E1. Build a logistic regression model to predict the probability that a student will be in the honors class, based on information we know about the student: male, math = 65, reading = 70. What is the probability?

```
> md <- glm(hon ~ female + math + read, data = honor, family = binomial)
> predict(md, data.frame(female=0,math=65,read=70), type='response')
        1
0.5881947
>
```

E2.

```
> md <- glm(hon ~ math, data = honor, family = binomial)
> coefficients(md)
(Intercept)        math
 -9.7939421   0.1563404
> OddsRatioAB <- exp(10*0.1563404)
> OddsRatioAB
[1] 4.775048
>
```

Therefore, the answer is **B**. A's odds is 3.77 times more than B's odds.

E3.

(a)

```
> md <- glm(hon ~ female + math, data = honor, family=binomial)
> coefficients(md)
(Intercept)      female        math
-10.8059538   0.9653096   0.1642220
```

Since the female has a positive coefficient, **Mary has a higher chance** of getting into the honors class if both Mary and John have the same math score.
(b) The answer is A. See handwritten notes below.

p: prob. of being in honors class

$$\log\left(\frac{p}{1-p}\right) = \log(\text{odds})$$

$$= -10.8 + 0.97 * \text{Female} + 0.16 * \text{math}$$

① $\log(\text{John's odds}) = -10.8 + 0 + 0.16 * (\text{John's math})$

② $\log(\text{Mary's odds}) = -10.8 + 0.97 + 0.16 * (\text{Mary's math})$

$(\text{John's math} - 7)$

① − ② $= -0.97 + 0.16 * 7$

$= 0.15$

$$= \log\left(\frac{\text{John's odds}}{\text{Mary's odds}}\right) = 0.15$$

$$\frac{\text{John's odds}}{\text{Mary's odds}} = e^{0.15} = 1.16$$

John's odds is ~ 16% higher than Mary's odds of being in the honor class.

Section 4.7 Exercises

5. (a) The Bayes decision boundary being linear indicates that a linear model represents the real world well enough. For a given training data set, we expect the QDA to fit the data better (i.e., perform better on the training set) than LDA, because it is more flexible. On the test set, we expect the LDA to perform better because QDA amounts to an overfitting of the real world (which is linear).

(b) If the Bayes decision boundary is non-linear, we expect QDA to perform better on both the training set and the test set. In this case, LDA would be an underfitting of the real world.

(c) When sample size in the training sample increases, we expect the parameter estimates for both LDA and QDA models to become more accurate, and this would lead to improved test set prediction accuracies, respectively, for both methods. The question asks whether the test prediction accuracy of QDA to improve more than the improvement in the test prediction accuracy of LDA. The answer is Yes. Here are the reasons. Two cases: Case 1: If the common covariance assumption holds for the underlying real world, then LDA is an accurate model even when the training sample size is small, thus increasing the sample size would only help to improve the estimation accuracy of a few parameters (i.e., the by-class means and the common covariance matrix). In comparison, the increase in the sample size would help to improve the estimation accuracy of more parameters (i.e., the by-class means and the by-class covariance matrices) in the QDA. Thus, we expect the test performance of QDA to improve more than the test performance improvements of LDA. Case 2: If the common covariance assumption does not hold, then this implies that LDA is a fundamentally biased model, and increasing the training sample size would not help decrease LDA's bias. In this case, most of the prediction errors on the test set is due to the irreducible error. For QDA, when sample size is small, most of QDA's test set errors comes from the inaccuracy of the parameter estimates, and increasing training sample size will reduce the estimation error in the QDA's parameter estimation, therefore helps to significantly increase its test set accuracy. In both cases, we would expect that an increase in training sample size would bring about more improvements to QDA than to LDA in terms of test set prediction accuracy.

(d) Given enough training samples, QDA will learn the linearity in the decision boundary and converge to what would be produced by LDA. However, when the training sample size is small, the QDA will have a higher variance, thus likely to perform worse than LDA. So the answer is False.

6. Let p denotes the probability to get an A. The model estimate tells us that

$$\log\left(\frac{p}{1-p}\right) = -6 + 0.05 * (Hours) + 1 * (UG\ GPA)$$

(a)

```
> logodds = -6 + 0.05*40 +1*3.5
> p = exp(logodds)/(1+exp(logodds))
> p
[1] 0.3775407
> 
```

(b) Solve the equation: 1 = -6 + 0.05*Hours + 1*3.5, to get **Hours = 70.**

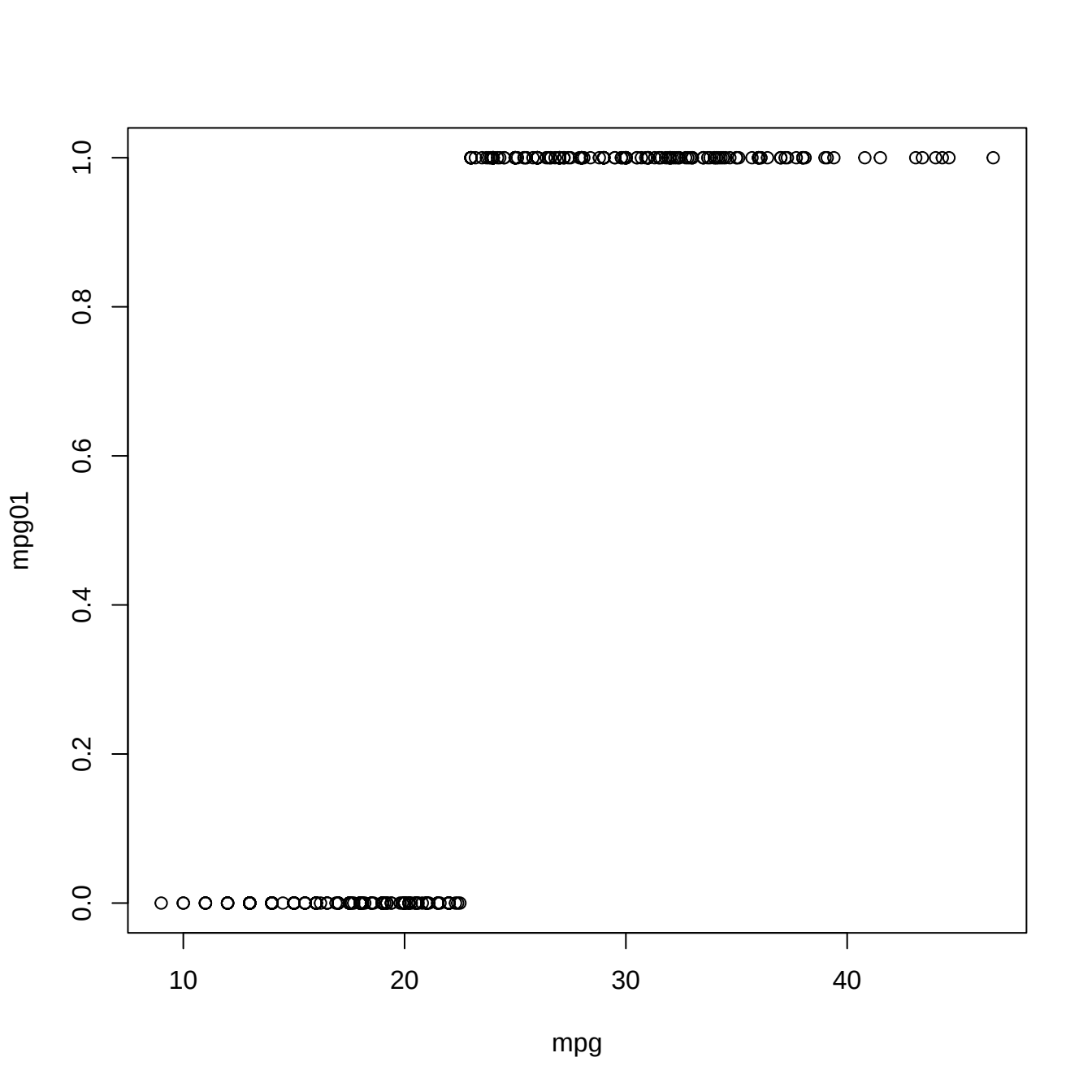
8. The logistics regression model overfits the training data, and has a higher test error rate than 1NN. So I would prefer to use 1NN for classification of new observations.

