回归分析-第三次上机

南开大学统计与数据科学学院，马东升（2212882）

2024 年 11 月 15 日

1 第（1）问

1.1 题目

对文件中的数据，考虑如下方法选择子集回归模型：
用向前法建立子集回归模型

1.2 源代码（R）

```r
# 加载readxl包
library(readxl)

# 读取Excel文件
data <- read_excel("5-7.xlsx")

head(data)

# 定义响应变量Y和自变量X
Y <- data$y
X <- data[, 2:6]

# 设置显著性水平 alpha
alpha <- 0.05

# 初始化空模型（只有截距项）
current_model <- lm(Y ~ 1, data = data)

# 存储已经选择的变量
selected_variables <- c()

# 存储每一步的结果
step_results <- data.frame(Variable = character(), F_value = numeric(),
    stringsAsFactors = FALSE)

# 获取样本数量
n <- length(Y)

# 循环逐步加入显著性最强的变量
remaining_variables <- colnames(X)  # 初始剩余变量

```

```r
# 获取当前模型的残差平方和 RSS
get_rss <- function(model) {
  return(sum(residuals(model)^2))
}

# 开始逐步选择
while (length(remaining_variables) > 0) {
  f_values <- sapply(remaining_variables, function(var) {
    # 当前模型的RSS
    rss_current <- get_rss(current_model)

    # 为每个剩余变量拟合模型并计算新的RSS
    formula <- as.formula(paste("Y ~", paste(c(selected_variables, var), collapse = "
        + ")))
    model <- lm(formula, data = data)
    rss_new <- get_rss(model)

    # 计算F值
    q <- length(selected_variables)
    f_value <- ((n -  q-2) * (rss_current - rss_new)) / rss_new
    return(f_value)
  })

  # 选择F值最大的变量（即引入当前模型后能显著提升模型的变量）
  max_f_value <- max(f_values)
  best_variable <- names(f_values)[which.max(f_values)]

  # 计算临界F值（根据F分布，使用显著性水平alpha）
  critical_f_value <- qf(1 - alpha, df1 = 1, df2 = n - length(selected_variables)- 2)
    # 记录当前步骤的结果
    step_results <- rbind(step_results, data.frame(Variable = best_variable, F_value =
        max_f_value, F_value_need = critical_f_value))
  # 如果F值大于临界F值，则接受该变量
  if (max_f_value > critical_f_value) {
    selected_variables <- c(selected_variables, best_variable)
    remaining_variables <- setdiff(remaining_variables, best_variable)  # 从剩余变量中
        删除已选的变量
    current_model <- lm(as.formula(paste("Y ~", paste(selected_variables, collapse = "
        + "))), data = data)


  } else {
    break  # 如果没有变量能显著提高F值，停止选择
  }
}

# 打印每一步的变量和F值
print(step_results)

# 打印最终的回归模型
summary(current_model)
```

1.3 统计分析结论

在 $\alpha = 0.05$ 的前提下，向前法所选入的两个自变量为 x_4, x_3（按顺序）。对应的经验回归方程为：

$$\hat{y} = 483.6703 - 24.2150x_4 + 4.7963x_3 \tag{1}$$

这一模型整体是显著的，而且各系数都是显著的，且其调整的 $R^2 = 0.8478$，也相对较高。

在 $\alpha = 0.1$ 的前提下，向前法所选入的四个自变量为 x_4, x_3, x_2, x_1（按顺序）。对应的经验回归方程为：

$$\hat{y} = 270.21013 - 21.11940x_4 + 5.33861x_3 + 2.95141x_2 + 0.05156x_1 \tag{2}$$

方程整体显著，但是 x_1 系数对应 p 值为 0.06676，稍微有点大。但其调整的 $R^2 = 0.8727$，相对来说还不错。

2 第（2）问

2.1 题目

用逐步回归法建立子集回归模型

2.2 源代码（R）

```
# 定义响应变量Y和自变量X
Y <- data$y
X <- data[, 2:6]

# 设置显著性水平 alpha
alpha <- 0.05

# 初始化空模型（只有截距项）
current_model <- lm(Y ~ 1, data = data)

# 存储已经选择的变量
selected_variables <- c()

# 存储每一步的结果
step_results <- data.frame(Variable = character(), Action = character(), F_value =
    numeric(), stringsAsFactors = FALSE)

# 获取样本数量
n <- length(Y)

# 获取当前模型的残差平方和 RSS
get_rss <- function(model) {
  return(sum(residuals(model)^2))
}

# 向后剔除和向前选择的逐步回归法
remaining_variables <- colnames(X)  # 初始剩余变量
full_model <- lm(Y ~ ., data = data)  # 全部变量模型

