Lab 6: Logistic Regression and Support Vector Machines

Kaori Hirano

2023-06-12

```
# load packages
suppressPackageStartupMessages(library(tidyverse))
library(broom) # for tidy function
library(patchwork) # for plot placement
library(ggplot2)
suppressPackageStartupMessages(library(openintro))
suppressPackageStartupMessages(library(boot))
suppressPackageStartupMessages(library(ROCR))
suppressPackageStartupMessages(library(pROC))
suppressPackageStartupMessages(library(plotROC))
suppressPackageStartupMessages(library(e1071))
suppressPackageStartupMessages(library(glmnet))
```

Data

```
d <- email
```

Exercises

Data Visualization

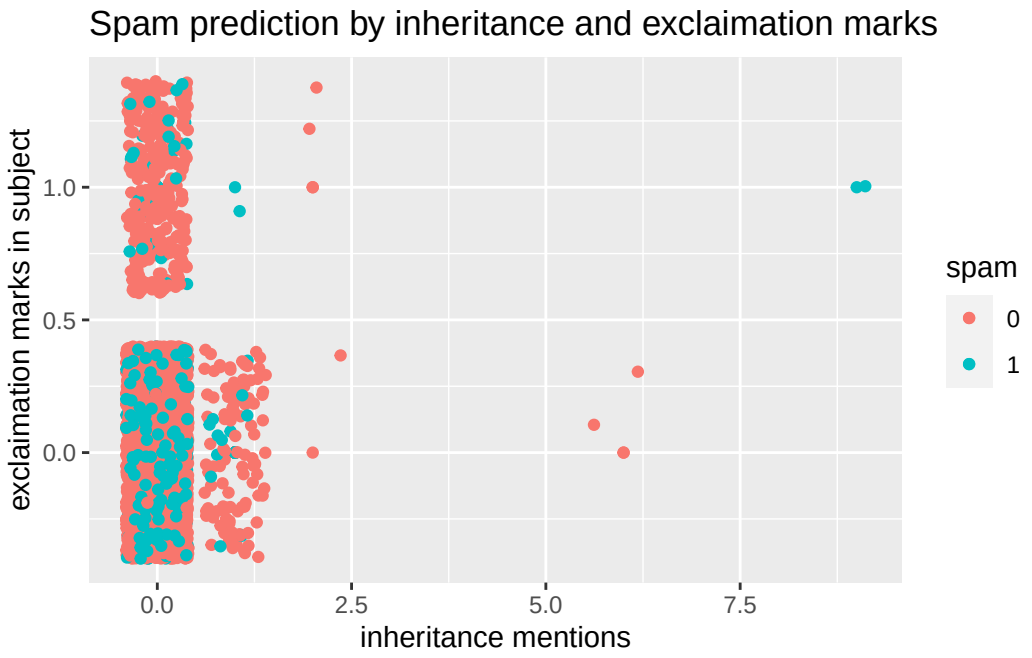
Q1

One variable I think is likely an indicator of spam is the word "inherit" This is something that shows up in a lot of spam emails because it is a common way of scamming people. Inherit also does not often show up in non-spam regular emails, so the presence of inherit/inheritance would likely indicate that the email is spam.

Another variable that would likely indicate spam or not is the number of exclaimation points in the subject. When I get advertisment emails, especially about a sale, there are always lots of exclaimation points and emojis, which would lead me to believe that emails with more exclaimation points in the subject are more likely to be spam emails.

Q2

```
ggplot(d, aes(x = inherit, y = exclaim_subj, colour = spam)) +
  geom_point() +
  geom_jitter() +
  labs(title = "Spam prediction by inheritance and exclaimation marks",
       x = 'inheritance mentions', y = 'exclaimation marks in subject')
```



The relationships are somewhat what I would expect to see. The emails with more inheritance mentions and not having exclamation points in the subject make sense as not being spam, while the ones with some mentions and more exclamation points being spam do make sense in context. I was surprised by the number of emails that included inheritance related words that were not spam, which was the biggest part of this relationship that surprised me.

