Lab 4 Write-Up

Kurtis Potier, Nick Hiller, Zane Shango

3/25/2020

Exercise 1 (first part)

1. Execute the code above. Based on the results, rank the models from "most underfit" to "most overfit"

```
library(kernlab)
library(ggplot2)
```

```
##
## Attaching package: 'ggplot2'

## The following object is masked from 'package:kernlab':
##
##     alpha
```

```
data("spam")
tibble::as.tibble(spam)
```

```
## # A tibble: 4,601 x 58
##     make address   all num3d   our  over remove internet order  mail
##    <dbl>   <dbl> <dbl> <dbl> <dbl> <dbl>  <dbl>    <dbl> <dbl> <dbl>
##  1  0       0.64  0.64     0  0.32  0      0        0     0     0
##  2  0.21    0.28  0.5      0  0.14  0.28   0.21     0.07  0     0.94
##  3  0.06    0     0.71     0  1.23  0.19   0.19     0.12  0.64  0.25
##  4  0       0     0        0  0.63  0      0.31     0.63  0.31  0.63
##  5  0       0     0        0  0.63  0      0.31     0.63  0.31  0.63
##  6  0       0     0        0  1.85  0      0        1.85  0     0
##  7  0       0     0        0  1.92  0      0        0     0     0.64
##  8  0       0     0        0  1.88  0      0        1.88  0     0
##  9  0.15    0     0.46     0  0.61  0      0.3      0     0.92  0.76
## 10  0.06    0.12  0.77     0  0.19  0.32   0.38     0     0.06  0
## # ... with 4,591 more rows, and 48 more variables: receive <dbl>,
## #   will <dbl>, people <dbl>, report <dbl>, addresses <dbl>, free <dbl>,
## #   business <dbl>, email <dbl>, you <dbl>, credit <dbl>, your <dbl>,
## #   font <dbl>, num000 <dbl>, money <dbl>, hp <dbl>, hpl <dbl>,
## #   george <dbl>, num650 <dbl>, lab <dbl>, labs <dbl>, telnet <dbl>,
## #   num857 <dbl>, data <dbl>, num415 <dbl>, num85 <dbl>, technology <dbl>,
## #   num1999 <dbl>, parts <dbl>, pm <dbl>, direct <dbl>, cs <dbl>,
## #   meeting <dbl>, original <dbl>, project <dbl>, re <dbl>, edu <dbl>,
## #   table <dbl>, conference <dbl>, charSemicolon <dbl>,
## #   charRoundbracket <dbl>, charSquarebracket <dbl>,
## #   charExclamation <dbl>, charDollar <dbl>, charHash <dbl>,
## #   capitalAve <dbl>, capitalLong <dbl>, capitalTotal <dbl>, type <fct>
```

```r
is.factor(spam$type)
```

```
## [1] TRUE
```

```r
levels(spam$type)
```

```
## [1] "nonspam" "spam"
```

```r
set.seed(42)
# spam_idx = sample(nrow(spam), round(nrow(spam) / 2))
spam_idx = sample(nrow(spam), 1000)
spam_trn = spam[spam_idx, ]
spam_tst = spam[-spam_idx, ]

fit_caps = glm(type ~ capitalTotal,
               data = spam_trn, family = binomial)
fit_selected = glm(type ~ edu + money + capitalTotal + charDollar,
                   data = spam_trn, family = binomial)
fit_additive = glm(type ~ .,
                   data = spam_trn, family = binomial)
fit_over = glm(type ~ capitalTotal * (.),
               data = spam_trn, family = binomial, maxit = 50)

# training misclassification rate
mean(ifelse(predict(fit_caps) > 0, "spam", "nonspam") != spam_trn$type)
```

```
## [1] 0.339
```

```r
mean(ifelse(predict(fit_selected) > 0, "spam", "nonspam") != spam_trn$type)
```

```
## [1] 0.224
```

```r
mean(ifelse(predict(fit_additive) > 0, "spam", "nonspam") != spam_trn$type)
```

```
## [1] 0.066
```

```r
mean(ifelse(predict(fit_over) > 0, "spam", "nonspam") != spam_trn$type)
```

```
## [1] 0.136
```

```r
library(boot)
set.seed(1)
cv.glm(spam_trn, fit_caps, K = 5)$delta[1]
```

```
## [1] 0.2166961
```

```r
cv.glm(spam_trn, fit_selected, K = 5)$delta[1]
```

```
## [1] 0.1587043
```

```r
cv.glm(spam_trn, fit_additive, K = 5)$delta[1]
```

```
## [1] 0.08684467
```

```r
cv.glm(spam_trn, fit_over, K = 5)$delta[1]
```

```
## [1] 0.137
```

Overfit or underfit was decided based upon the change in the misclassification rate before and after cross validation. If cross validation increased the misclassification rate, then the model was overfit and vice versa.
- fit_caps
- fit_selected - fit_over - fit_additive

2. Re-run the code above with 100 folds and a different seed. Does your conclusion change?

```r
set.seed(3)
cv.glm(spam_trn, fit_caps, K = 100)$delta[1]
```

```
## [1] 0.2168394
```

```r
cv.glm(spam_trn, fit_selected, K = 100)$delta[1]
```

```
## [1] 0.1592784
```

```r
cv.glm(spam_trn, fit_additive, K = 100)$delta[1]
```

```
## [1] 0.07952213
```

```r
cv.glm(spam_trn, fit_over, K = 100)$delta[1]
```

```
## [1] 0.1490804
```

No, the conclusion does not change as the order of percentage change in the misclassification rate has not changed.

