assignment9

2023-04-14

#HW09 #Use the R programs covered in the MultipleTesting module as a guide for HW09.

#Perform a multiple regression on all 100 predictors from the meatspec dataset in the #faraway package and extract the p-values for the coefficients of the 100 variables.

```
library(faraway)
data(meatspec,package="faraway")  #the dataset meatspec is found in the faraway package

lmod=lm(fat ~ . , data=meatspec)
summary(lmod)
```

```
##
## Call:
## lm(formula = fat ~ ., data = meatspec)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.9833 -0.4982  0.0135  0.4864  3.1727
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)       7.302      1.876   3.892 0.000168 ***
## V1            10898.047   3003.614   3.628 0.000428 ***
## V2           -12174.864   5520.233  -2.205 0.029426 *
## V3            -5953.285   8868.517  -0.671 0.503398
## V4            23229.862  15426.530   1.506 0.134875
## V5           -28386.219  19758.501  -1.437 0.153554
## V6            12748.270  17381.421   0.733 0.464794
## V7           -11422.335  11454.169  -0.997 0.320769
## V8             7102.332   7123.935   0.997 0.320892
## V9              783.655   5228.808   0.150 0.881130
## V10            3512.239   6787.803   0.517 0.605856
## V11          -10547.574  10580.407  -0.997 0.320926
## V12           34638.288  18344.772   1.888 0.061543 .
## V13          -38705.447  23098.395  -1.676 0.096542 .
## V14           28895.947  19952.355   1.448 0.150293
## V15          -13726.347  13312.307  -1.031 0.304676
## V16           -7062.769   8172.878  -0.864 0.389308
## V17            2571.597   6279.661   0.410 0.682932
## V18            5263.427   6183.397   0.851 0.396432
## V19            8860.827   8925.154   0.993 0.322914
## V20          -12149.937  15184.189  -0.800 0.425276
## V21          -19284.872  20536.132  -0.939 0.349680
## V22           36626.953  22847.592   1.603 0.111680
```