The classes do not seem linearly separable because of the overlap they share.

Two-Variable Models

Q3

```
# splits data in training and test set by 70/30
set.seed(145)
train <- sample(c(TRUE, FALSE), nrow(d),
     replace = TRUE, prob=c(.7,.3))
test <- (!train)

# doing it as dataframes instead of subsetting the d frame
set.seed(145)
sample1 <- sample(c(TRUE, FALSE), nrow(d), replace=TRUE, prob=c(0.7,0.3))
train1 <- d[sample1, ]
```

```r
test1 <- d[!sample1, ]
```

Q4

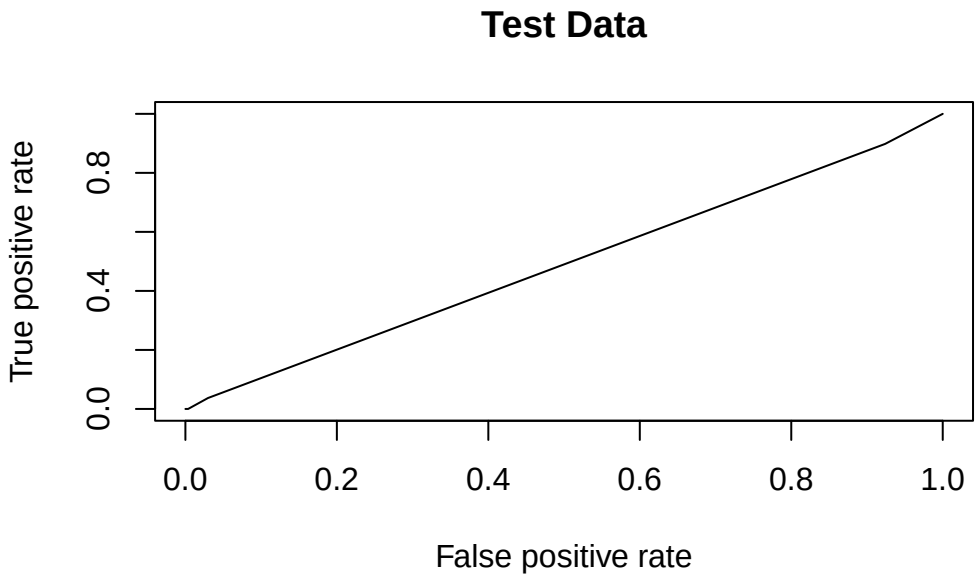
```r
# creates glm and prints summary tab
glm_fits <- glm(spam ~ inherit + exclaim_subj, train1, family = binomial)
summary(glm_fits)
```

```
Call:
glm(formula = spam ~ inherit + exclaim_subj, family = binomial,
    data = train1)

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)  -2.25180    0.06849 -32.879   <2e-16 ***
inherit       0.35075    0.19111   1.835   0.0665 .
exclaim_subj -0.14317    0.25253  -0.567   0.5708
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1708.2  on 2707  degrees of freedom
Residual deviance: 1704.7  on 2705  degrees of freedom
AIC: 1710.7

Number of Fisher Scoring iterations: 5
```

```r
# gets predictions
predicted <- predict(glm_fits, test1, type ='response')

# gets object to plot, I decided not to use this because it didn't look as nice
roc <- roc(test1$spam, predicted)
```

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases
```

```
# roc function
rocplot <- function(pred, truth, ...) {
  predob <- prediction(pred, truth)
  perf <- performance(predob, "tpr", "fpr")
  plot(perf, ...)
}