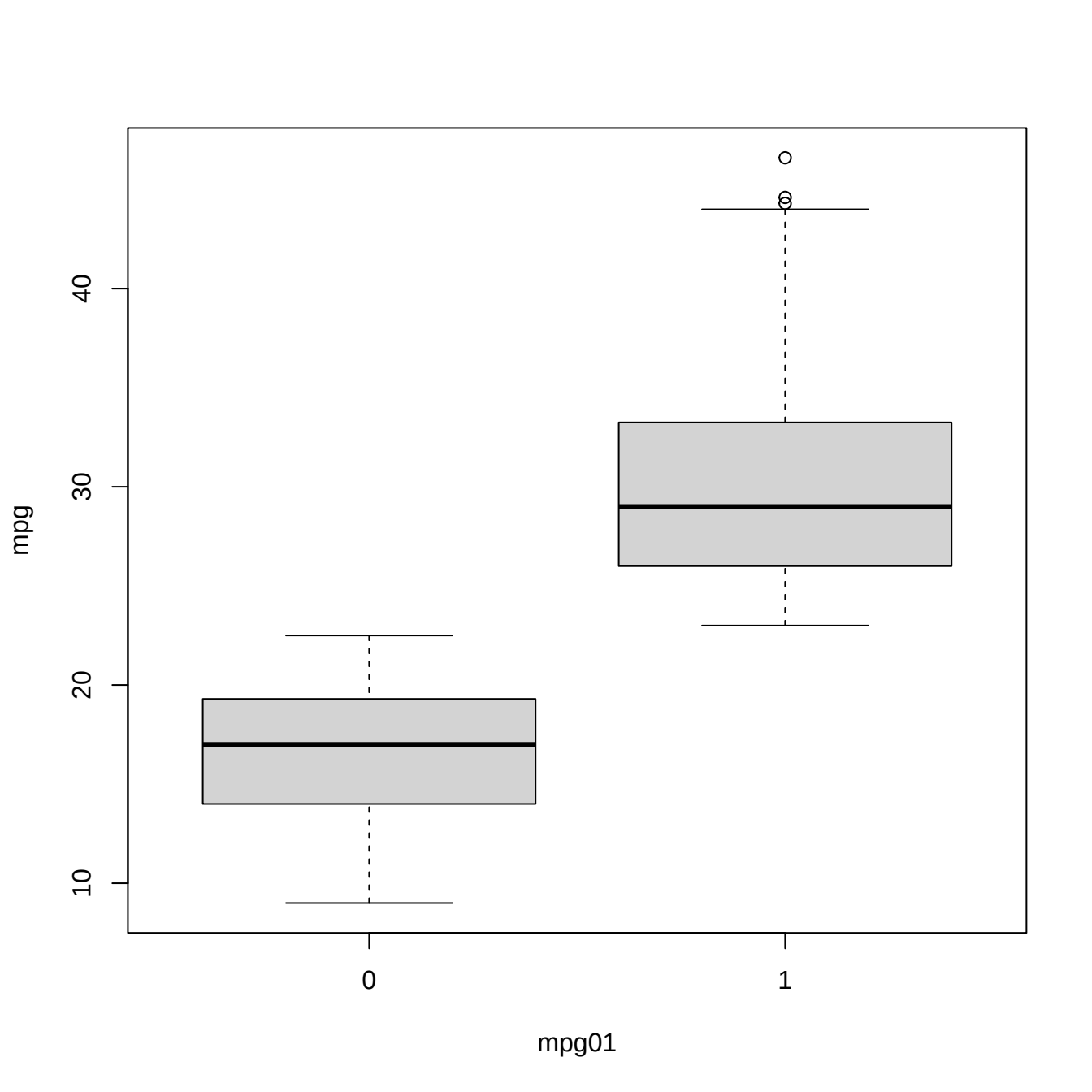
9. (a) $p = 0.37/(1+0.37) = 0.27$

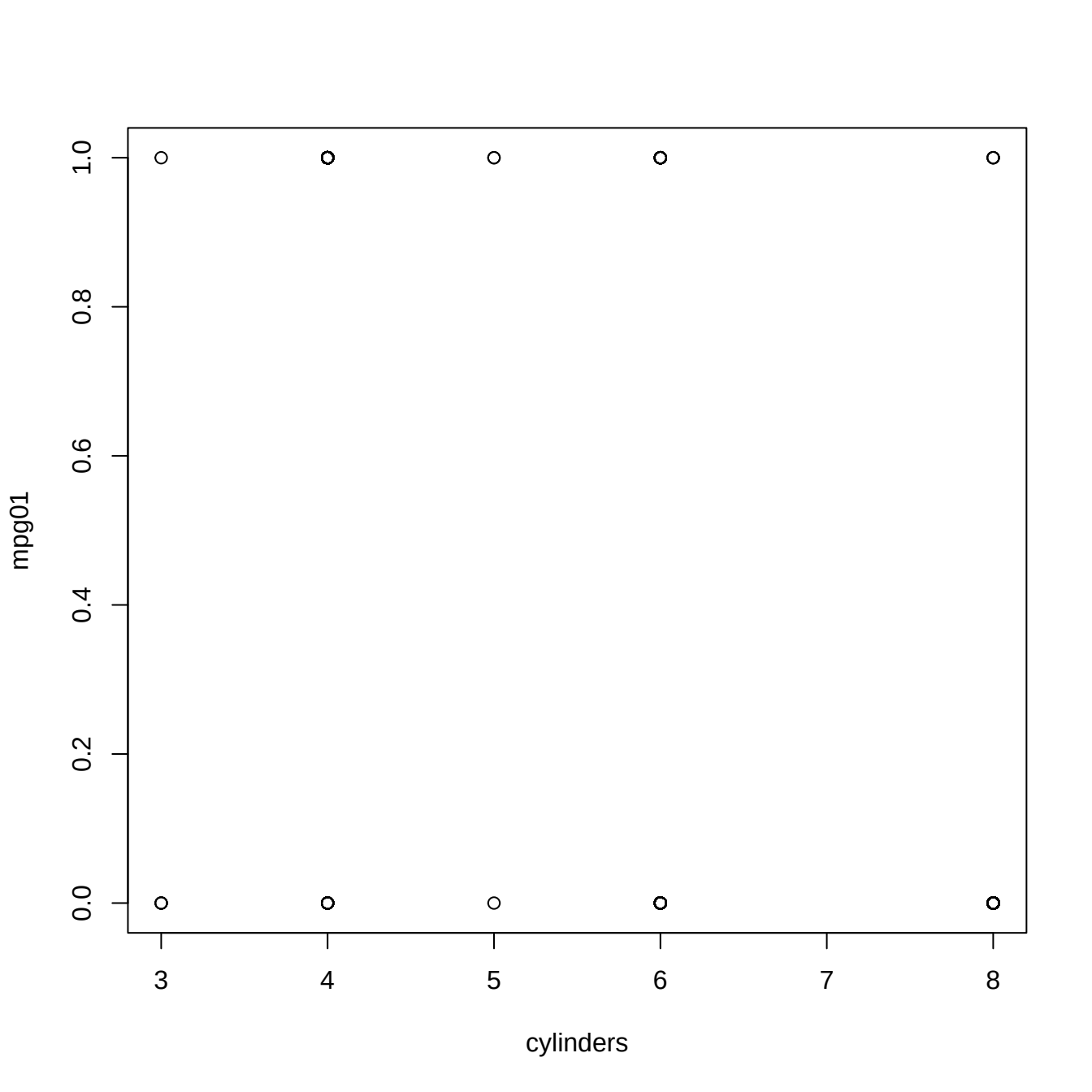
(b) $\text{odds} = p/(1-p) = 0.16/(1-0.16) = 0.19$

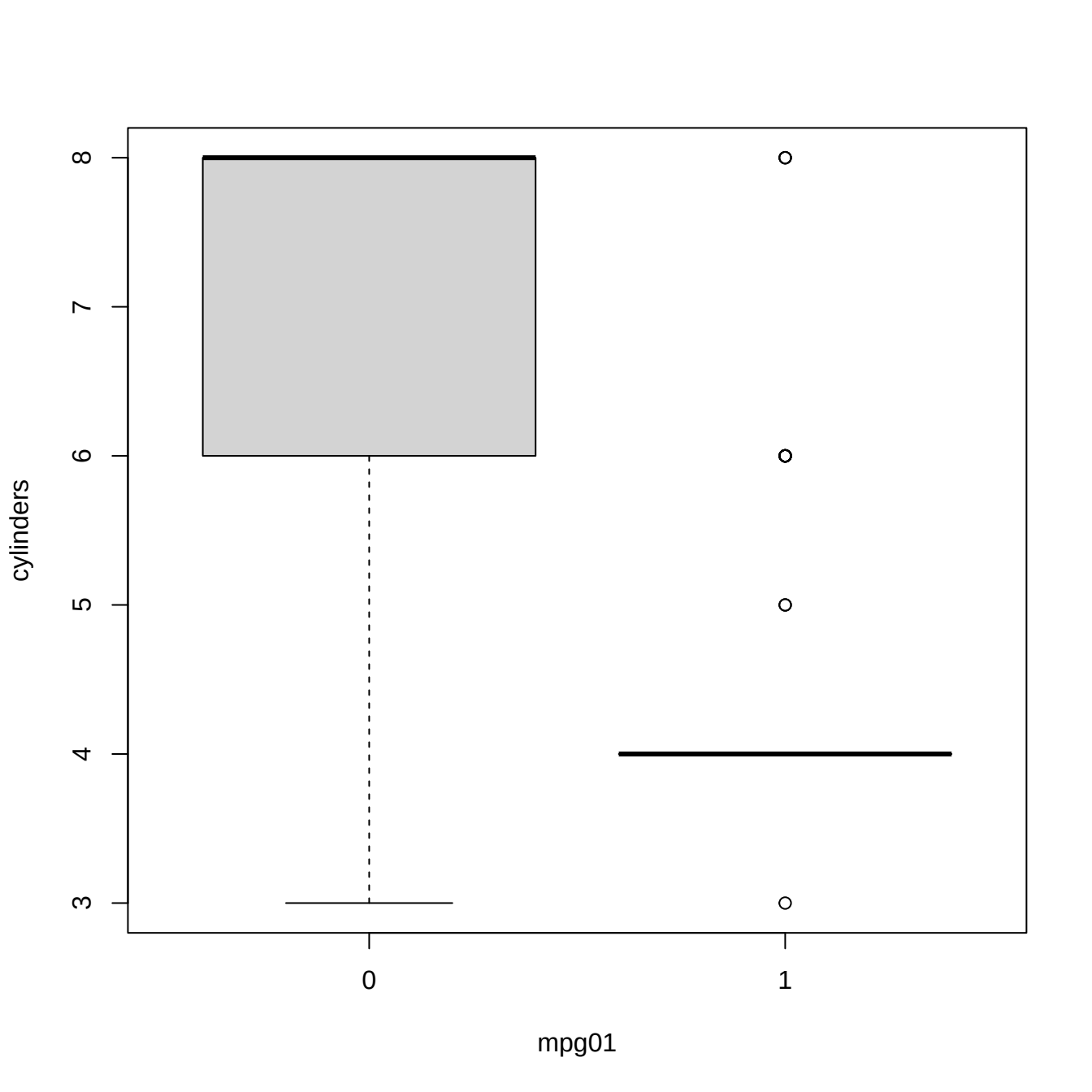
```
# Section 4.7 Exercise 11
library(ISLR)
# (a)
myAuto <- Auto
myAuto$mpg01 <- ifelse(myAuto$mpg > median(myAuto$mpg), 1, 0)
# (b)
pairs(mpg01 ~., data = myAuto)  # Note: this statement may take too long if
the sample size is big
# plotting mpg01 against each predictor individually is prefered, like this:
all_vars <- colnames(myAuto)
all_vars <- setdiff(all_vars, 'mpg01')
pdf("mpg01_vs_all.pdf")
for(v in all_vars){
  plot(myAuto[,v], myAuto[,'mpg01'], xlab=v, ylab='mpg01')
  boxplot(as.formula(paste(v, '~ mpg01')), data = myAuto)
}
dev.off()
# Observations:
# (1) mpg is highly associated with mpg01, as expected. It should be excluded
from modeling.
# (2) displacement, horsepower, weight, acceleration and year all seem to be
predictive for mpg01.
# (3) origin and name should not be used in a predictive model for mpg01

# (c)
train_idx <- sample(1:nrow(myAuto), 0.5*nrow(myAuto))
test_idx <- setdiff(1:nrow(myAuto), train_idx)
train_data <- myAuto[train_idx,]
test_data <- myAuto[test_idx,]

# (d) Let us pick displacement and weight as predictors
formula_string <- "mpg01 ~ displacement + weight"
predictors <- c('displacement','weight')
respvar <- 'mpg01'
library(MASS)
library(class)
# I will write a function to automate the process of (d) to (g). The same
approach can be useful for the Regression and Classification course projects.
evalulate_model <- function(model_name, train_data, test_data){
  if(model_name == 'lda'){
    md <- lda(as.formula(formula_string), data = train_data)
    pred_labels <- predict(md, newdata = test_data)$class
    accu <- sum(pred_labels == test_data[,respvar])/length(pred_labels)
  }
  else if(model_name == 'qda'){
    md <- qda(as.formula(formula_string), data=train_data)
    pred_labels <- predict(md, newdata = test_data)$class
    accu <- sum(pred_labels == test_data[,respvar])/length(pred_labels)
  } else if(model_name == 'lr'){
    md <- glm(as.formula(formula_string), data=train_data, family = binomial)
    pred_probs <- predict(md, newdata = test_data, type='response')
    pred_labels <- ifelse(pred_probs > 0.5, 1, 0)  # use 0.5 as the
classification threshold
    accu <- sum(pred_labels == test_data[,respvar])/length(pred_labels)
```