```
## V23         -11165.390  19302.712  -0.578 0.564111
## V24         -15008.939  13616.072  -1.102 0.272655
## V25          16698.992   8582.462   1.946 0.054151 .
## V26          -4891.852   5901.456  -0.829 0.408880
## V27          -6334.752   6072.685  -1.043 0.299084
## V28          24043.786   8144.906   2.952 0.003834 **
## V29         -39940.900  12335.575  -3.238 0.001578 **
## V30          33309.092  17674.622   1.885 0.062034 .
## V31         -23174.509  20974.708  -1.105 0.271539
## V32          18764.305  18959.821   0.990 0.324423
## V33          -3747.892  13458.994  -0.278 0.781158
## V34          -6671.747   9353.448  -0.713 0.477122
## V35          -5318.549   7534.861  -0.706 0.481716
## V36          10488.898   5773.159   1.817 0.071869 .
## V37          -8410.539   5892.265  -1.427 0.156202
## V38           -408.228   7970.269  -0.051 0.959241
## V39          19815.971  11338.219   1.748 0.083206 .
## V40         -23690.179  15971.026  -1.483 0.140748
## V41          29398.659  19340.032   1.520 0.131256
## V42         -32055.252  20639.448  -1.553 0.123170
## V43          11826.000  17491.895   0.676 0.500356
## V44          -9994.257  11435.392  -0.874 0.383969
## V45          23017.798   8927.175   2.578 0.011200 *
## V46          -9041.633   6218.630  -1.454 0.148705
## V47          -4846.799   3520.124  -1.377 0.171246
## V48           1536.042   4401.789   0.349 0.727764
## V49           2188.418   7363.225   0.297 0.766848
## V50         -13170.870   9829.843  -1.340 0.182947
## V51          26420.737  13371.372   1.976 0.050580 .
## V52         -23565.834  16339.395  -1.442 0.151968
## V53          -2005.210  16742.496  -0.120 0.904878
## V54          30327.413  14023.378   2.163 0.032658 *
## V55         -31802.344  10650.780  -2.986 0.003461 **
## V56          12428.271   6395.916   1.943 0.054463 .
## V57           -102.107   4676.993  -0.022 0.982620
## V58            210.251   4388.133   0.048 0.961869
## V59          -7679.011   4511.526  -1.702 0.091465 .
## V60          11590.949   3967.244   2.922 0.004199 **
## V61          -6559.639   3756.703  -1.746 0.083485 .
## V62           2533.819   3939.248   0.643 0.521370
## V63          11950.924   5296.267   2.256 0.025947 *
## V64         -18515.851   7070.171  -2.619 0.010021 *
## V65           4051.697   8539.248   0.474 0.636066
## V66            222.861   9691.472   0.023 0.981694
## V67          10439.030  10111.231   1.032 0.304061
## V68         -22570.742   9493.417  -2.378 0.019094 *
## V69          17285.149   8168.742   2.116 0.036520 *
## V70            -45.036   7357.838  -0.006 0.995127
## V71          -8134.714   6796.093  -1.197 0.233802
## V72          -1768.780   6344.295  -0.279 0.780905
## V73          15744.948   5531.706   2.846 0.005246 **
## V74         -11219.545   5666.910  -1.980 0.050132 .
## V75           5289.427   5067.718   1.044 0.298810
## V76          -2454.612   4760.274  -0.516 0.607101
```

```
## V77           740.608   4922.688   0.150 0.880677
## V78         -5730.806   5518.607  -1.038 0.301257
## V79         12166.493   6026.835   2.019 0.045863 *
## V80        -22688.979   7023.823  -3.230 0.001616 **
## V81         14991.763   8595.338   1.744 0.083824 .
## V82          3331.367   9984.910   0.334 0.739264
## V83         -6651.082  11358.746  -0.586 0.559337
## V84         -6752.949  12405.922  -0.544 0.587276
## V85         16271.066  12434.546   1.309 0.193323
## V86          5512.031  13689.180   0.403 0.687955
## V87        -21092.220  15770.171  -1.337 0.183730
## V88          9657.690  15143.593   0.638 0.524921
## V89           273.586  13103.448   0.021 0.983379
## V90         -5489.915  13927.199  -0.394 0.694180
## V91          2891.941  15479.740   0.187 0.852133
## V92         10160.850  14407.777   0.705 0.482103
## V93         -3183.235  11882.686  -0.268 0.789269
## V94         -7330.650  10959.287  -0.669 0.504913
## V95          5551.521   9450.485   0.587 0.558075
## V96         -3320.415   8349.562  -0.398 0.691613
## V97         -2512.787   7974.922  -0.315 0.753272
## V98         -5979.563   7355.289  -0.813 0.417935
## V99          8283.253   7911.765   1.047 0.297336
## V100         -101.926   3591.166  -0.028 0.977407
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.22 on 114 degrees of freedom
## Multiple R-squared:  0.9951, Adjusted R-squared:  0.9908
## F-statistic:   232 on 100 and 114 DF,  p-value: < 2.2e-16
```

```r
# Extract p-values for coefficients and exluding the intercept
p_values <- summary(lmod)$coefficients[-1, 4]
