**Question 10.3**

# *Using the GermanCredit data set germancredit.txt from* [*http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german*](http://archive.ics.uci.edu/ml/machine-learning-databases/statlog/german) */ (description at* [*http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29*](http://archive.ics.uci.edu/ml/datasets/Statlog+%28German+Credit+Data%29) *), use logistic regression to find a good predictive model for whether credit applicants are good credit risks or not. Show your model (factors used and their coefficients), the software output, and the quality of fit. You can use the glm function in R. To get a logistic regression (logit) model on data where the response is either zero or one, use family=binomial(link=”logit”) in your glm function call.*

Here’s one possible solution. Please note that a good solution doesn’t have to try all of the possibilities in the code; they’re shown to help you learn, but they’re not necessary.

The file solution 10.3.R shows a possible approach. The data includes a lot of categorical variables, and for some of them, not all of the values are significant. So, as the R code shows, we have new variables for each value of each categorical variable that’s significant.

Here’s a table of significant variables and their coefficients for the model (which has AIC = 673.5):

|  |  |  |
| --- | --- | --- |
| **Variable** | **Value** | **Coefficient** |
| Intercept |  | 0.293 |
| V1 | A13 | -1.47 |
| V1 | A14 | -1.53 |
| V2 |  | 0.0334 |
| V3 | A32 | -0.658 |
| V3 | A33 | -0.860 |
| V3 | A34 | -1.44 |
| V4 | A41 | -1.99 |
| V4 | A410 | -2.82 |
| V4 | A42 | -0.677 |
| V4 | A43 | -0.793 |
| V4 | A49 | -0.788 |
| V5 |  | 0.000125 |
| V6 | A65 | -1.14 |
| V8 |  | 0.264 |
| V9 | A93 | -0.628 |
| V10 | A103 | -1.15 |
| V14 | A143 | -0.770 |

# On the validation data, the model accuracy is 75.3% using a threshold of 0.5. The ROC curve (with AUC=67.3%) is shown below:



# That’s all for a threshold of 0.5 We also tested some other threshold values:

|  |  |  |
| --- | --- | --- |
| Threshold | Accuracy | AUC |
| 0.1 | 0.513 | 0.624 |
| 0.2 | 0.647 | 0.688 |
| 0.3 | 0.697 | 0.688 |
| 0.4 | 0.727 | 0.679 |
| 0.5 | 0.753 | 0.673 |
| 0.6 | 0.730 | 0.619 |
| 0.7 | 0.707 | 0.559 |
| 0.8 | 0.697 | 0.527 |
| 0.9 | 0.690 | 0.511 |

1. *Because the model gives a result between 0 and 1, it requires setting a threshold probability to separate between “good” and “bad” answers. In this data set, they estimate that incorrectly identifying a bad customer as good, is 5 times worse than incorrectly classifying a good customer as bad. Determine a good threshold probability based on your model.*

# The last part of the file solution 10.3.R tests different thresholds at a higher granularity (steps of 0.01) than above. It finds that the minimum total cost is using a threshold of 0.13 (see figure below).



# The range from 0.7-0.14 is all pretty good.

# The expected loss at 0.13 is 165 over the 300 validation data points, compared to 282 for a threshold of 0.5. So it shows that it’s important to account for the specific situation; if we just used the generic threshold of 0.5, it would be much more costly. The accuracy measure for the 0.13 threshold is 0.57, with an AUC of 0.66.

# The R output below shows the loss associated with each of the thresholds (from 0.01 to 1.00). It’s clear that it can be very costly to choose a bad threshold.

# [1] 201 198 190 183 185 178 169 169 168 174 173 166 165 171 176 178 176 187 187 182 188 189 195 190

# [25] 197 204 212 210 220 219 217 225 228 230 238 244 248 251 254 254 261 265 263 270 279 278 276 281

# [49] 277 282 287 302 307 311 309 314 312 332 336 341 339 353 358 367 372 377 386 390 403 408 407 416

# [73] 421 430 440 439 439 443 442 447 446 451 456 461 461 461 461 460 460 465 470 470 470 470 475 475

# [97] 475 475 475 475