```

```
29 # 开始逐步选择
30 while (TRUE) {
31
32   # 向前选择: 选择最显著的变量加入模型
33   f_values_forward <- sapply(remaining_variables, function(var) {
34     rss_current <- get_rss(current_model)
35     formula <- as.formula(paste("Y ~", paste(c(selected_variables, var), collapse = "
         + ")))
36     model <- lm(formula, data = data)
37     rss_new <- get_rss(model)
38     q <- length(selected_variables)
39     f_value <- ((n - q-2) * (rss_current - rss_new)) /rss_new
40     return(f_value)
41   })
42
43   # 选择最大F值的变量进行加入
44   max_f_value_forward <- max(f_values_forward)
45   best_variable_forward <- names(f_values_forward)[which.max(f_values_forward)]
46
47
48   # 计算临界F值 (根据F分布, 使用显著性水平alpha)
49   critical_f_value<- qf(1 - alpha, df1 = 1, df2 = n - length(selected_variables)-2)
50     print(critical_f_value)
51     print(max_f_value_forward)
52
53   # 如果向前选择的F值大于临界F值, 则加入该变量
54   if (max_f_value_forward >= critical_f_value) {
55     selected_variables <- c(selected_variables, best_variable_forward)
56     remaining_variables <- setdiff(remaining_variables, best_variable_forward) # 从剩
         余变量中删除已选的变量
57     current_model <- lm(as.formula(paste("Y ~", paste(selected_variables, collapse = "
         + "))), data = data)
58     step_results <- rbind(step_results, data.frame(Variable = best_variable_forward,
         Action = "Add", F_value = max_f_value_forward))
59   print(paste("Best variable to add:", best_variable_forward))} else {print('no add')
60     # 如果没有变量能显著提高F值, 停止选择
61     break
62   }
63
64     if (length(selected_variables) == 1) {
65       next  # 跳过当前循环并进入下一次循环
66     }
67
68   # 向后剔除: 检查每个已选变量, 考虑剔除最不显著的变量
69   f_values_backward <- sapply(selected_variables, function(var) {
70     rss_current <- get_rss(current_model)
71     remaining_selected_vars <- setdiff(selected_variables, var)
72     formula <- as.formula(paste("Y ~", paste(remaining_selected_vars, collapse = " + "
         )))
73     model <- lm(formula, data = data)
74     rss_new <- get_rss(model)
75     q <- length(selected_variables)
```

```
76      f_value <- ((n - q-1) * (rss_new- rss_current )) / rss_current
77      return(f_value)
78    })
79
80    # 计算临界F值（根据F分布，使用显著性水平alpha）
81    critical_f_value_backward <- qf(1 - alpha, df1 = 1, df2 = n - length(selected_
          variables)-1)
82
83    # 选择最大F值的变量进行剔除
84    min_f_value_backward <- min(f_values_backward)
85    worst_variable_backward <- names(f_values_backward)[which.min(f_values_backward)]
86      print(critical_f_value_backward)
87      print(min_f_value_backward)
88    # 如果向后剔除的F值小于临界F值，则剔除该变量
89    if (min_f_value_backward < critical_f_value_backward) {
90      selected_variables <- setdiff(selected_variables, worst_variable_backward)
91      current_model <- lm(as.formula(paste("Y ~", paste(selected_variables, collapse = "
            + "))), data = data)
92      step_results <- rbind(step_results, data.frame(Variable = worst_variable_backward,
            Action = "Remove", F_value = max_f_value_backward))
93    print(paste("Worst variable to remove:", worst_variable_backward))} else{print('no
          remove')}
94  }
95
96  # 打印每一步的变量和F值
97  print(step_results)
98
99  # 打印最终的回归模型
100 summary(current_model)
```

2.3 统计分析结论

在 $\alpha = 0.05$ 的前提下，逐步回归法第一步选入的自变量为 x_4，第二步选入的自变量为 x_3，第三步无法选入自变量，算法结束。于是经验回归方程就是 (1)，分析也相同，这里略过。

在 $\alpha = 0.1$ 的前提下，逐步回归法第一步选入的自变量为 x_4，第二步选入的自变量为 x_3，第三步选入的自变量为 x_2，第四步选入的自变量为 x_1，第五步无法选择变量，算法结束。期间也无法剔除变量。因此经验回归方程就是 (2)，分析也相同，这里略过。