# plots the roc plot
rocplot(predicted, test1$spam, main = 'Test Data')
```




```
# prints the AUC
(auc <- auc(test1$spam,predicted))
```

```
Setting levels: control = 0, case = 1
Setting direction: controls < cases

Area under the curve: 0.4907
```

Based on the coefficients, I don't see the relationship I expected to see. The inherit one makes sense, with the more times inherit is mentioned, the more likely the email is to be spam, but the same is not true with exclamation points, which has a negative coefficient, so an email with more is actually less likely to be spam.

The AUC for the test data is .49

Q5

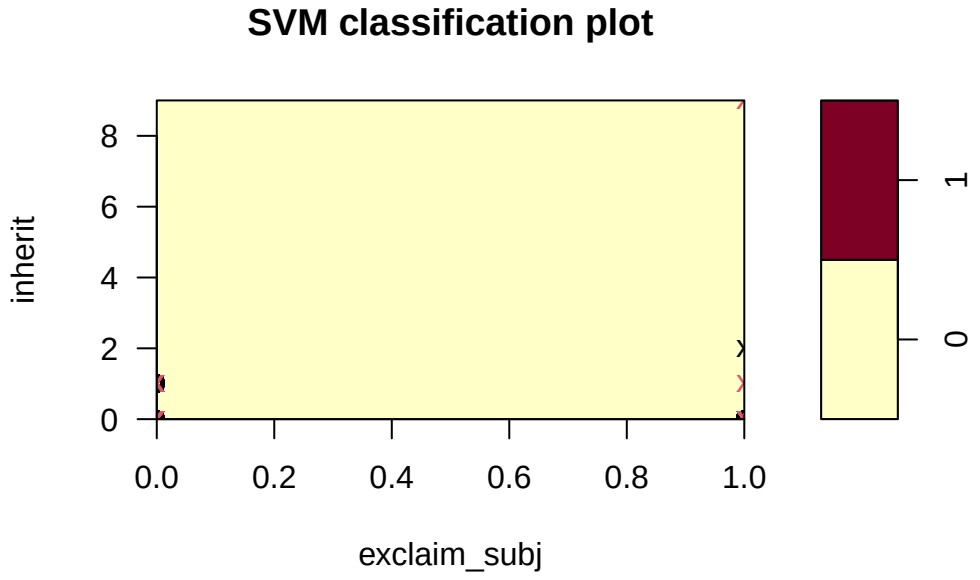
```
# data is already in the proper format with y as factor

# cross validation for tuning parameters
set.seed(246)
tune_out_two <- tune(svm, spam ~ inherit + exclaim_subj, data = d[train,], ranges =
  list(
  cost = c(0.1, 1, 10),
  gamma = c(0.5, 1, 2, 3),
  kernel = c("linear", "radial")
    ))

# prints optimal parameters
(tune_out_two$best.parameters)
```

```
  cost gamma kernel
1  0.1   0.5 linear
```

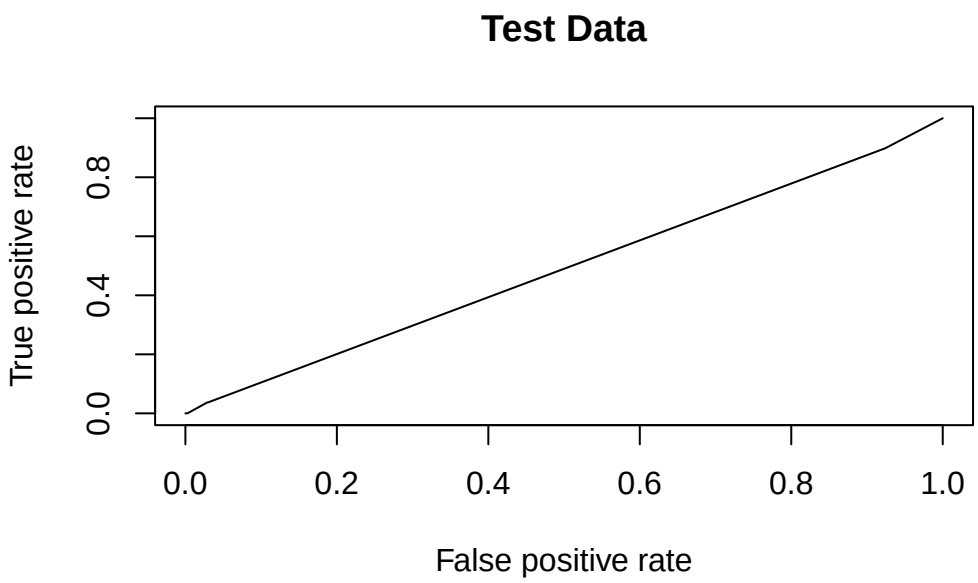
```
# gets optimal model
svmbest <- tune_out_two$best.model
```

The optimal tuning parameters are for linear kernel, gamma .5, and cost .1.

