Week 10 - Homework

STAT 420, Summer 2023, Adriane Yi

2023-07-24

Exercise 1 (Simulating Wald and Likelihood Ratio Tests)

In this exercise we will investigate the distributions of hypothesis tests for logistic regression. For this exercise, we will use the following predictors.

```
sample_size = 150
set.seed(120)
x1 = rnorm(n = sample_size)
x2 = rnorm(n = sample_size)
x3 = rnorm(n = sample_size)
```

Recall that

$$p(\mathbf{x}) = P[Y = 1 \mid \mathbf{X} = \mathbf{x}]$$

Consider the true model

$$\log\left(\frac{p(\mathbf{x})}{1 - p(\mathbf{x})}\right) = \beta_0 + \beta_1 x_1$$

where

- $\beta_0 = 0.4$
- $\beta_1 = -0.35$

(a) To investigate the distributions, simulate from this model 2500 times. To do so, calculate

$$P[Y = 1 \mid \mathbf{X} = \mathbf{x}]$$

for an observation, and then make a random draw from a Bernoulli distribution with that success probability. (Note that a Bernoulli distribution is a Binomial distribution with parameter $n = 1$. There is no direction function in `R` for a Bernoulli distribution.)

Each time, fit the model:

$$\log\left(\frac{p(\mathbf{x})}{1 - p(\mathbf{x})}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

Store the test statistics for two tests:

- The Wald test for $H_0 : \beta_2 = 0$, which we say follows a standard normal distribution for "large" samples
- The likelihood ratio test for $H_0 : \beta_2 = \beta_3 = 0$, which we say follows a χ^2 distribution (with some degrees of freedom) for "large" samples

```
num_sims = 2500
beta_0 = 0.4
beta_1 = -0.35

eta = beta_0 + beta_1 * x1
p = 1 / (1 + exp(-eta))
y = rep(0, sample_size)
sim_data = data.frame(y = y, x1 = x1, x2 = x2, x3 = x3)

wald_test = rep(0, num_sims)
lrt = rep(0, num_sims)

for(i in 1 : num_sims){
  sim_data$y = rbinom(n = sample_size, size = 1, prob = p)

  glm_mode = glm(y ~ x1 + x2 + x3, family = binomial, data = sim_data)
  null_mode = glm(y ~ x1, family = binomial, data = sim_data)

  wald_test[i] = summary(glm_mode)$coef["x2", "z value"]
  lrt[i] = anova(null_mode, glm_mode, test = "LRT")[2, "Deviance"]
}
```

```
head(wald_test)
```

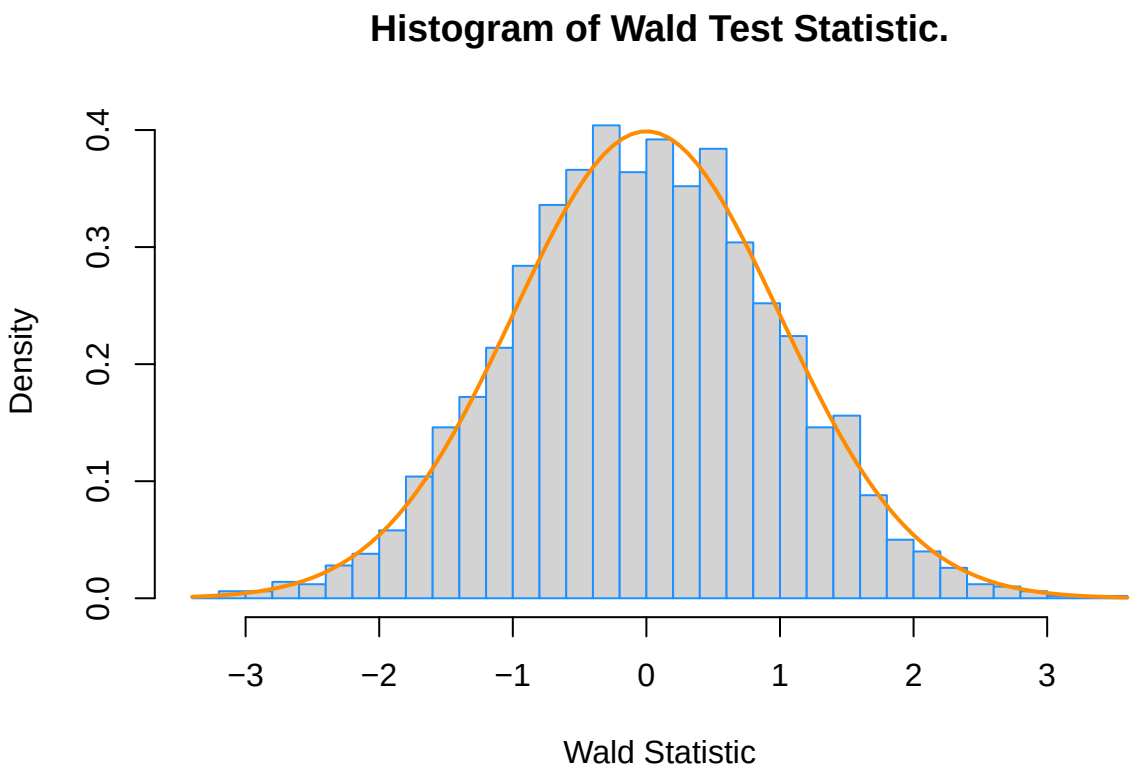
```
## [1] -0.02506 -1.88683 -0.87149 -0.50917 -0.42352 -0.88244
```

```
head(lrt)
```

```
## [1] 0.4763 4.2069 3.7059 0.9890 0.6986 1.1226
```

