WEEK 4 HOMEWORK – SAMPLE SOLUTIONS

IMPORTANT NOTE

These homework solutions show multiple approaches and some optional extensions for most of the questions in the assignment. You don't need to submit all this in your assignments; they're included here just to help you learn more – because remember, the main goal of the homework assignments, and of the entire course, is to help you learn as much as you can, and develop your analytics skills as much as possible!

Question 9.1

*Using the same crime data set uscrime.txt as in Question 8.2, apply Principal Component Analysis and then create a regression model using the first few principal components. Specify your new model in terms of the original variables (not the principal components), and compare its quality to that of your solution to Question 8.2. You can use the R function prcomp for PCA. (**Note** that to first scale the data, you can include scale. = TRUE to scale as part of the PCA function. Don't forget that, to make a prediction for the new city, you'll need to unscale the coefficients (i.e., do the scaling calculation in reverse)!)*

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 9.1.R shows how to calculate the principal components, run a regression, and find the resulting coefficients in terms of the original variables.

The model using the first four principal components is the following:

Crime ~ 1667 – 16.9M + 21.3So + 12.8Ed + 21.4Po1 + 23.1Po2 – 347LF – 8.3MF + 1.0Pop + 1.5NW – 1510U1 + 1.7U2 + 0.040Wealth – 6.9Ineq + 144.9Prob – 0.9Time

Its R^2 is just 0.309, much lower than the 0.803 found by the regression model found in the previous homework. But, remember that cross-validation showed a lot of overfitting in the previous homework. The R code shows the results using the first k principal components, for k=1..15:

Model	R-squared on training data	Cross-validation R-squared
Top 1 principal component	0.17	0.07
Top 2 principal components	0.26	0.09
Top 3 principal components	0.27	0.07
Top 4 principal components	0.31	0.11
Top 5 principal components	0.65	0.49
Top 6 principal components	0.66	0.46
Top 7 principal components	0.69	0.46
Top 8 principal components	0.69	0.37
Top 9 principal components	0.69	0.33
Top 10 principal components	0.70	0.30
Top 11 principal components	0.70	0.19
Top 12 principal components	0.77	0.39
Top 13 principal components	0.77	0.39
Top 14 principal components	0.79	0.47
Top 15 principal components	0.80	0.41
All original variables [from HW3]	0.80	0.41

From the table above, it seems clear that there's still over-fitting (not surprising with so few data points for the number of factors, as we saw in HW5). And it seems clear that adding the 5th principal component is important.

There's a lot more detail in the R code.

Question 10.1

Using the same crime data set uscrime.txt as in Questions 8.2 and 9.1, find the best model you can using

(a) a regression tree model, and

(b) a random forest model.

In R, you can use the tree package or the rpart package, and the randomForest package. For each model, describe one or two qualitative takeaways you get from analyzing the results (i.e., don't just stop when you have a good model, <u>but interpret it too</u>).