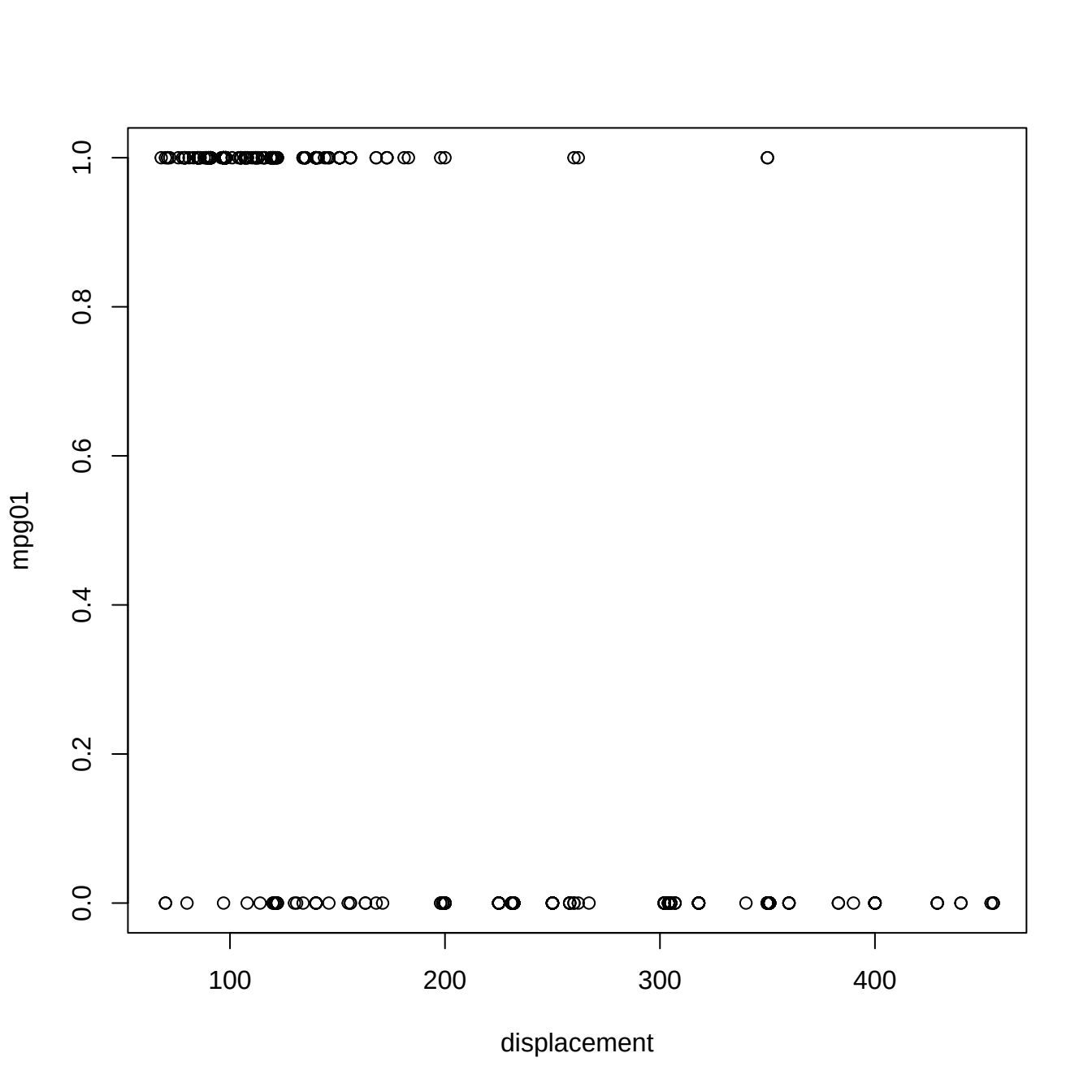
```
  } else if(model_name == 'knn'){
    # for simplicity, hard code k = 3 here
    k <- 3
    pred_labels <- knn(train_data[,predictors], test_data[,predictors],
train_data[,respvar], k)
    accu <- sum(pred_labels == test_data[,respvar])/length(pred_labels)
  }
  return(accu)
}

methods <- c('lda','qda','lr','knn')
for(method in methods){
  accu <- evalulate_model(method, train_data, test_data)
  cat(paste("Test accuracy of",method, "is", format(accu, digits = 3), "\n"))
}

# (g) Try different K values for KNN
for(k in seq(1,10)){
  pred_labels = knn(train_data[,predictors], test_data[,predictors],
as.factor(train_data[,respvar]), k)
  accu <- sum(pred_labels == test_data[,respvar])/length(pred_labels)
  cat(paste("Test accuracy of KNN with k =", k, "is", format(accu, digits =
5), "\n"))
}
```