Exercise 1 (part 2)

3. Generate four confusion matrices for each of the four models fit in Part 1.

```r
make_conf_mat = function(predicted, actual) {
  table(predicted = predicted, actual = actual)
}

#############caps########
spam_tst_pred = ifelse(predict(fit_caps, spam_tst, type = "response") > 0.5,
                       "spam",
                       "nonspam")
(conf_mat_caps = make_conf_mat(predicted = spam_tst_pred, actual = spam_tst$type))
```

```
##          actual
## predicted nonspam spam
##   nonspam    2022 1066
##   spam        162  351
```

```r
(accuracy_caps <-length(which(spam_tst_pred==spam_tst$type))/length(spam_tst$type))
```

```
## [1] 0.6589836
```

```r
############selected##############
spam_tst_pred = ifelse(predict(fit_selected, spam_tst, type = "response") > 0.5,
                       "spam",
                       "nonspam")
(conf_mat_selected = make_conf_mat(predicted = spam_tst_pred, actual = spam_tst$type))
```

```
##          actual
## predicted nonspam spam
##   nonspam    2073  615
##   spam        111  802
```

```r
(accuracy_selected <-length(which(spam_tst_pred==spam_tst$type))/length(spam_tst$type))
```

```
## [1] 0.7983893
```

```r
###########additive############
spam_tst_pred = ifelse(predict(fit_additive, spam_tst, type = "response") > 0.5,
                       "spam",
                       "nonspam")
(conf_mat_additive = make_conf_mat(predicted = spam_tst_pred, actual = spam_tst$type))
```

```
##          actual
## predicted nonspam spam
##   nonspam    2057  157
##   spam        127 1260
```

```r
(accuracy_additive <-length(which(spam_tst_pred==spam_tst$type))/length(spam_tst$type))
```

```
## [1] 0.921133
```

```
###########over##############
spam_tst_pred = ifelse(predict(fit_over, spam_tst, type = "response") > 0.5,
                       "spam",
                       "nonspam")
(conf_mat_over = make_conf_mat(predicted = spam_tst_pred, actual = spam_tst$type))
```

```
##          actual
## predicted nonspam spam
##   nonspam    1725  103
##   spam        459 1314
```

```
(accuracy_over <-length(which(spam_tst_pred==spam_tst$type))/length(spam_tst$type))
```

```
## [1] 0.8439322
```

```
#############table#############
table(spam_tst$type) / nrow(spam_tst)
```

```
##
##   nonspam      spam
## 0.6064982 0.3935018
```

Model	glm	cv, 5 folds	5 fold, % change	cv, 100 folds	100 fold, % change
caps	0.339	0.2164	-36.2%	0.2168	-36%
selected	0.224	0.1582	-29.4%	0.1593	-28.9%
additive	0.066	0.07866	19.2%	0.0795	20.5%
over	0.136	0.14	2.9%	0.1491	9.6%

4. Which is the best model? Write 2 paragraphs justifying your decision. You must mention (a) the overall accuracy of each model; and (b) whether some errors are better or worse than others, and you must use the terms specificity and sensitivity. For (b) think carefully... misclassified email is a pain in the butt for users!

The best model is the additive model (a model that uses all the variables). A few different metrics were used to determine this. The first is overall accuracy. The additive model had a much higher overall accuracy than the other models. Next, sensitivity, or the true positive rate helps show how much percent of the total spam is being identified (assuming spam is positive since it is coded as 1). The additive model was a close second at this. The specificity rate shows how much percent of the total nonspam is classified properly. The additive model was a very close second as this.

Finally, an additional metric was created that presents the number of spam detected per nonspam detected as spam (positives per false positives). This metric is especially important for this application as a nonspam email that is classified as spam is the worst error case. This can lead a user to miss important information (even more important than free shipping at a site shopped at 7 years ago). The additive model and the selected model are the only models that perform well in this metric. Each model was ranked for each metric evenly and the rankings were totaled as shown in Table 3 (lower is better).

