select(-Species)
hawks_train_y<-hawks_train%>%select(Species)


hawks_test_x<-hawks_test%>%select(-Species)
hawks_test_y<-hawks_test%>%select(Species)
```

Q4

```
n_positive<-hawks_train_y%>%
  filter(Species==1)

Prob_train <-n_positive/194
```

Q5

```
n_positive_test<-hawks_test_y%>%
  filter(Species==1)


prob_test<-n_positive_test/130
```

Based on the train and test, the test error should be 0.022

Mutivariate distributions and parameter estimation

Q1

```
hawks_rt<-Hawks%>%
  filter(Species =='RT')%>%
  select(Wing,Weight,Tail)%>%
  drop_na()
```

Q2

```
sigma_rt<- cov(hawks_rt,use = "complete.obs")
sigma_rt
```

```
##             Wing     Weight     Tail
## Wing    970.2672  1983.0489 148.8996
## Weight 1983.0489 35800.5187 705.7409
## Tail    148.8996   705.7409 211.4762
```

Investigate hypotheis testing where the Gaussian model assumptions are violated

Q1

$$F_X(x) = F(a+1) - F(a) = (a+1)^2 - 2a(a+1) - (a^2 - 2a^2) = 1$$

do not know how to compute the CDF.

Q2

```
generate_sample_X<-function(sample_size,mu){

  sample_U<-runif(sample_size)
  generate_sample_X<-sample_U
}

return(generate_sample_X)
```

```
## function(sample_size,mu){
##
##   sample_U<-runif(sample_size)
##   generate_sample_X<-sample_U
## }
```

Q4

```
set.seed(0)
sample_size_seq<- seq(2,40,2)
num_trials_per_size<-10000
mu_0<-3
mu_1<-3

df<-crossing(sample_size=sample_size_seq,trials=seq(num_trials_per_size))
```