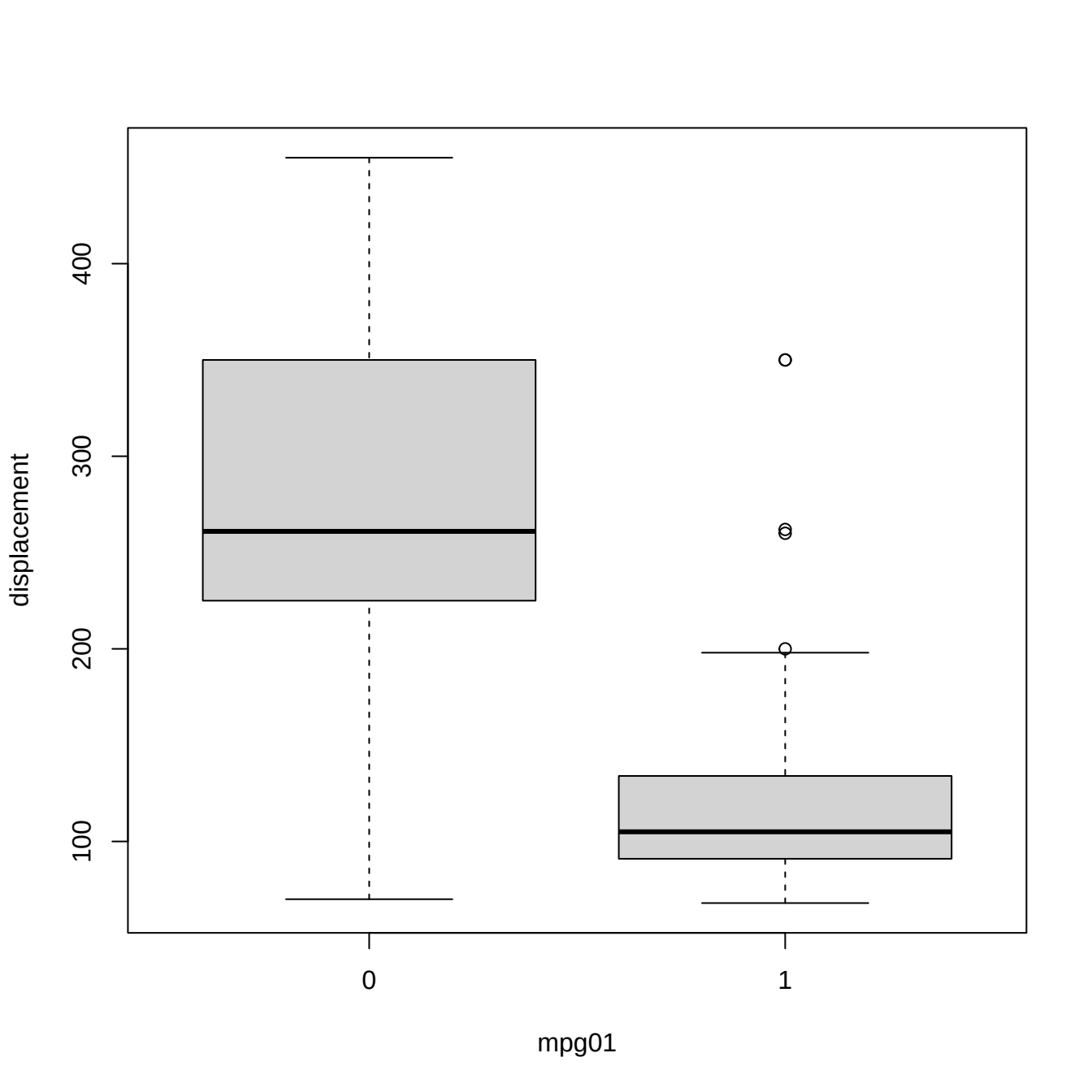

mpg01
0.0
0.2
0.4
0.6
0.8
1.0
10
20
30
40
mpg

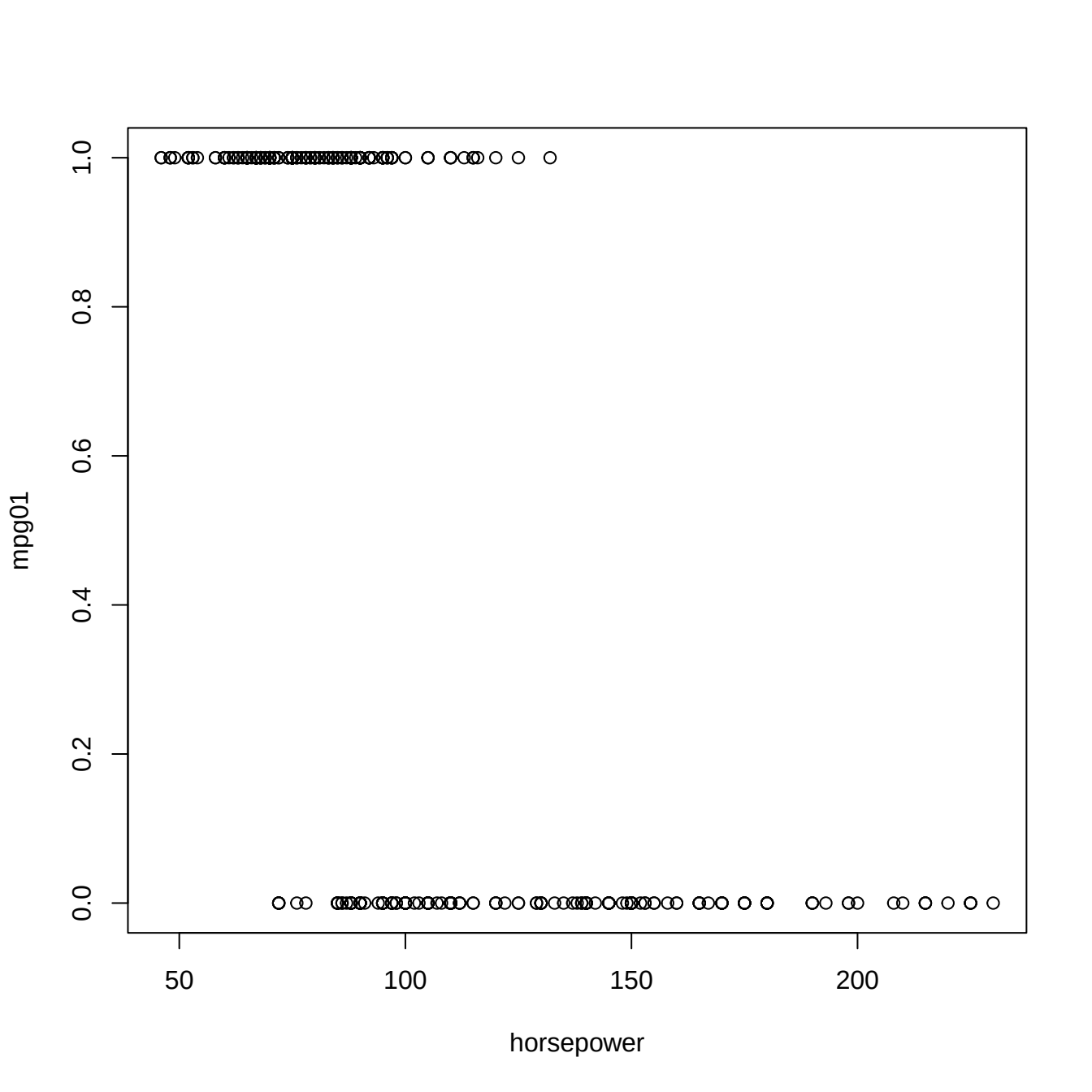

mpg
10
20
30
40
0
1
mpg01

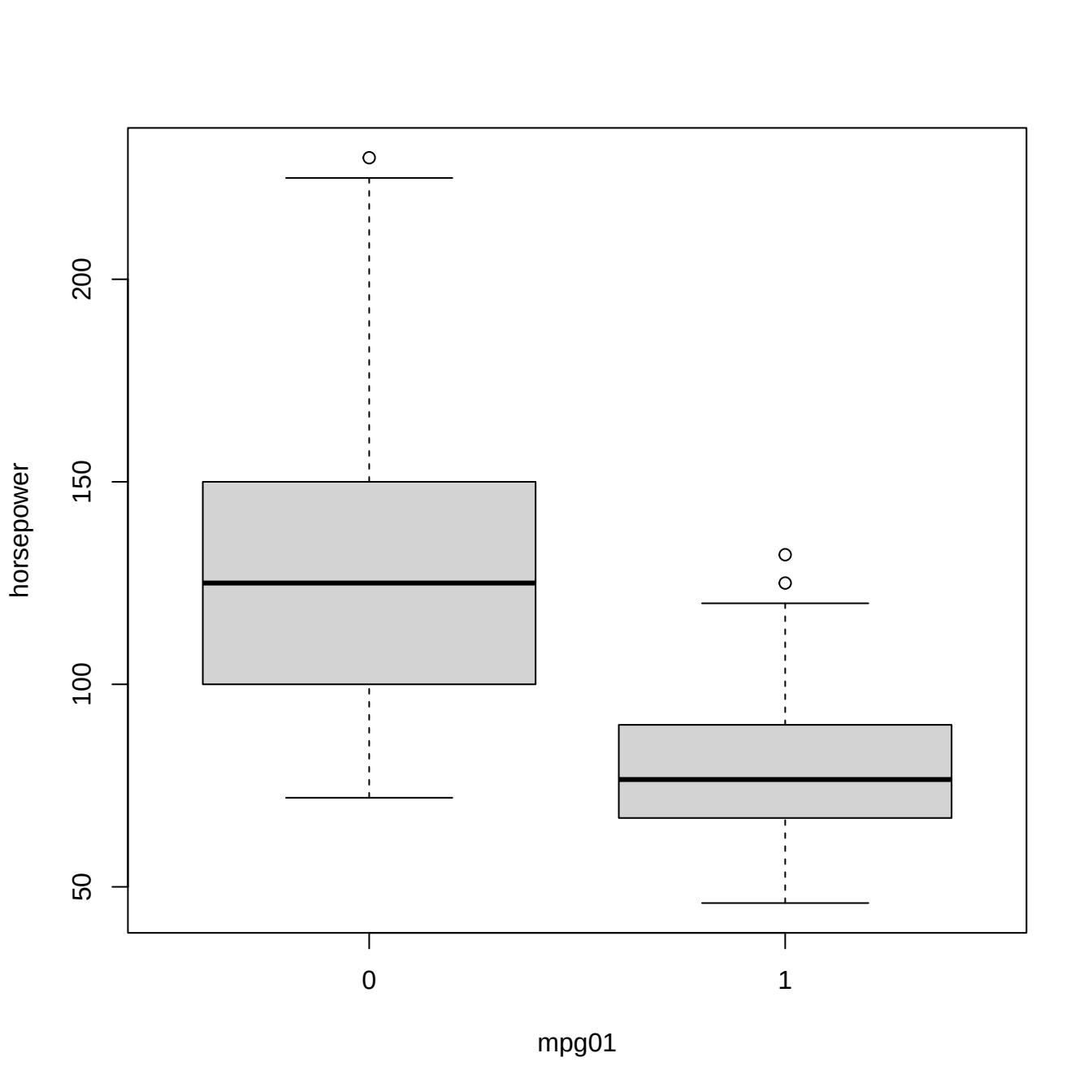

mpg01
0.0
0.2
0.4
0.6
0.8
1.0
3
4
5
6
7
8
cylinders

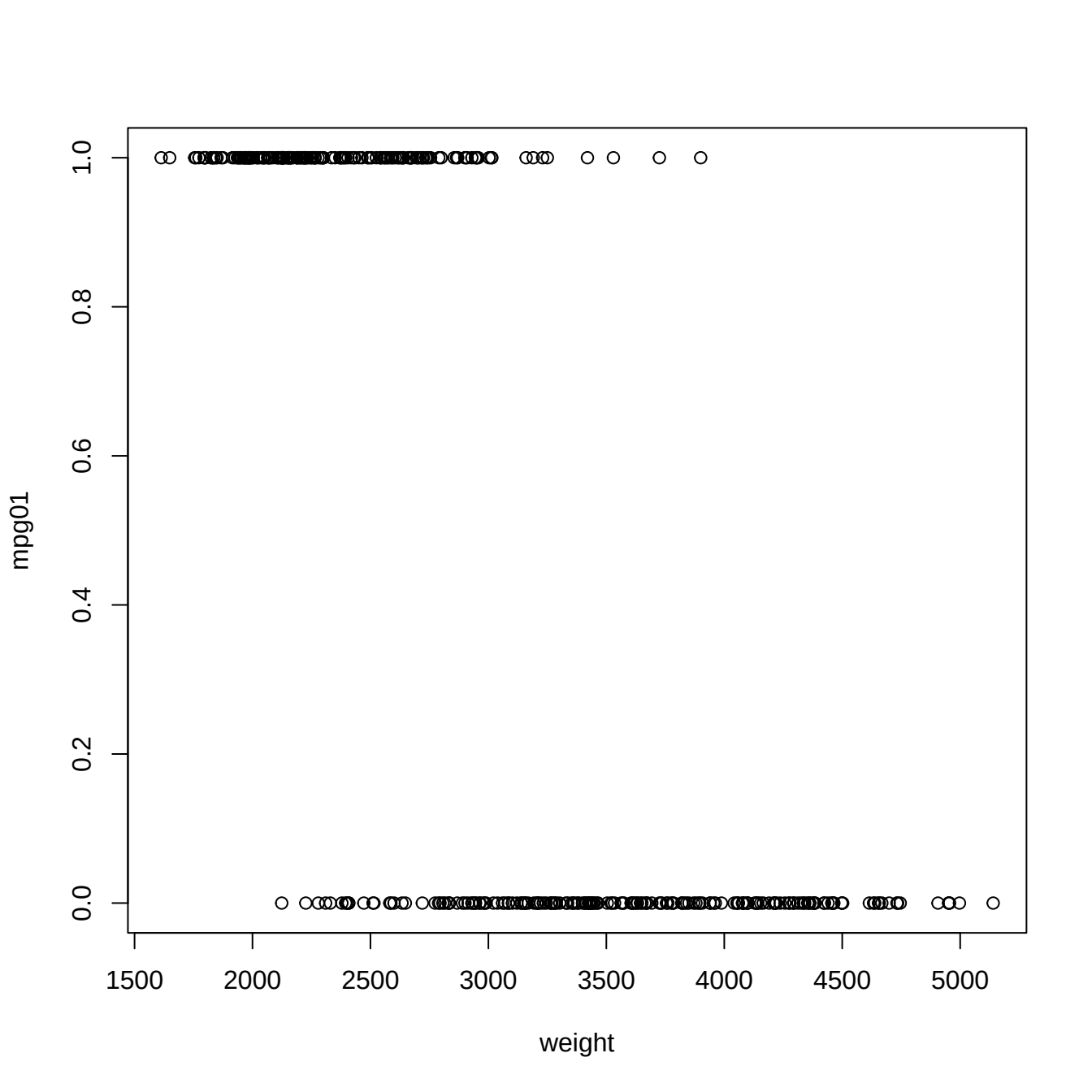