```
svmbest_factor <- svm(spam ~ inherit + exclaim_subj, data = train1, kernel = "radial",
    cost = .1, gamma = .5, scale = TRUE)
# new data frame with only the variables in the model
datasvc_factor <- d[train, c('spam', 'inherit', 'exclaim_subj')]

# plotting decision boundary with built in plot
plot(svmbest_factor, datasvc_factor)
```




```
# uses svmbest from prior question as model
fitteds <- attributes(
    predict(svmbest, train1,
            decision.values = TRUE)
  )$decision.values

# test data is used for predictions
fitted_tests <- attributes(
    predict(svmbest, test1, decision.values = T)
  )$decision.values

# plots ROC
rocplot(-fitted_tests, test1$spam, main = "Test Data")
```




```
# gets predictions so AUC can be calculated
pred_opts <- prediction(-fitted_tests, test1$spam)
auc_opts <- performance(pred_opts, "auc")

# prints AUC
auc_opts@y.values
```

```
[[1]]
[1] 0.4907491
```

The AUC is .49 for the model with the optimal parameters above.

Full Models

Q6

Repeat Q4, but this time use all variables in emails. Compare performance on the test set with the two-variable model

```
# creates glm and prints summary tab
glm_fit_full <- glm(spam ~ to_multiple + from + cc + sent_email + time +
                      image + attach + dollar + winner + inherit + viagra +
                      password + num_char + line_breaks + format + re_subj +
```

```
                      exclaim_subj + urgent_subj + exclaim_mess + number,
                    train1, family = binomial)
```

```
Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
# prints summary to see coefficients
summary(glm_fit_full)
```

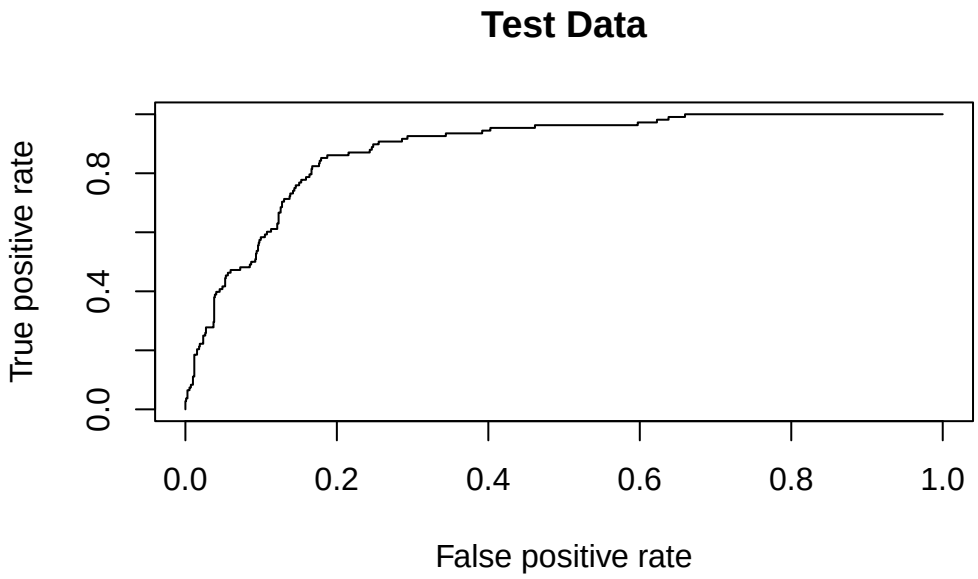
```
Call:
glm(formula = spam ~ to_multiple + from + cc + sent_email + time +
    image + attach + dollar + winner + inherit + viagra + password +
    num_char + line_breaks + format + re_subj + exclaim_subj +
    urgent_subj + exclaim_mess + number, family = binomial, data = train1)