(b) Plot a histogram of the empirical values for the Wald test statistic. Overlay the density of the true distribution assuming a large sample.

```
hist(wald_test, freq = FALSE, main = "Histogram of Wald Test Statistic.", xlab = "Wald Statistic",
     border = "dodgerblue", col = "lightgray",
     breaks = 25)

curve(dnorm(x, mean = 0, sd = 1), add = TRUE, col = "darkorange", lwd = 2)
```



(c) Use the empirical results for the Wald test statistic to estimate the probability of observing a test statistic larger than 1. Also report this probability using the true distribution of the test statistic assuming a large sample.

```
# empirical wald probability
(empirical_prob_wald = mean(wald_test > 1))
```

```
## [1] 0.1524
```

```
# true wald probability
(true_prob_wald = pnorm(1, mean = 0, sd = 1, lower.tail = FALSE))
```

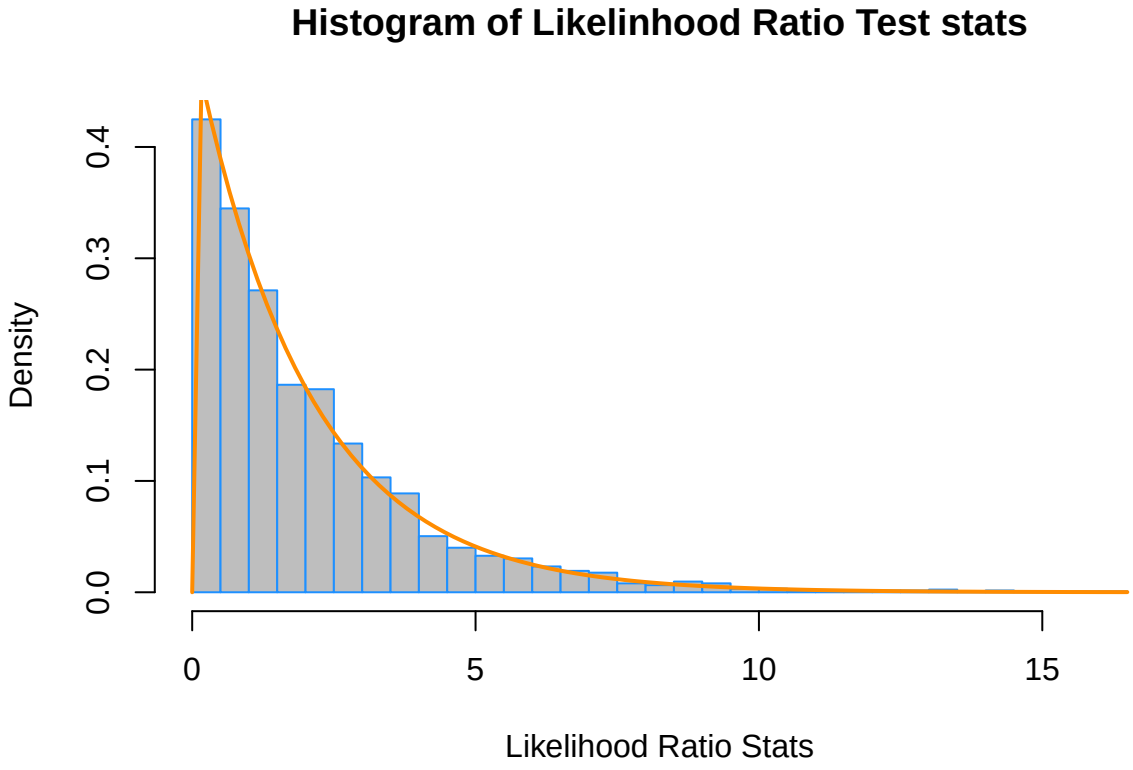
```
## [1] 0.1587
```

Solution:

- Empirical wald probability : 0.1524
- True wald probability: 0.1587

(d) Plot a histogram of the empirical values for the likelihood ratio test statistic. Overlay the density of the true distribution assuming a large sample.

```
hist(lrt, freq = FALSE,
     main = "Histogram of Likelinhood Ratio Test stats",
     xlab = "Likelihood Ratio Stats",
     col = "grey",
     border= "dodgerblue",
     breaks = 25)

curve(dchisq(x, df = 2), add = TRUE, col = "darkorange", lwd =2)
```



(e) Use the empirical results for the likelihood ratio test statistic to estimate the probability of observing a test statistic larger than 5. Also report this probability using the true distribution of the test statistic assuming a large sample.

```
(empricial_prob_est = mean(lrt > 5))
```

```
## [1] 0.0872
```

```
(true_prob_est = 1 - pchisq(5, df = 2))
```

```
## [1] 0.08208
```

(f) Repeat **(a)**-**(e)** but with simulation using a smaller sample size of 10. Based on these results, is this sample size large enough to use the standard normal and χ^2 distributions in this situation? Explain.

```
sample_size = 10
set.seed(120)
x1 = rnorm(n = sample_size)
x2 = rnorm(n = sample_size)
x3 = rnorm(n = sample_size)
```

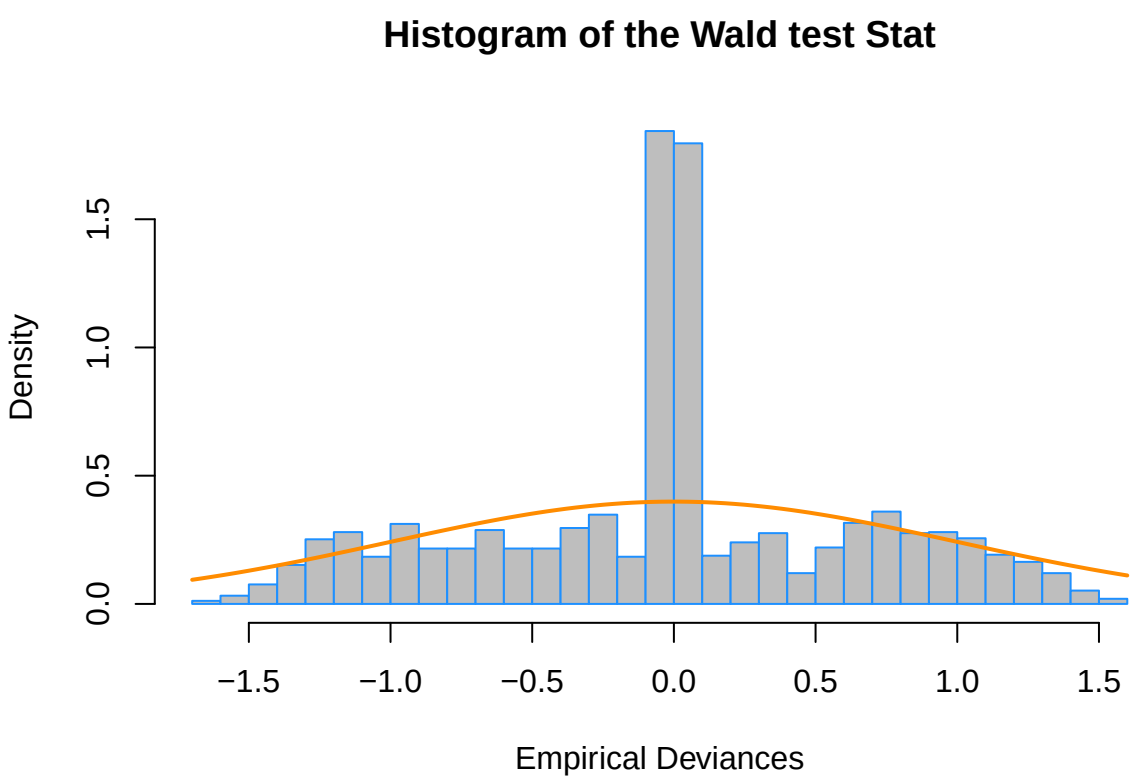
Simulation

```
num_sims = 2500
beta_0 = 0.4
beta_1 = -0.35
```

```r
eta = beta_0 + beta_1 * x1
p = 1 / (1 + exp(-eta))

y = rep(0, sample_size)
sim_data_2 = data.frame(y = y, x1 = x1, x2 = x2, x3 = x3)

wald_test_2 = rep(0, num_sims)
lrt_2 = rep(0, num_sims)

for(i in 1 : num_sims){
  sim_data_2$y = rbinom(n = sample_size, size = 1, prob = p)

  glm_mode_2 = glm(y ~ x1 + x2 + x3, family = binomial, data = sim_data_2)
  null_mode_2 = glm(y ~ x1, family = binomial, data = sim_data_2)

  wald_test_2[i] = summary(glm_mode_2)$coef["x2", "z value"]
  lrt_2[i] = anova(null_mode_2, glm_mode_2, test = "LRT")[2, "Deviance"]
}
```