cylinders
8
7
6
5
4
3
0
1
mpg01

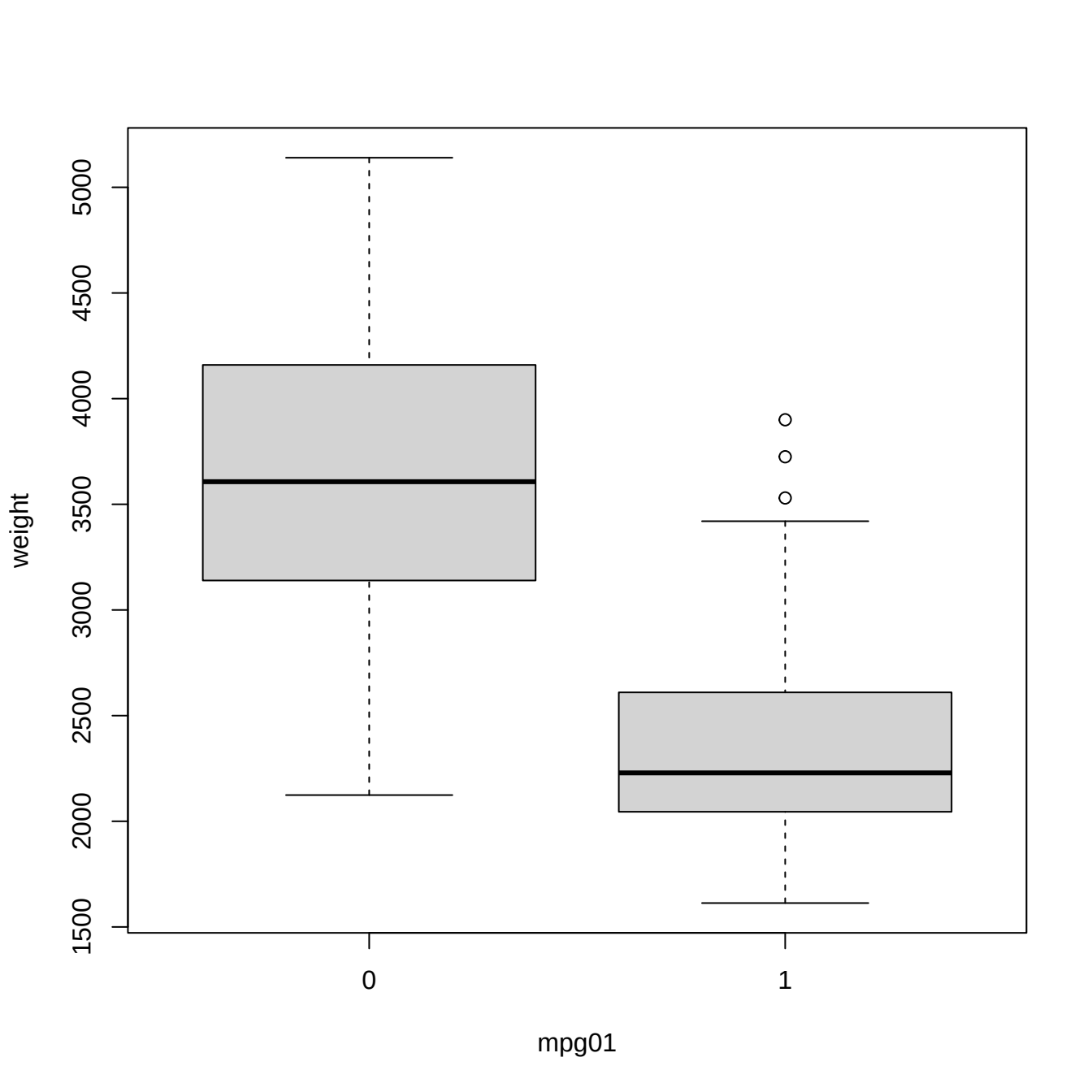

mpg01
0.0
0.2
0.4
0.6
0.8
1.0
100
200
300
400
displacement

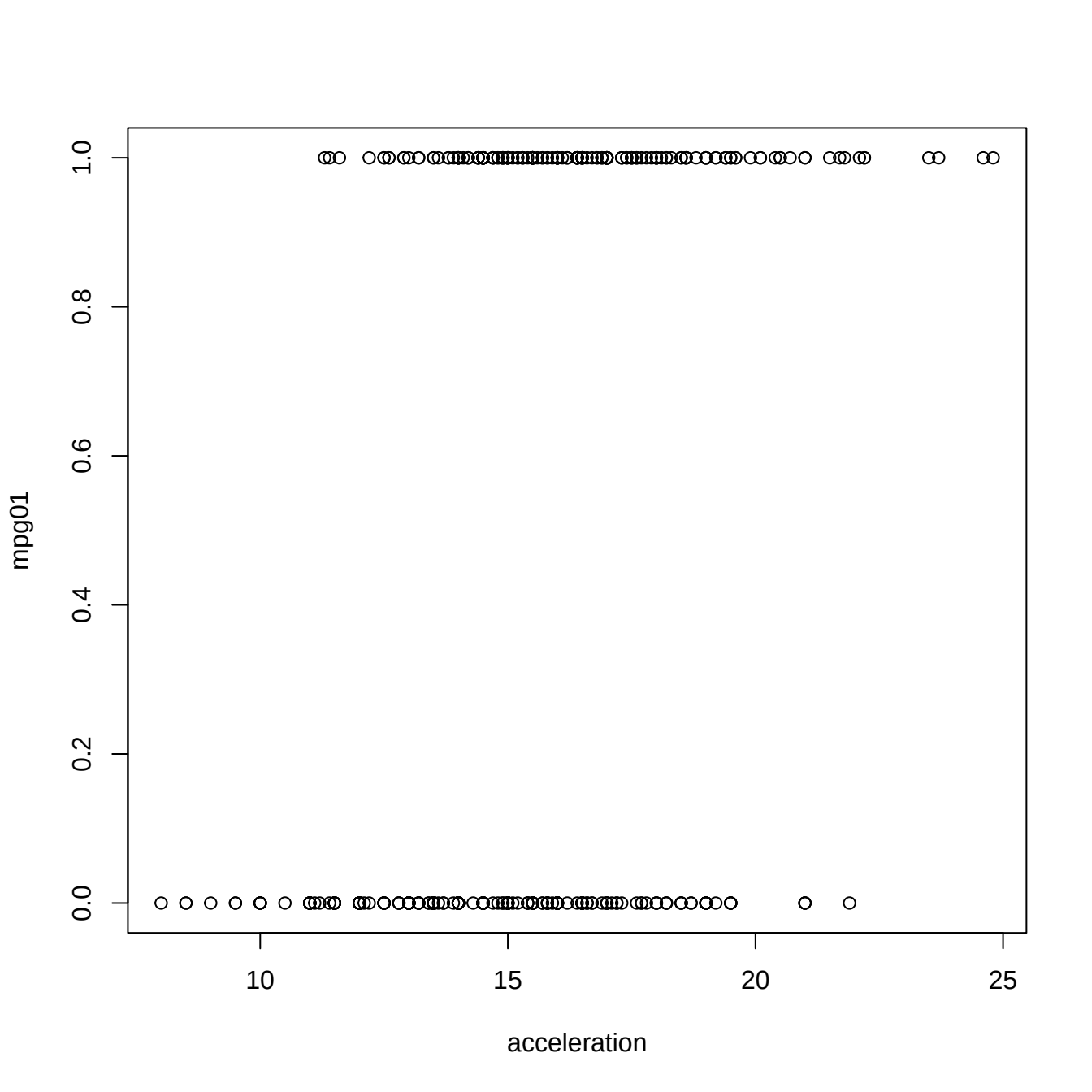

displacement
100
200
300
400
0
1
mpg01

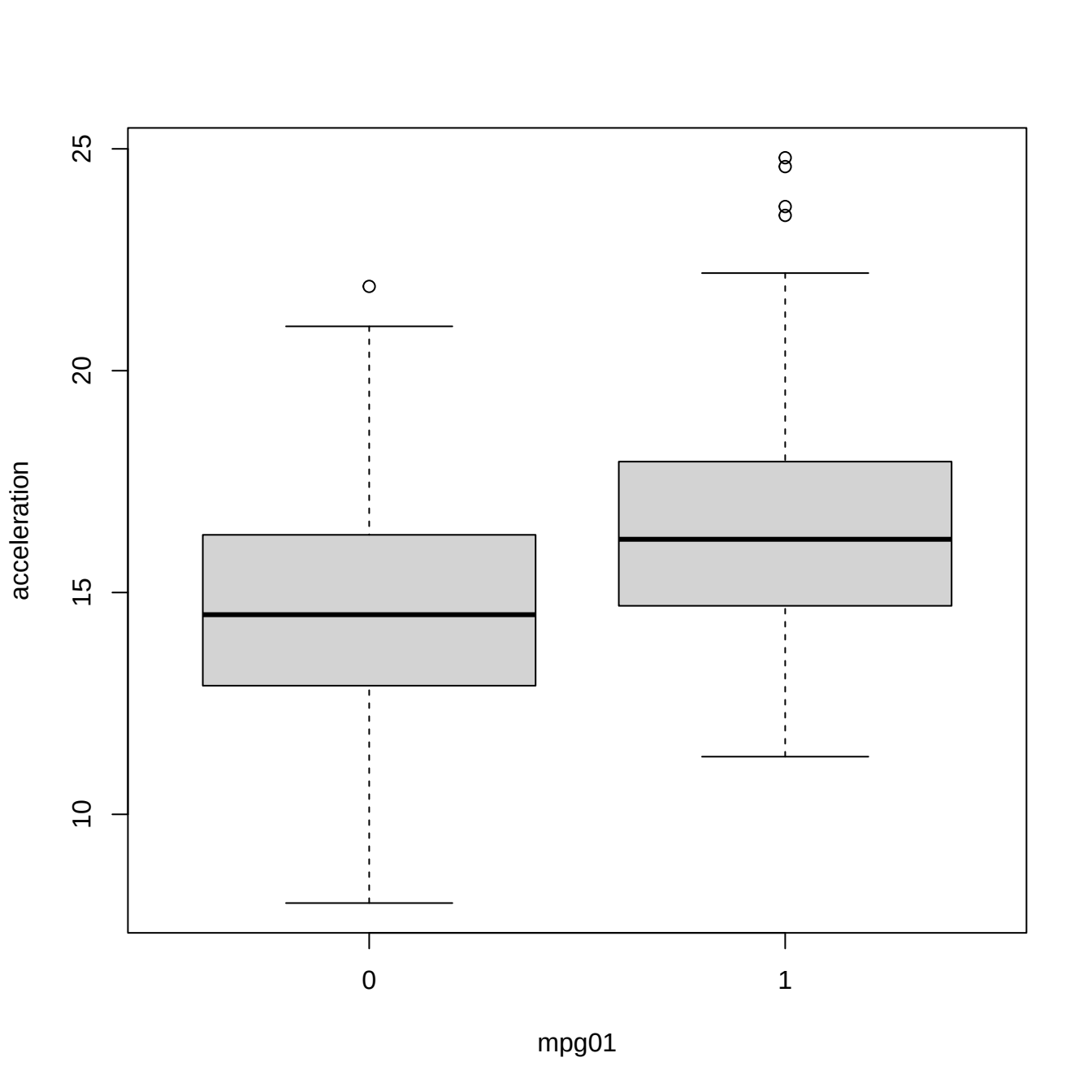

mpg01
1.0
0.8
0.6
0.4
0.2
0.0