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)  -4.457e+01  1.075e+04  -0.004 0.996693
to_multiple1 -2.703e+00  4.072e-01  -6.638 3.17e-11 ***
from1        -2.143e+01  1.075e+04  -0.002 0.998410
cc            1.868e-03  3.341e-02   0.056 0.955401
sent_email1  -1.716e+01  3.438e+02  -0.050 0.960204
time          4.957e-08  3.471e-08   1.428 0.153245
image        -2.182e+00  8.078e-01  -2.700 0.006925 **
attach        8.621e-01  1.728e-01   4.988 6.10e-07 ***
dollar       -7.238e-02  3.306e-02  -2.189 0.028574 *
winneryes     2.307e+00  4.199e-01   5.494 3.92e-08 ***
inherit       4.896e-01  2.617e-01   1.871 0.061398 .
viagra        2.705e+00  1.344e+03   0.002 0.998395
password     -7.091e-01  2.959e-01  -2.396 0.016565 *
num_char      6.189e-02  3.090e-02   2.003 0.045191 *
line_breaks  -6.968e-03  1.799e-03  -3.874 0.000107 ***
format1      -5.435e-01  1.777e-01  -3.059 0.002223 **
re_subj1     -1.452e+00  4.477e-01  -3.244 0.001178 **
exclaim_subj -3.235e-03  3.040e-01  -0.011 0.991509
urgent_subj1  3.758e+00  1.299e+00   2.892 0.003822 **
exclaim_mess  1.384e-02  2.383e-03   5.808 6.31e-09 ***
numbersmall  -1.122e+00  1.850e-01  -6.065 1.32e-09 ***
numberbig    -2.986e-01  2.594e-01  -1.151 0.249654
```

```
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 1708.2  on 2707  degrees of freedom
Residual deviance: 1125.2  on 2686  degrees of freedom
AIC: 1169.2

Number of Fisher Scoring iterations: 18
```

```r
# gets predictions
predicted_full <- predict(glm_fit_full, test1, type ='response')

# gets object to plot
# roc_full <- roc(test1$spam, predicted_full)

# plots the roc plot
# plot(roc_full, main = 'Test Data',
#     xlab = 'False positive rate',
#     ylab = 'True positive rate')

# prints the AUC
(auc_full <- auc(test1$spam,predicted_full))
```

```
Setting levels: control = 0, case = 1

Setting direction: controls < cases

Area under the curve: 0.8839
```

```r
# plots roc
rocplot(predicted_full, test1$spam, main = 'Test Data')
```



The AUC is .88 for the full model.

The coefficients here largely match what I would expect. I thought it was interesting that attachments were associated with spam, but it makes sense when thinking about how many common emails actually don't include them and how its a common way to get people to click on things However, words like urgent and winner all make sense as spam and I'm not surprised to see them. Overall, these coefficients were not that surprising.

The AUC here is much higher than the two variable model, indicating that this is a better model for predicting the data. We can see this because the 2 variable model has an AUC of .49, while the full model has an AUC of .88, which is closer to 1, indicating better prediction ability. We can also tell based on the curves of the ROC plots, where the full model arches toward the northeast/top left corner far more than the 2 variable model, which remains almost linear in appearance.

Q7

```
# data is already in the proper format with y as factor

# cross validation for tuning parameters
set.seed(247)
tune_out_full <- tune(svm, spam ~ to_multiple + from + cc + sent_email
                      + time + image + attach + dollar + winner + inherit
                      + password + num_char + line_breaks + format +
                        re_subj + exclaim_subj + urgent_subj + exclaim_mess
```

```
                        + number, data = train1,
                        ranges = list(gamma = c(.5, 1, 2, 3),
                                      cost = c(0.1,  1.0, 10.0),
                                      kernel = c('radial', 'linear')),
                        tunecontrol = tune.control(cross=5))