```

```r
p_values
```

```
##           V1           V2           V3           V4           V5           V6
## 0.0004283297 0.0294256650 0.5033981889 0.1348745281 0.1535538037 0.4647943207
##           V7           V8           V9          V10          V11          V12
## 0.3207693528 0.3208916975 0.8811299832 0.6058559085 0.3209258938 0.0615433570
##          V13          V14          V15          V16          V17          V18
## 0.0965422164 0.1502926730 0.3046755629 0.3893083081 0.6829323019 0.3964319860
##          V19          V20          V21          V22          V23          V24
## 0.3229136323 0.4252764439 0.3496800608 0.1116799533 0.5641105785 0.2726547758
##          V25          V26          V27          V28          V29          V30
## 0.0541511081 0.4088802236 0.2990835887 0.0038340431 0.0015775341 0.0620343696
##          V31          V32          V33          V34          V35          V36
## 0.2715386246 0.3244226674 0.7811579873 0.4771222175 0.4817158475 0.0718690219
##          V37          V38          V39          V40          V41          V42
## 0.1562020058 0.9592407305 0.0832059828 0.1407482329 0.1312561915 0.1231696311
##          V43          V44          V45          V46          V47          V48
## 0.5003562519 0.3839687639 0.0111996624 0.1487052405 0.1712463541 0.7277641070
##          V49          V50          V51          V52          V53          V54
```

```
## 0.7668477428 0.1829470928 0.0505798445 0.1519677289 0.9048779908 0.0326584621
##          V55          V56          V57          V58          V59          V60
## 0.0034611034 0.0544625942 0.9826203770 0.9618689776 0.0914645627 0.0041986807
##          V61          V62          V63          V64          V65          V66
## 0.0834846286 0.5213704458 0.0259469095 0.0100214548 0.6360656884 0.9816939643
##          V67          V68          V69          V70          V71          V72
## 0.3040610504 0.0190942824 0.0365201162 0.9951269670 0.2338017348 0.7809045001
##          V73          V74          V75          V76          V77          V78
## 0.0052463432 0.0501317942 0.2988096151 0.6071008125 0.8806771054 0.3012574109
##          V79          V80          V81          V82          V83          V84
## 0.0458632490 0.0016162837 0.0838241617 0.7392639838 0.5593371748 0.5872756605
##          V85          V86          V87          V88          V89          V90
## 0.1933231459 0.6879553379 0.1837297663 0.5249211429 0.9833787242 0.6941796989
##          V91          V92          V93          V94          V95          V96
## 0.8521330321 0.4821032694 0.7892692561 0.5049125530 0.5580750742 0.6916131495
##          V97          V98          V99         V100
## 0.7532723377 0.4179350446 0.2973355717 0.9774068543
```

```r
#Use the p-values vector to list the variables with coefficients that are significantly
#different from 0 with
#a) using no FWER adjustment (alpha=.05 for each test of hypothesis).
alpha <- 0.05
print(p_values[p_values < alpha])
```

```
##           V1           V2          V28          V29          V45          V54
## 0.0004283297 0.0294256650 0.0038340431 0.0015775341 0.0111996624 0.0326584621
##          V55          V60          V63          V64          V68          V69
## 0.0034611034 0.0041986807 0.0259469095 0.0100214548 0.0190942824 0.0365201162
##          V73          V79          V80
## 0.0052463432 0.0458632490 0.0016162837
```

```r
names(p_values[p_values < alpha])
```

```
##  [1] "V1"  "V2"  "V28" "V29" "V45" "V54" "V55" "V60" "V63" "V64" "V68" "V69"
## [13] "V73" "V79" "V80"
```

```r
# We reject these 15 features based on the p values
```

```r
#b) using the Bonferroni procedure to control the FWER alpha at 0.05.
#Get adjusted p value
p_values_bonfadjust <- p.adjust(p_values, method = "bonferroni")

# Identify variables with significant coefficients
adj_bonferroni <- p_values_bonfadjust[p_values_bonfadjust < alpha]

adj_bonferroni
```

```
##         V1
## 0.04283297
```

```r
names(adj_bonferroni)
```

```
## [1] "V1"
```

```r
# We reject only V1 based on the Bonferroni procedure
```

```r
#c) using the Holm procedure to control the FWER alpha at 0.05.

# Get the adjusted p-val
p_values_holmadjust <- p.adjust(p_values, method = "holm")

# Identify variables with significant coefficients
adj_holm <- p_values_holmadjust[p_values_holmadjust < alpha]
adj_holm
```

```
##         V1
## 0.04283297
```

```r
names(adj_holm)
```

```
## [1] "V1"
```

```r
#We again reject only V1 based on Holm procedure as well
```