50
100
150
200
horsepower

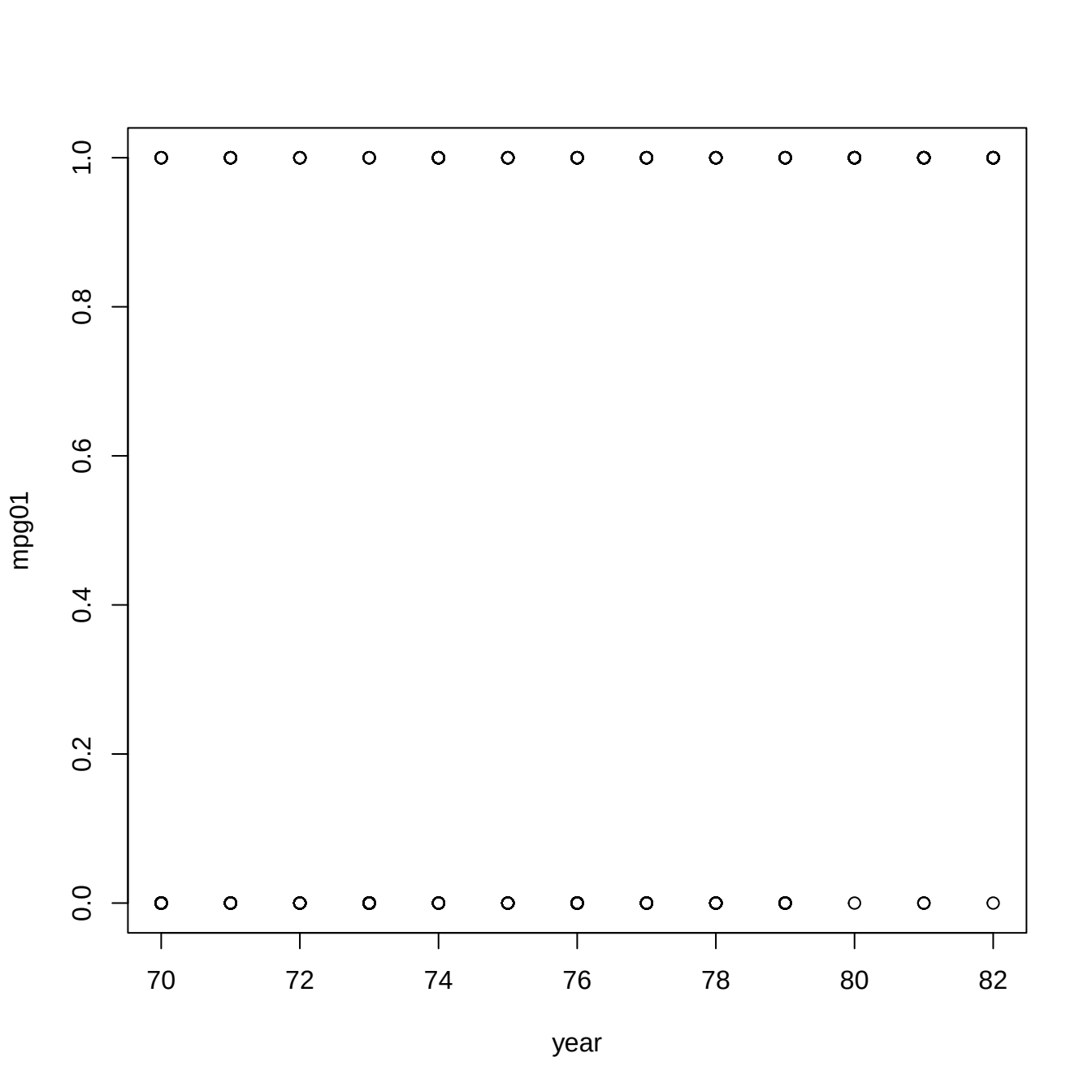

horsepower
50
100
150
200
0
1
mpg01

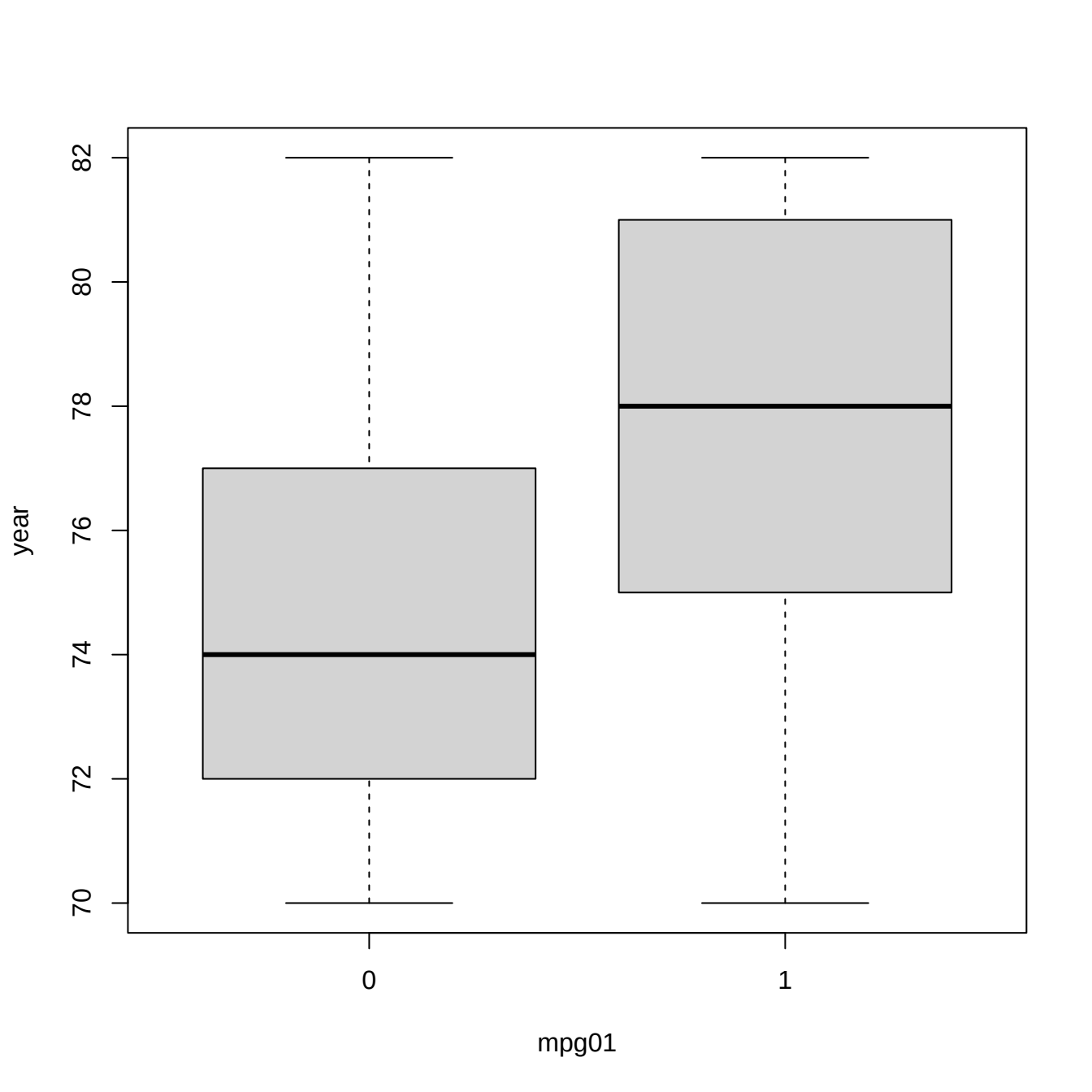

mpg01
1.0
0.8
0.6
0.4
0.2
0.0
1500
2000
2500
3000
3500
4000
4500
5000
weight

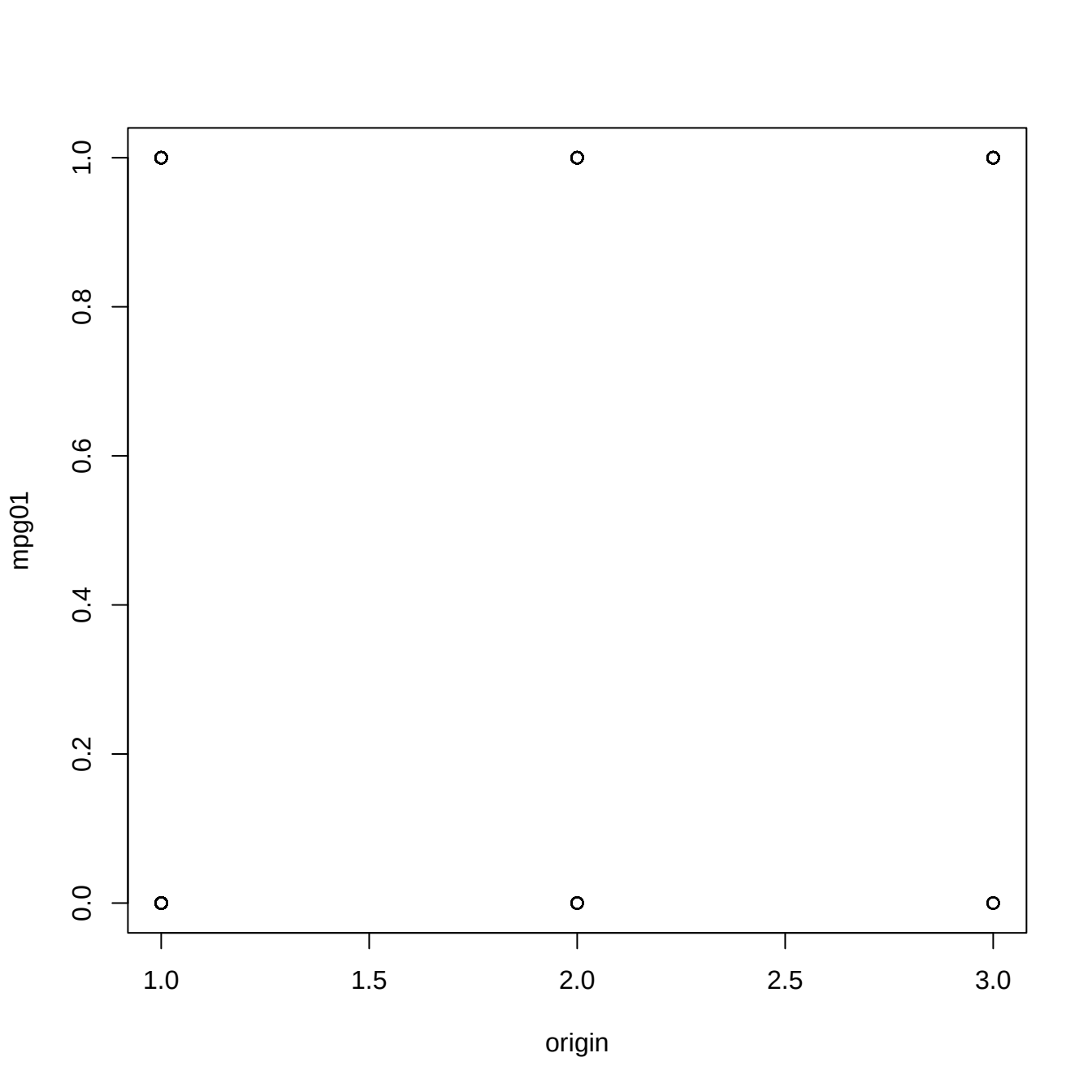

weight
1500
2000
2500
3000