Plot a histogram of the empirical values for the Wald test statistic. Overlay the density of the true distribution assuming a large sample.

```r
hist(wald_test_2, freq = FALSE,
     main = "Histogram of the Wald test Stat",
     xlab = "Empirical Deviances",
     col = "grey",
     border = "dodgerblue",
     breaks = 25)
curve(dnorm(x, mean = 0, sd = 1), col = "darkorange", add = TRUE, lwd = 2)
```



Use the empirical results for the Wald test statistic to estimate the probability of observing a test statistic larger than 1. Also report this probability using the true distribution of the test statistic assuming a large sample.

```
mean(wald_test_2 > 1)
```

```
## [1] 0.0804
```

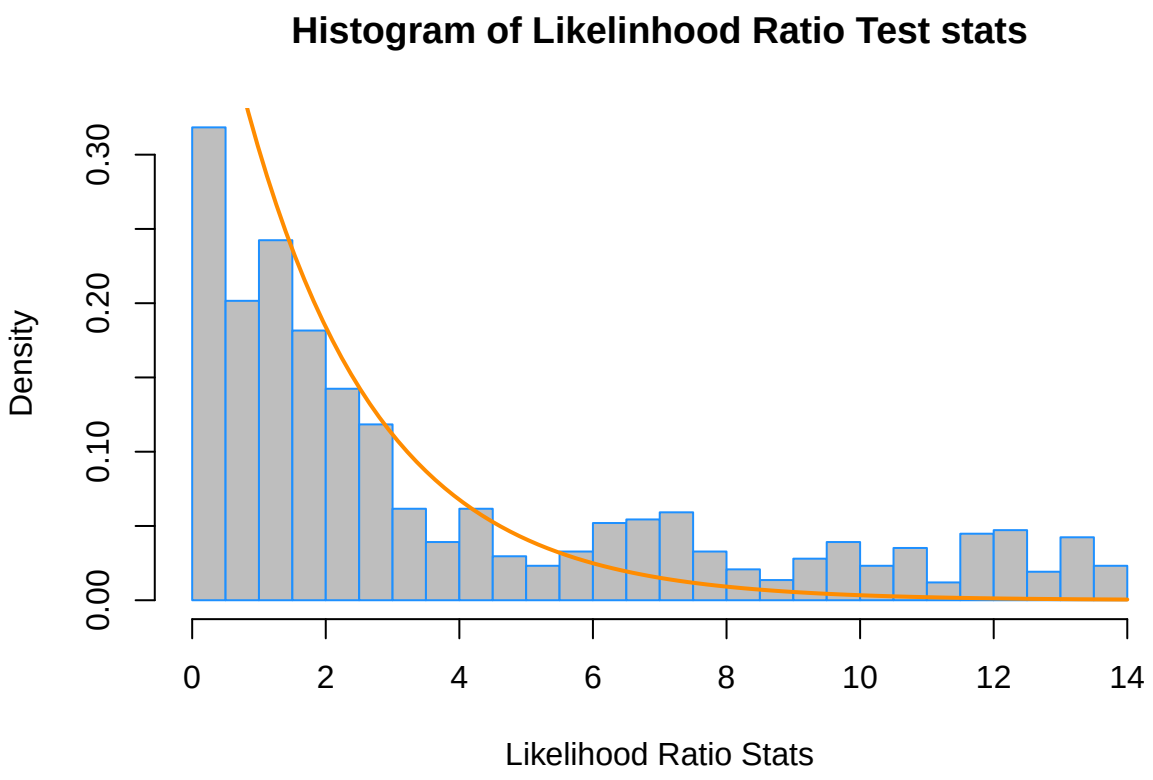
```
pnorm(1, mean = 0, sd = 1, lower.tail = FALSE)
```

```
## [1] 0.1587
```

Empirical probability of observing a test statistic larger than 1 : 0.0804 Same probability for true distribution of the test stat assuming larger sample: 0.1587

Plot a histogram of the empirical values for the likelihood ratio test statistic. Overlay the density of the true distribution assuming a large sample.

```
hist(lrt_2, freq = FALSE,
     main = "Histogram of Likelinhood Ratio Test stats",
     xlab = "Likelihood Ratio Stats",
     col = "grey",
     border= "dodgerblue",
     breaks = 25)

curve(dchisq(x, df = 2), add = TRUE, col = "darkorange", lwd =2)
```



Use the empirical results for the likelihood ratio test statistic to estimate the probability of observing a test statistic larger than 5. Also report this probability using the true distribution of the test statistic assuming a large sample.

```r
(empirical_prob_est = mean(lrt_2 > 5))
```

```
## [1] 0.3016
```

```r
(true_prob_est = 1 - pchisq(5, df = 2))
```

```
## [1] 0.08208
```

Solution:.
empirical probability : 0.3016 true probability: 0.0821

- Using sample size of 10, it definitely doesn't seem to be enough to use distributions–normal or chi-squared. As we can see in the histograms above, overlaying empirical results and the true distribution of test states, don't match well, that is the test statistics don't follow the distribution. For the wald test statistic, we see that the plot does not seem to follow standard normal distribution. Also the comparison between the empirical probability and true probability of a test statistic > 1 . Empricial probabilty is 0.0804 and true probability is 0.1587. For the likelihood ratio test statistics, empirical probability of a test statistic >5, (0.3016), and that of using true distribution of chi-squared, (0.08208), are significantly different suggesting that the empirical statistics don't follow chi-squared distribution.