3 第（3）问

3.1 题目

应用所有可能子集回归法，建立子集回归模型，计算 RMS_q 和 AIC 值，你推荐哪一个子集回归模型，为什么？

3.2 源代码（R）

```
1  Y <- data$y
2  X <- data[, c("x1", "x2", "x3", "x4","x5")]
3
```

```r
# 获取所有可能的变量组合
all_combinations <- unlist(lapply(1:length(X), function(i) combn(names(X), i, simplify
    = FALSE)), recursive = FALSE)

# 定义一个空的结果表格，用于存储模型的结果
results <- data.frame(
  Variables = character(),
  RMSq = numeric(),
  AIC = numeric(),
  BIC = numeric(),
  stringsAsFactors = FALSE
)

# 定义计算RMSq的函数
calculate_rmsq <- function(model) {
  rss <- sum(residuals(model)^2)  # 计算残差平方和
  n <- length(model$fitted.values)  # 样本数量
  q <- length(coef(model))  # 模型的参数个数（包括截距项）
  return(rss / (n - q))  # 返回RMSq
}

# 遍历所有组合，建立回归模型，并计算 RMSq、AIC和 BIC
for (comb in all_combinations) {
  formula <- as.formula(paste("Y ~", paste(comb, collapse = " + ")))

  # 拟合回归模型
  model <- lm(formula, data = data)

  # 计算 RMSq
  rmsq_value <- calculate_rmsq(model)

  # 计算 AIC
  aic_value <- AIC(model)


  # 计算 BIC (贝叶斯信息准则)
  bic_value <- BIC(model)

  # 将结果存入表格
  results <- rbind(results, data.frame(
    Variables = paste(comb, collapse = ", "),
    RMSq = rmsq_value,
    AIC = aic_value,
    BIC = bic_value
  ))
}

# 打印结果表格
print(results)
```

3.3 统计分析结论

我们将不同子集的 RMS_q、AIC 和 BIC 值都计算出来，数据见表 1。

表 1: 不同子集的 RMS_q、AIC 和 BIC 值

子集	RMS_q	AIC	BIC
x_1	329.56104	254.3612	258.4631
x_2	538.07856	268.5783	272.6801
x_3	536.94287	268.5170	272.6189
x_4	**151.97199**	**231.9133**	**236.0152**
x_5	477.68754	265.1259	269.2278
x_1, x_2	311.01665	253.5872	259.0564
x_1, x_3	323.92609	254.7666	260.2358
x_1, x_4	150.28610	232.4953	237.9645
x_1, x_5	341.99399	256.3407	261.8098
x_2, x_3	545.33247	269.8721	275.3413
x_2, x_4	157.81262	233.9125	239.3817
x_2, x_5	492.48494	266.9161	272.3853
x_3, x_4	**79.78193**	**214.1313**	**219.6005**
x_3, x_5	357.73774	257.6459	263.1150
x_4, x_5	101.12920	221.0072	226.4764
x_1, x_2, x_3	276.19711	251.0066	257.8431
x_1, x_2, x_4	154.95108	234.2444	241.0809
x_1, x_2, x_5	319.94661	255.2707	262.1072
x_1, x_3, x_4	79.40632	214.8570	221.6935
x_1, x_3, x_5	314.59150	254.7812	261.6177
x_1, x_4, x_5	80.15910	215.1306	221.9671
x_2, x_3, x_4	**73.92246**	**212.7817**	**219.6182**
x_2, x_3, x_5	319.41523	255.2225	262.0590
x_2, x_4, x_5	105.16741	223.0053	229.8418
x_3, x_4, x_5	79.86099	215.0226	221.8591
x_1, x_2, x_3, x_4	**66.74578**	**210.6363**	**218.8400**
x_1, x_2, x_3, x_5	257.02747	249.7367	257.9405
x_1, x_2, x_4, x_5	79.72603	215.7897	223.9935
x_1, x_3, x_4, x_5	72.93744	213.2089	221.4127
x_2, x_3, x_4, x_5	76.46323	214.5779	222.7817
x_1, x_2, x_3, x_4, x_5	**64.39352**	**210.3616**	**219.9326**

我们发现，在题目要求的 RMS_q 和 AIC 两个评价指标下，都是 x_1, x_2, x_3, x_4, x_5 全模型表现最好，其经验回归方程如下：

$$\hat{y} = 326.37699 - 22.97522x_4 + 3.77078x_3 + 2.54722x_2 + 0.06780x_1 + 2.45572x_5 \tag{3}$$

但请注意，(3) 虽然整体是显著的，但 x_5 的系数并不显著，p 值为较大的 0.18394。

而如果以对模型复杂度惩罚更大的 BIC 来看，则 x_1, x_2, x_3, x_4 是最好的模型，其经验回归方程已在 (2) 中给出。