3500
4000
4500
5000
0
1
mpg01

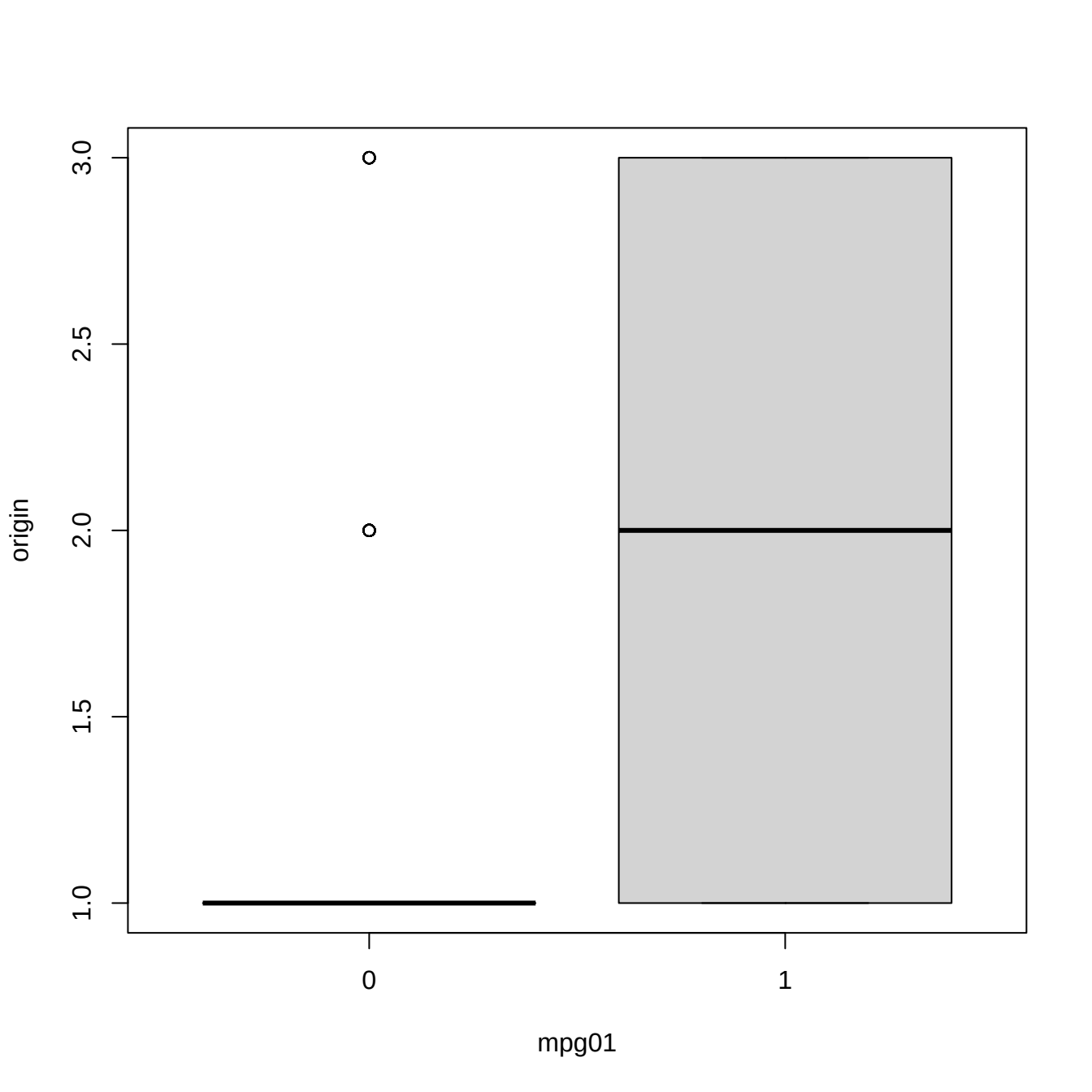

mpg01
0.0
0.2
0.4
0.6
0.8
1.0
10
15
20
25
acceleration

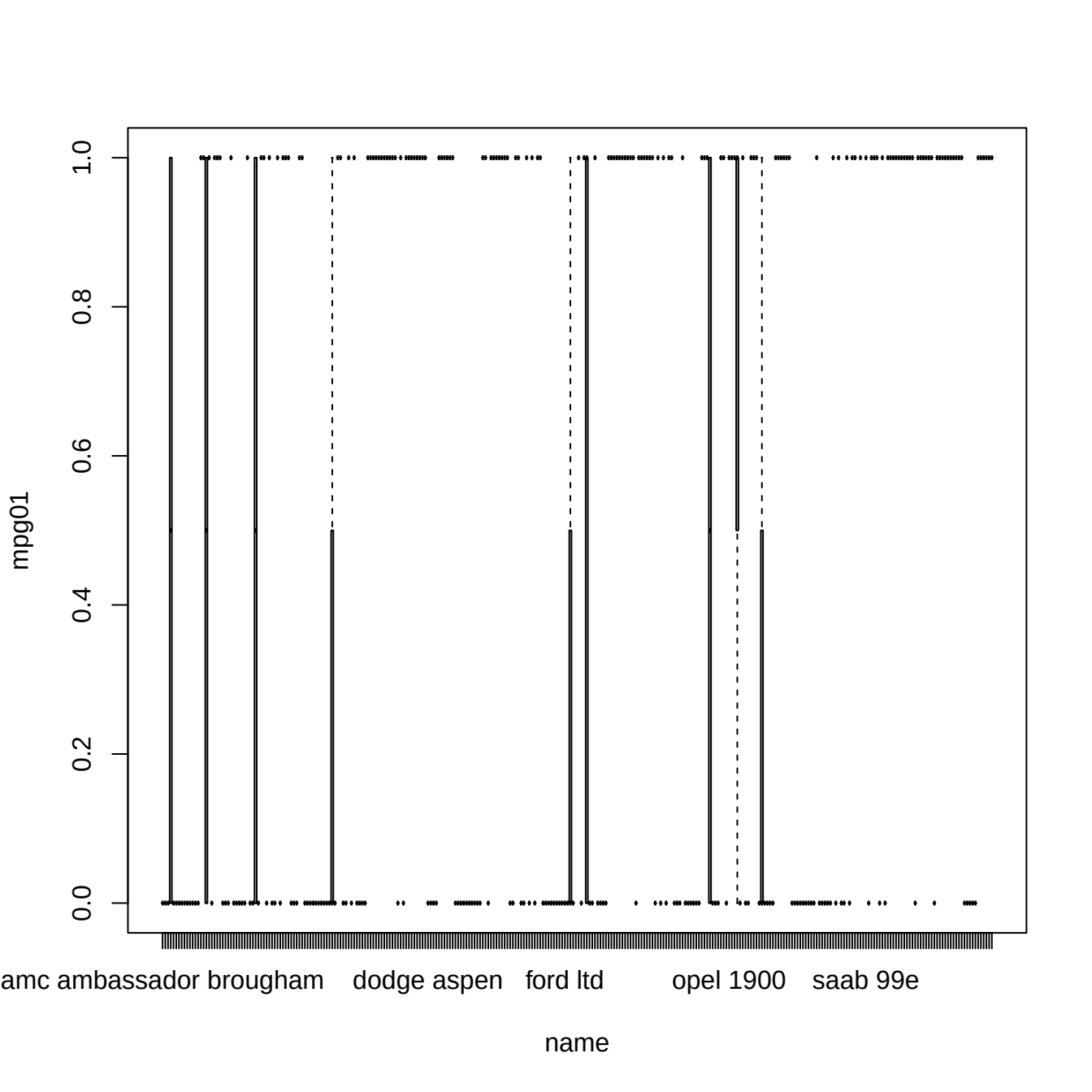

acceleration
25
20
15
10
0
1
mpg01

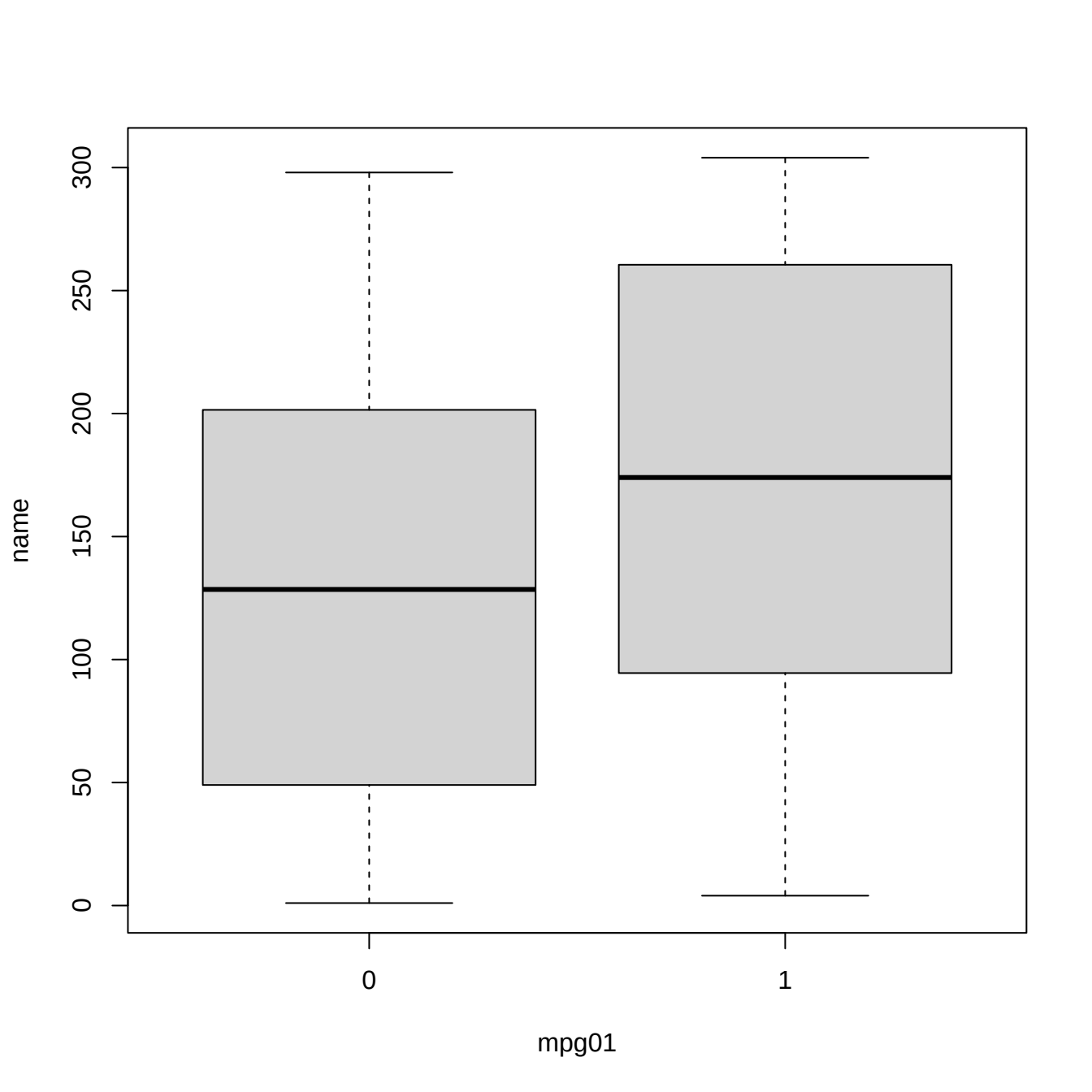

mpg01
0.0
0.2
0.4
0.6
0.8
1.0
70
72
74
76
78
80
82
year


year
82
80
78
76
74
72
70
0
1
mpg01


mpg01
0.0
0.2
0.4
0.6
0.8
1.0
1.0
1.5
2.0
2.5
3.0
origin


origin
3.0
2.5
2.0
1.5
1.0
0
1
mpg01


mpg01
0.0
0.2
0.4
0.6
0.8
1.0
amc ambassador brougham
dodge aspen
ford ltd
opel 1900
saab 99e
name


name
0
50
100
150
200
250
300
0
1
mpg01