Exercise 2 (Surviving the Titanic)

For this exercise use the `ptitanic` data from the `rpart.plot` package. (The `rpart.plot` package depends on the `rpart` package.) Use `?rpart.plot::ptitanic` to learn about this dataset. We will use logistic regression to help predict which passengers aboard the Titanic will survive based on various attributes.

```r
# install.packages("rpart")
# install.packages("rpart.plot")
library(rpart)
library(rpart.plot)
data("ptitanic")
```

For simplicity, we will remove any observations with missing data. Additionally, we will create a test and train dataset.

```r
ptitanic = na.omit(ptitanic)
set.seed(2021)
trn_idx = sample(nrow(ptitanic), 300)
ptitanic_trn = ptitanic[trn_idx, ]
ptitanic_tst = ptitanic[-trn_idx, ]
```

(a) Consider the model

$$\log\left(\frac{p(\mathbf{x})}{1-p(\mathbf{x})}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_3 x_4$$

where

$$p(\mathbf{x}) = P[Y = 1 \mid \mathbf{X} = \mathbf{x}]$$

is the probability that a certain passenger survives given their attributes and

- x_1 is a dummy variable that takes the value 1 if a passenger was 2nd class.
- x_2 is a dummy variable that takes the value 1 if a passenger was 3rd class.
- x_3 is a dummy variable that takes the value 1 if a passenger was male.
- x_4 is the age in years of a passenger.

Fit this model to the training data and report its deviance.

```
titanic_mod = glm(survived ~ pclass + sex + age + sex:age, family = "binomial", data = ptitanic_trn)
(titanic_mod_dev = titanic_mod$deviance)
```

```
## [1] 259.2
```

Solution:
Deviance for the model: 259.1653
.

(b) Use the model fit in **(a)** and an appropriate statistical test to determine if class played a significant role in surviving on the Titanic. Use $\alpha = 0.01$. Report:

- The null hypothesis of the test
- The test statistic of the test
- The p-value of the test
- A statistical decision
- A practical conclusion

```
alpha = 0.01
small_mod = glm(survived ~ sex + age + sex:age, family = "binomial", data = ptitanic_trn)
(lrs = anova(small_mod, titanic_mod, test = "LRT"))
```

```
## Analysis of Deviance Table
##
## Model 1: survived ~ sex + age + sex:age
## Model 2: survived ~ pclass + sex + age + sex:age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       296        304
## 2       294        259  2     45.1  1.6e-10 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
lrs[2, "Pr(>Chi)"] < alpha
```

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

Solution:

- The null hypothesis of the test :

  - $H_0 : \beta_1 = \beta_2 = 0$
  - $H_1 : \beta_1 \neq \beta_2 \neq 0$

- The test statistic of the test: likelihood ratio stat
  - 45.0617
- The p-value of the test
  - 1.6405×10^{-10}
- A statistical decision
  - For the small p-value $<$ alpha(0.01), we reject the null hypothesis
- A practical conclusion
  - We prefer smaller model and that there is significant relationship between survival from titanic and the passenger classes.

(c) Use the model fit in **(a)** and an appropriate statistical test to determine if an interaction between age and sex played a significant role in surviving on the Titanic. Use $\alpha = 0.01$. Report:

- The null hypothesis of the test
- The test statistic of the test
- The p-value of the test
- A statistical decision
- A practical conclusion

```
alpha = 0.01
small_mod2 = glm(survived ~ pclass + age + sex, family = "binomial", data = ptitanic_trn)
(lrs2 = anova(small_mod2, titanic_mod, test = "LRT"))
```

```
## Analysis of Deviance Table
##
## Model 1: survived ~ pclass + age + sex
## Model 2: survived ~ pclass + sex + age + sex:age
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1       295        270
## 2       294        259  1     11.4  0.00075 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
lrs2[2, "Pr(>Chi)"] < alpha
```

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

Solution:

- The null hypothesis of the test
  - $H_0 : \beta_5 = 0$
  - $H_1 : \beta_5 \neq 0$

- The test statistic of the test
  - 11.3717
- The p-value of the test
  - 0.0007
- A statistical decision
  - Looking at very small p-value of the test, we reject the null hypothesis. We prefer the bigger model.
- A practical conclusion
  - We prefer the smaller model without sex:age interaction term which does seem to be significant to survival.