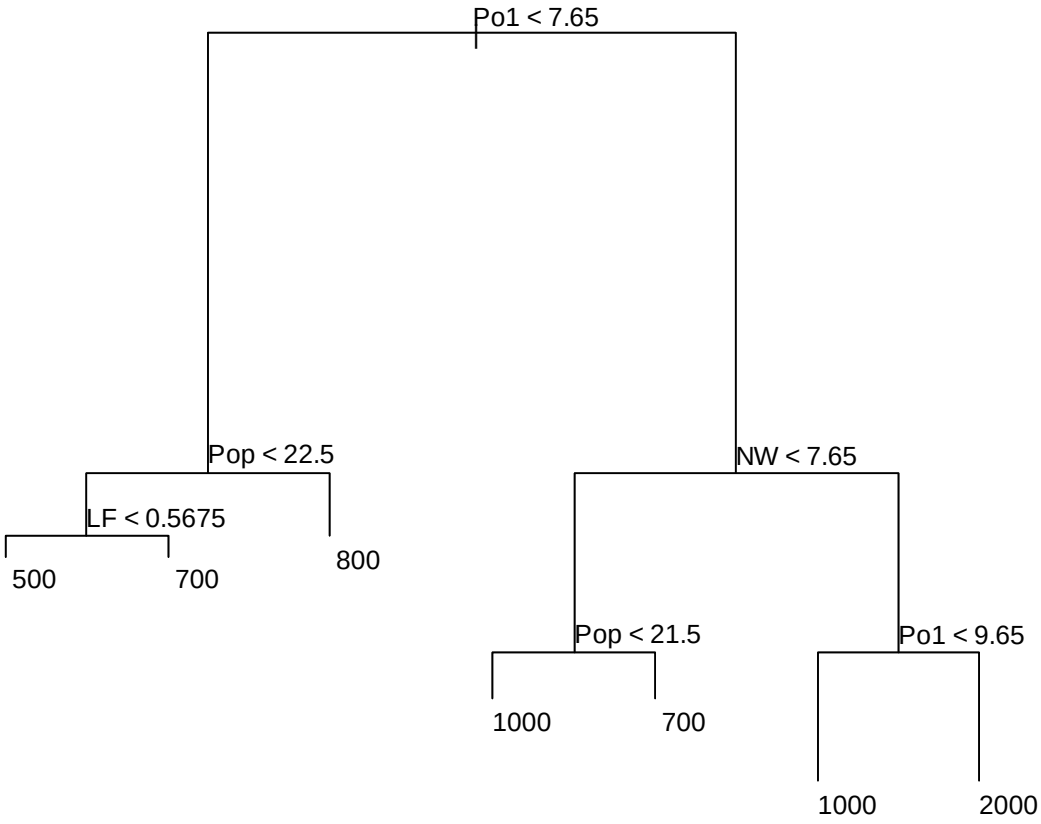
(a) <u>Regression Tree</u>

Here's one possible solution. <u>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</u>.

The file solution 10.1-a.R shows how to build the regression tree. It shows two different approaches: one using part of the data for training and part for testing, and one uses all of the data for training (because we have only 47 data points). A visualization of one of the trees is below.

The tree in the figure (sometimes called a "dendro gram") shows that four factors are used in branching: Po1, Pop, NW, and LF. Notice that Pop is used in two places in the tree. Also notice that following the

rightmost branches down the tree, Po1 is used twice in the branching: once at the top, and then again lower down.



It turns out that the model is overfit (see the R file). The R file shows a lot more possible models, including pruning to smaller trees, using regressions for each leaf instead of averages, etc.

The models show that Po1 is the primary branching factor, and when Po1 < 7.65, we can build a regression model with PCA that can account for about 30% of the variability. But for Po1 > 7.65, we don't have a good linear regression model; none of the factors are significant. This shows that we would need to either try other types of models, or find new explanatory factors.

(b) <u>Random Forest</u>

The file solution 10.1-b.R shows how to run a random forest model for this same problem. In the lessons, we saw that the random forest process avoids some of the potential for overfitting. And (as the R code shows) , cross-validation shows that it works better than the previous models we've found for this data set.

The R code also shows the same sort of qualitative behavior as we saw from the regression tree model. The random forest model too thinks that Po1 is the most important predictive factor for Crime. And, as we saw from the regression tree, the random forest model gives better predictive quality for data points where Po1 < 7.65 than it does for Po1 > 7.65. But unlike the regression tree, the random forest does have predictive value even for Po1 > 7.65.

Question 10.2

Describe a situation or problem from your job, everyday life, current events, etc., for which a logistic regression model would be appropriate. List some (up to 5) predictors that you might use.

Here's one potential suggestion. In business-to-business sales and marketing, buying-propensity models can be useful in ranking customers as high value or low value. A logistic regression model could be used to predict the probability of a customer buying or not, or (using a threshold) to classify into yes or no categories. Some of the predictors that could be used to develop such a model are:

1. Size of the customer in terms of revenue/year;
2. Average yearly spend (for the last 3 years) in the relevant product category;
3. The rating of the competitor(s) from where the customer currently buys the product (a market leader would have a higher rating as compared to a new entrant);
4. The number of years associated with the customer (is the customer long-standing? Have they bought from us for the last 5/10 years or is it a new relationship?);
5. Industry rankings for the product by various rating agencies (e.g., Gartner).