# prints optimal parameters
(tune_out_full$best.parameters)
```

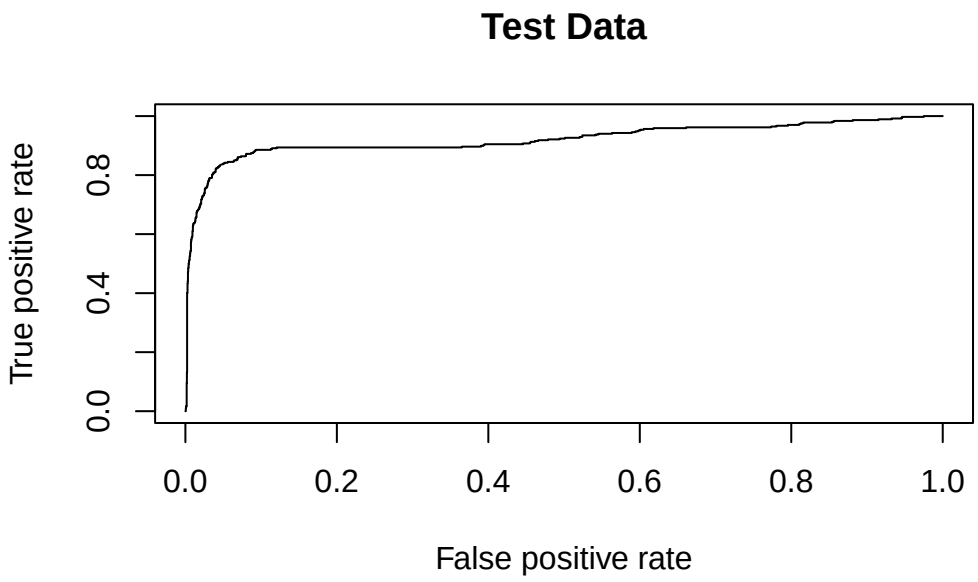
```
   gamma cost kernel
10     1   10 radial
```

```
# optimal model
opt_model <- tune_out_full$best.model

# uses optimal parameters to make svm
#svmfull <- svm(spam ~ to_multiple + from + cc + sent_email +
#time + image + attach + dollar + winner + inherit + password +
 # num_char + line_breaks + format + re_subj + exclaim_subj +
 # urgent_subj + exclaim_mess + number, data = train1, kernel =
 # "radial", cost = 10, gamma = .5)

svmfull <- opt_model # the two are the same! renames for consistence
```

The optimal tuning parameters are radial, cost 10, and gamma 1.

```
# plotting full svm with optimal parameters
rocplot <- function(pred, truth, ...) {
  predob <- prediction(pred, truth)
  perf <- performance(predob, "tpr", "fpr")
  plot(perf, ...)
}

# uses svmfull from prior question as model
fittedsf <- attributes(
    predict(opt_model, d[train, ],
            decision.values = TRUE)
  )$decision.values

# test data is used for predictions
```

```
fitted_testsf <- attributes(
    predict(opt_model, d[-train, ], decision.values = T)
  )$decision.values

# plots ROC
rocplot(-fitted_testsf, d[-train, "spam"], main = "Test Data")
```




```
# gets predictions so AUC can be calculated
pred_optsf <- prediction(-fitted_testsf, d[-train, "spam"])
auc_optsf <- performance(pred_optsf, "auc")

# prints AUC
auc_optsf@y.values
```

```
[[1]]
[1] 0.9203091
```

The AUC for the full model with the optimal parameters is .92.

Knowing that the almost full model with optimal parameters is .92 and the 2 variable model had an AUC of .49, we know that the full model is a better method of predicting if an email will be spam or not. This is because the AUC is much closer to one, and we can tell when looking at the plots by the shape of the curve that goes toward the top left in the full model vs remaining almost straight in the 2 variable model.

Q8

In the full logistic model, the AUC is .88, while for the almost-all-variable svm model the AUC is .92. This indicates that the svm model is better at predicting whether or not an email is spam because the value is closer to one. When comparing the plots from the previous questions, we can also see that the svm model hugs the northeast/top left corner more closely than the logistic model, indicating a better predictive ability.