(d) Use the model fit in **(a)** as a classifier that seeks to minimize the misclassification rate. Classify each of the passengers in the test dataset. Report the misclassification rate, the sensitivity, and the specificity of this classifier. (Use survived as the positive class.)

```
pred = ifelse(predict(titanic_mod, ptitanic_tst, type = "response") > 0.5, "survived", "died")
mean(pred != ptitanic_tst$survived)
```

```
## [1] 0.2172
```

```
make_conf_mat = function(predicted, actual) {
  table(predicted = predicted, actual = actual)
}

get_sens = function(conf_mat){
  conf_mat[2,2] / sum(conf_mat[,2])
}

get_spec = function(conf_mat){
  conf_mat[1,1] / sum(conf_mat[,1])
}
```

```
(conf_mat_titanic = make_conf_mat(predicted = pred, actual = ptitanic_tst$survived))
```

```
##           actual
## predicted  died survived
##   died      404      135
##   survived   27      180
```

```
get_sens(conf_mat_titanic)
```

```
## [1] 0.5714
```

```
get_spec(conf_mat_titanic)
```

```
## [1] 0.9374
```

Solution:

- Missclassification rate : 0.2172
- Sensitivity: 0.5714
- Specificity : 0.9374

Exercise 3 (Breast Cancer Detection)

For this exercise we will use data found in `wisc-train.csv` and `wisc-test.csv`, which contain train and test data, respectively. `wisc.csv` is provided but not used. This is a modification of the Breast Cancer Wisconsin (Diagnostic) dataset from the UCI Machine Learning Repository. Only the first 10 feature variables have been provided. (And these are all you should use.)

- UCI Page
- Data Detail

You should consider coercing the response to be a factor variable if it is not stored as one after importing the data.

```
library(readr)
wisc_trn = read.csv("wisc-train.csv")
wisc_tst = read.csv("wisc-test.csv")
```

```
is.factor(wisc_trn$class)
```

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

```
is.factor(wisc_tst$class)
```

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

```
wisc_trn$class = as.factor(wisc_trn$class)
is.factor(wisc_trn$class)
```

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

```
wisc_tst$class = as.factor(wisc_tst$class)
is.factor(wisc_tst$class)
```

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

(a) The response variable `class` has two levels: `M` if a tumor is malignant, and `B` if a tumor is benign. Fit three models to the training data.

- An additive model that uses `radius`, `smoothness`, and `texture` as predictors

```
wisc_add_rst = glm(class ~ radius + smoothness + texture, family = binomial, data = wisc_trn)
```

- An additive model that uses all available predictors

```
wisc_add_all = glm(class ~ ., family = binomial, data = wisc_trn)
```

- A model chosen via backwards selection using AIC. Use a model that considers all available predictors as well as their two-way interactions for the start of the search.

```
wisc_add_int = glm(class ~ . ^ 2, data = wisc_trn, family = binomial, maxit = 50)
wisc_add_int_back_aic = step(wisc_add_int, direction = "backward", k = 2, trace = 0)
```

For each, obtain a 5-fold cross-validated misclassification rate using the model as a classifier that seeks to minimize the misclassification rate. Based on this, which model is best? Relative to the best, are the other two underfitting or over fitting? Report the test misclassification rate for the model you picked as the best.

```
library(boot)
set.seed(1)
wisc_add_rst_cv = cv.glm(wisc_trn, wisc_add_rst, K = 5)$delta[1]
wisc_add_all_cv = cv.glm(wisc_trn, wisc_add_all, K = 5)$delta[1]
#cv.glm(wisc_trn, wisc_add_int, K = 5)$delta[1]
wisc_add_int_back_aic_cv = cv.glm(wisc_trn, wisc_add_int_back_aic, K = 5)$delta[1]

cv_results = data.frame("radius+smoothness+texture_model" = wisc_add_rst_cv,
           "additive_all_model" = wisc_add_all_cv,
           "interaction_model_Backward_AIC" = wisc_add_int_back_aic_cv)
print(cv_results)
```

```
##   radius.smoothness.texture_model additive_all_model
## 1                         0.07707             0.1253
##   interaction_model_Backward_AIC
## 1                         0.15
```

```
which.min(cv_results)
```

```
## radius.smoothness.texture_model
##                               1
```

```
pred = ifelse(predict(wisc_add_rst, wisc_tst, type = "response") > 0.5, "M", "B")
(mean(pred != wisc_tst$class))
```

```
## [1] 0.08955
```

Solution:

Based on the result, the `additive model with radius, smootheness, and texture` as predictor variables. Both of the other models are over-fitting. missclassification rate for the selected model is 0.0896

(b) In this situation, simply minimizing misclassifications might be a bad goal since false positives and false negatives carry very different consequences. Consider the `M` class as the "positive" label. Consider each of the probabilities stored in `cutoffs` in the creation of a classifier using the **additive** model fit in **(a)**.

```
cutoffs = seq(0.01, 0.99, by = 0.01)
```

That is, consider each of the values stored in `cutoffs` as c. Obtain the sensitivity and specificity in the test set for each of these classifiers. Using a single graphic, plot both sensitivity and specificity as a function of the cutoff used to create the classifier. Based on this plot, which cutoff would you use? (0 and 1 have not been considered for coding simplicity. If you like, you can instead consider these two values.)