Model	Accuracy	Positives per False Positives	Sensitivity	Specificity
nothing	60.64%	0.000	0.000	1.000
caps	65.90%	2.170	0.248	0.926
selected	79.84%	7.230	0.566	0.949
additive	92.11%	9.920	0.889	0.942
over	84.39%	2.860	0.927	0.790

Rankings (lower is better)					
Model	Accuracy	Positives per False Positives	Sensitivity	Specificity	Total
nothing	5	5	5	1	16
caps	4	4	4	4	16
selected	3	2	3	2	10
additive	1	1	2	3	7
over	2	3	1	5	11

Exercise 2

```
#bank <-read.csv("bank.csv",header=TRUE) #local copy, change to web once fixed
bank <-as.data.frame(read.table("https://msudataanalytics.github.io/SSC442/Labs/data/bank.csv",
                                header = TRUE,sep = ","))
tibble::as.tibble(bank)
```

```
## # A tibble: 4,521 x 15
##      age job   marital education default balance housing loan  contact
##    <int> <fct> <fct>   <fct>     <fct>     <int> <fct>   <fct> <fct>
##  1    30 unem~ married primary   no         1787 no      no    cellul~
##  2    33 serv~ married secondary no         4789 yes     yes   cellul~
##  3    35 mana~ single  tertiary  no         1350 yes     no    cellul~
##  4    30 mana~ married tertiary  no         1476 yes     yes   unknown
##  5    59 blue~ married secondary no            0 yes     no    unknown
##  6    35 mana~ single  tertiary  no          747 no      no    cellul~
##  7    36 self~ married tertiary  no          307 yes     no    cellul~
##  8    39 tech~ married secondary no          147 yes     no    cellul~
##  9    41 entr~ married tertiary  no          221 yes     no    unknown
## 10    43 serv~ married primary   no          -88 yes     yes   cellul~
## # ... with 4,511 more rows, and 6 more variables: day <int>, month <fct>,
## #   duration <int>, campaign <int>, previous <int>, y <fct>
```

```
is.factor(bank$y)
```

```
## [1] TRUE
```

```
levels(bank$y)
```

```
## [1] "no"  "yes"
```

```
bank$job_collapsed = ifelse(bank$job %in% c('blue-collar','housemaid','technician',
                                            'services'),"blue-collar",
ifelse(bank$job %in% c("entrepreneur","admin.","management","self-employed"), "white-collar",
ifelse(bank$job %in% c("student","retired","unemployed"), "not-working",
ifelse(bank$job %in% c("unknown"), "unknown", NA))))

set.seed(15)
bank_idx = sample(nrow(bank), round(nrow(bank) / 2))
bank_trn = bank[bank_idx, ]
bank_tst = bank[-bank_idx, ]

#bank_reg <- glm(y ~ .,data = bank_trn, family = binomial)
bank_reg <- glm(y~ job_collapsed + marital + housing + loan + contact +
                  day + month + poly(duration,2) +
                  previous,data = bank_trn, family = binomial)
#Discuss coefficients in markdown
set.seed(3)
cv.glm(bank_trn, bank_reg, K=10)$delta[1]
```

```
## [1] 0.07742174
```

```
bank_tst_pred = ifelse(predict(bank_reg, bank_tst, type = "response") > 0.5,
                       "yes",
                       "no")
(conf_mat_bank_reg = make_conf_mat(predicted = bank_tst_pred, actual = bank_tst$y))
```

```
##           actual
## predicted   no  yes
##       no  1934  199
##       yes   45   83
```

```
(accuracy_bank_reg <-length(which(bank_tst_pred==bank_tst$y))/length(bank_tst$y))
```

```
## [1] 0.8920831
```

Interpretations of the Coefficients

job_collapsed shows the effect of the type of job someone has on their likeliness to deposit. Types are blue collar, not working, unknown, and white collar. A positive coefficient shows an increased likelihood to deposit based upon someone falling into that category.

marital is a categorical variable that shows whether someone is single, married, or divorced. Married and single both have negative coefficients showing that people in these categories are less likely to deposit.

housing has two categories, yes and no. This is assumed to refer to whether or not someone owns a house. Those with a house had a negative coefficient, meaning they are less likely to deposit.

loan has two categories, yes and no. This simply shows whether or not someone currently has a loan. As one might expect, those who currently have a loan are less likely to deposit, as shown by the negative coefficient on yes.

contact is a categorical variable that shows whether the call was placed to a landline, a cell phone, or not known. Those on landlines were more likely to deposit as shown by the positive coefficient. Those with

unknown methods of contact were shown to be less likely to deposit as shown by the negative coefficient on unknown.

month is a categorical variable for what month it is. The coefficient on each month shows whether deposits were more or less likely in each month. Positive coefficients show that someone is more likely to deposit in that month and negative coefficient show that someone is less likely to deposit in that month.

duration is a continuous variable that is the number of seconds the phone conversation was. A second degree polynomial was used for this variable for better fit. The linear term has a positive coefficient and the squared term has a negative coefficient. This suggests that as phone conversations become long, there are diminishing returns to the likelihood of someone depositing.

previous is a continuous variable that shows how many times someone has been called before. As many cheesy quotes would suggest, persistence pays. The positive coefficient shows that as the number of times someone has been called increases, the likelihood of someone depositing increases.