Using this model, sales teams can rationalize their efforts and concentrate their efforts on customers that are more likely to buy, rather than going broad and losing focus while catering to all customers

Question 10.3

1. *Using the GermanCredit data set germancredit.txt from 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), 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.*

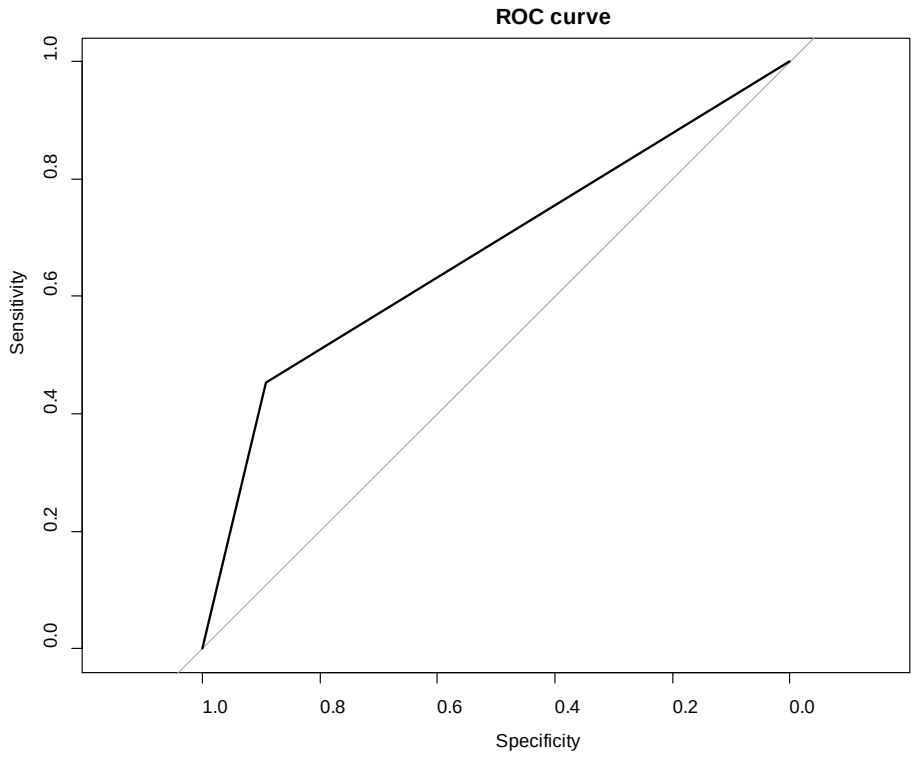
Here's one possible solution. Please note that a good solution doesn't have to try all of the possibilities in the code; they're shown to help you learn, but they're not necessary.

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:

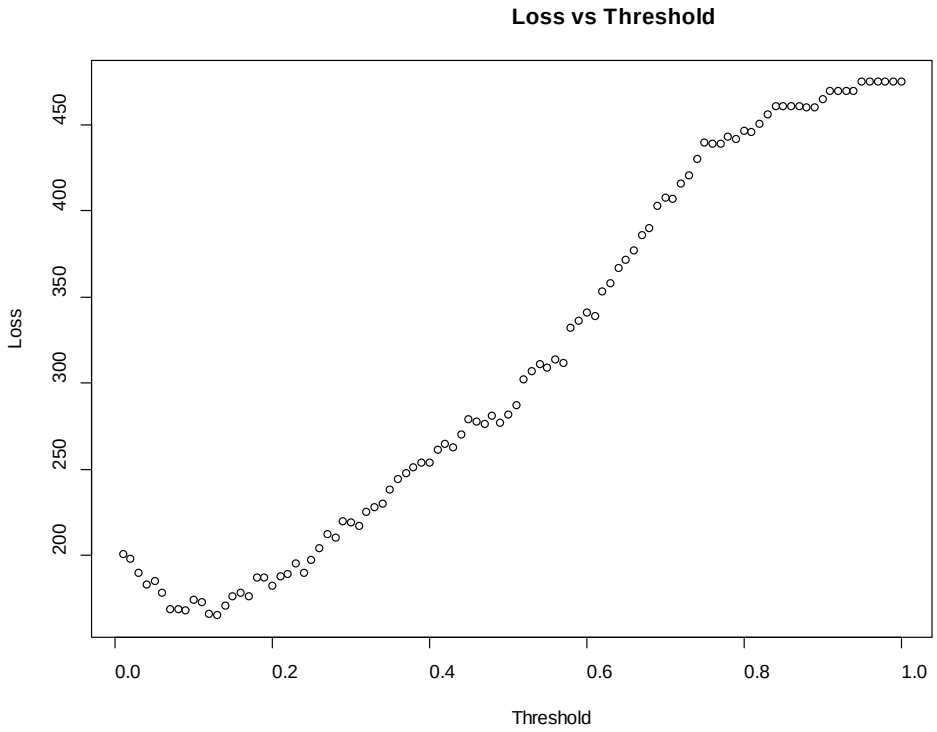


That's all for a threshold of 0.5 We also tested some other threshold values:

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

2. *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