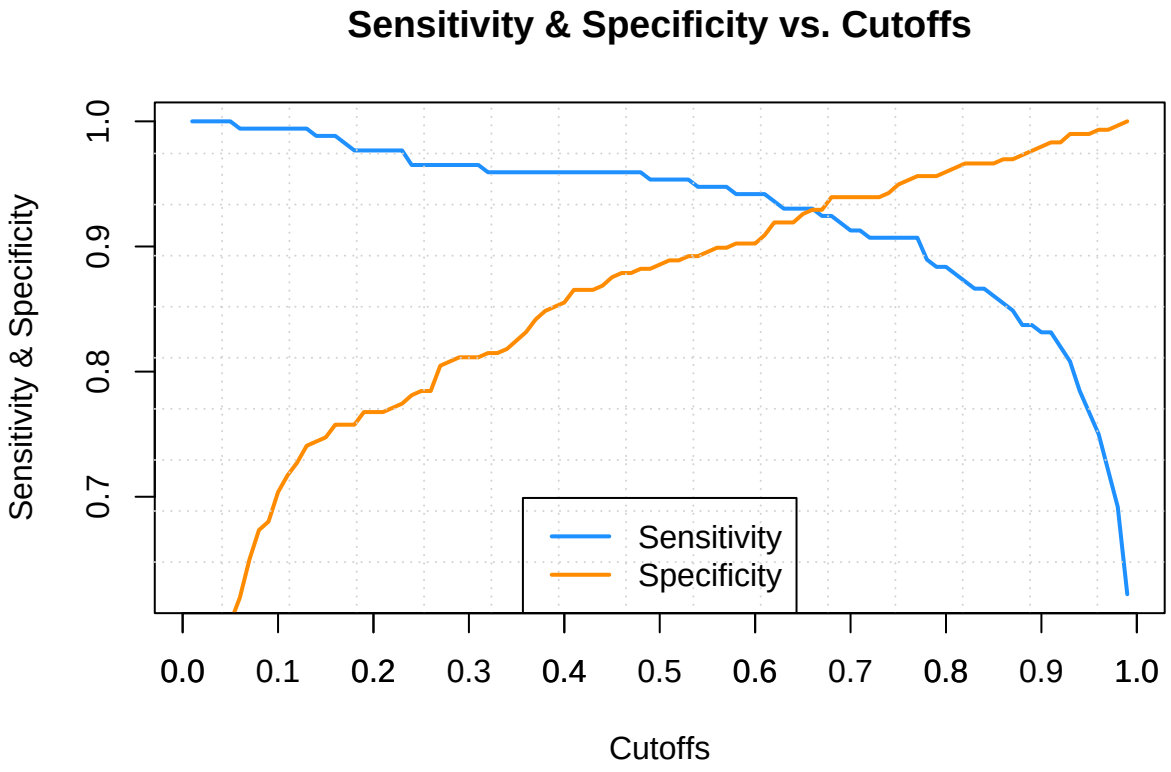
$$\hat{C}(\mathbf{x}) = \begin{cases} 1 & \hat{p}(\mathbf{x}) > c \\ 0 & \hat{p}(\mathbf{x}) \leq c \end{cases}$$

```
make_conf_mat = function(predicted, actual){
  table(predicted = predicted, actual = actual)
}

get_sens = function(conf_mat){
  conf_mat[2,2] / sum(conf_mat[,2])
}

get_spec = function(conf_mat){
  conf_mat[1,1] / sum(conf_mat[,1])
}
```

additive model with radius, smoothness, and texture

```
n = length(cutoffs)
sens = rep(0, n)
spec = rep(0, n)

result = data.frame(sens, spec, cutoffs)

for (i in 1:n){
  predictions = ifelse(predict(wisc_add_rst, wisc_tst, type = "response") > cutoffs[i], "M", "B")
  misclass = mean(predictions != wisc_tst$class)

  conf_matrix = make_conf_mat(predictions, wisc_tst$class)
  result$sens[i] = get_sens(conf_matrix)
  result$spec[i] = get_spec(conf_matrix)
}
```

```
plot(result$sens ~ result$cutoffs, type = "l", col = "dodgerblue",
     xlab ="Cutoffs",
     ylab = "Sensitivity & Specificity",
     main = "Sensitivity & Specificity vs. Cutoffs",
     lwd = 2)
lines(result$cutoffs, result$spec, col = "darkorange", lwd = 2)
grid(nx = 15, ny = 10)
axis(1, seq(0, 1, 0.1))
legend("bottom", c("Sensitivity", "Specificity"), lwd = 2 , col = c("dodgerblue", "darkorange"))
```



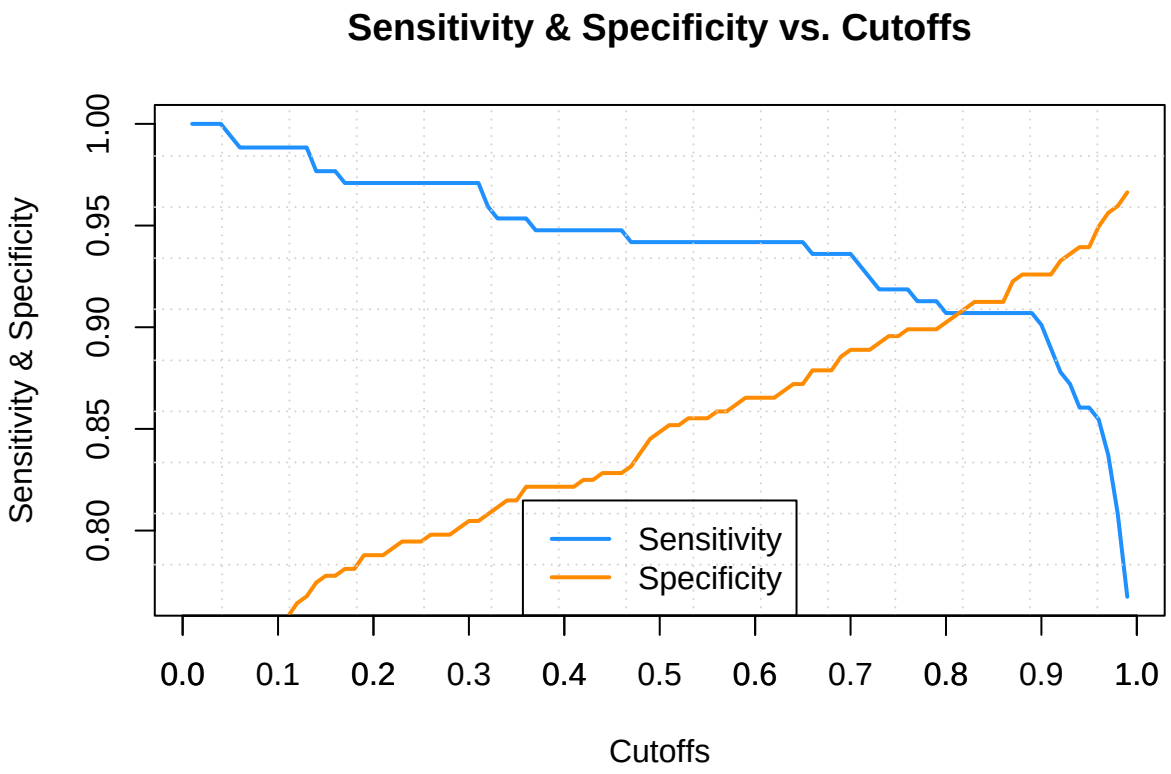
Solution:
Looking at the plot above, we can see that the intersection of the sensitivity and specificity lines is around 0.67 that is the cutoff value of choice usign additive model with radius, smoothness, and texture as predictors.

full additive model

```
n = length(cutoffs)
sens = rep(0, n)
spec = rep(0, n)

result = data.frame(sens, spec, cutoffs)

for (i in 1:n){
  predictions = ifelse(predict(wisc_add_all, wisc_tst, type = "response") > cutoffs[i], "M", "B")
  misclass = mean(predictions != wisc_tst$class)

  conf_matrix = make_conf_mat(predictions, wisc_tst$class)
  result$sens[i] = get_sens(conf_matrix)
  result$spec[i] = get_spec(conf_matrix)
}
```

```
plot(result$sens ~ result$cutoffs, type = "l", col = "dodgerblue",
     xlab ="Cutoffs",
     ylab = "Sensitivity & Specificity",
     main = "Sensitivity & Specificity vs. Cutoffs",
     lwd = 2)
lines(result$cutoffs, result$spec, col = "darkorange", lwd = 2)
```

```
grid(nx = 15, ny = 10)
axis(1, seq(0, 1, 0.1))
legend("bottom", c("Sensitivity", "Specificity"), lwd = 2 , col = c("dodgerblue", "darkorange"))
```



Solution:
Looking at the plot above, we can see that the intersection of the sensitivity and specificity lines is around 0.82 and that is the cutoff value of choice using the full additive model.