

Machine Learning, Spring 2021

Homework 6

Daniil Bulat, Jonas Josef Huwyler, Haochen Li, Giovanni Magagnin

22/04/2021

Exercise 1

```
# Shuffle and Split Data (70-30)
set.seed(123)
Data = Data[sample(nrow(Data)),]
ntrain = floor(nrow(Data)*0.7)
Data.Train = Data[1:ntrain,]
Data.Test = Data[(ntrain+1):nrow(Data),]
```

Question 1

Run logistic regression on the training data with respect to `perimeter_mean` and `concaveP_mean`.

```
glm.fit = glm(diagnosis~perimeter_mean + concaveP_mean,data = Data.Train,family = binomial)
summary(glm.fit)
```

```
##
## Call:
## glm(formula = diagnosis ~ perimeter_mean + concaveP_mean, family = binomial,
##      data = Data.Train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -2.45821  -0.26302  -0.10627   0.06101   2.59877
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -13.28007    1.77748  -7.471 7.94e-14 ***
## perimeter_mean   0.09716    0.01927   5.042 4.62e-07 ***
## concaveP_mean   77.06571   11.42843   6.743 1.55e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
## Null deviance: 515.00 on 397 degrees of freedom
## Residual deviance: 149.32 on 395 degrees of freedom
## AIC: 155.32
##
## Number of Fisher Scoring iterations: 7
```

Question 2

Get a list of predictions. How do you interpret these as probabilities.

```
yhat = predict(glm.fit, type = "response")
contrasts(Data.Train$diagnosis)
```

```
## M
## B 0
## M 1
```

Question 3

Predict the actual class label (B or M).

```
yhat = ifelse(yhat<0.5,"B","M")
```

Question 4

Compute the number of misclassifications, as well as the absolute number of false positives and false negatives.

```
result = data.frame(id = Data.Train$id,diagnosis = Data.Train$diagnosis,prediction = yhat)
result$error = ifelse(result$diagnosis != result$prediction,1,0)

# Number of misclassified cases
misclas = sum(result$error)
cat("Number of misclassified cases:", misclas,"[",round((100/nrow(result))*misclas,2),"%]")
```

```
## Number of misclassified cases: 27 [ 6.78 %]
```

```
# False positives
x = ifelse(result$prediction == "M" & result$diagnosis == "B", 1, 0)
falPos = sum(x)
cat("False positives :", falPos,"[",round((100/nrow(result))*falPos,2),"%]")
```

```
## False positives : 10 [ 2.51 %]
```

```
# False negatives
x = ifelse(result$prediction == "B" & result$diagnosis == "M", 1, 0)
falNeg = sum(x)
cat("False negatives :", falNeg,"[",round((100/nrow(result))*falNeg,2),"%] ")
```

```
## False negatives : 17 [ 4.27 %]
```

Exercise 2

Question 1

Briefly summarize what we are predicting in dependence on which data in this example.

We are using data from ISLR called Smarket, which consist of information of percentage returns for the S&P 500 stock index over 1250 days from the beginning of 2001 until the end of 2005. What we are predicting is the direction of the market (percentage returns of the S&P 500 index) changes on the specified date.

Question 2

What is the outcome, with 2 or 6 features, respectively? How does our prediction compare to predicting that the market goes up all the time.

With 6 features, the logistic regression correctly predicted the movement of the market 48% of the time.

With 2 features, the logistic regression correctly predicted the movement of the market 56% of the time.

In 2005, the market goes up in 141 days of 252 trading days, around 56% of the time. If we predicting the market goes up all the time, we would predict correctly 56% of the time. The logistic regression predicts the increase of the market 31% of the time in 2005 when the model uses 6 features, and 72% of the time when the model uses 2 features, respectively. So the model with 6 features performs worse than predicting the market goes up all the time, whereas the model with 2 features performs better than the other two.

Question 3

Run your own logistic regression with only Lag1 as input variable.

```
# prepare data
library(ISLR)
attach(Smarket)

# split data into train (before 2004) and test (2005) data
train = (Year < 2005)
Smarket.2005 = Smarket[!train,]
Direction.2005 = Direction[!train]
```

```
# Logistic model with only Lag1
glm.fits = glm(Direction ~ Lag1, data = Smarket, family = binomial, subset = train)
summary(glm.fits)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1, family = binomial, data = Smarket,
##      subset = train)
##
## Deviance Residuals:
##      Min       1Q   Median       3Q      Max
## -1.308  -1.188   1.102   1.163   1.291
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.03215    0.06335   0.507   0.612
## Lag1        -0.05460    0.05165  -1.057   0.290
##
## (Dispersion parameter for binomial family taken to be 1)
##
##      Null deviance: 1383.3  on 997  degrees of freedom
## Residual deviance: 1382.1  on 996  degrees of freedom
## AIC: 1386.1
##
## Number of Fisher Scoring iterations: 3
```

```
# predict for the test data
glm.probs = predict(glm.fits, Smarket.2005, type = "response")
```

```
# classify the probabilities to binary label
glm.pred = rep("Down", 252)
glm.pred[glm.probs > .5] = "Up"
```

```
ups = ifelse(glm.pred == "Up", 1, 0)
downs = ifelse(glm.pred == "Down", 1, 0)
perc = 100/(sum(ups)+sum(downs))
```

```
# Show the result
table(glm.pred, Direction.2005)
```

```
##           Direction.2005
## glm.pred Down  Up
##      Down   20  25
##      Up    91 116
```

```
# compute the accuracy rate
mean(glm.pred == Direction.2005)
```

```
## [1] 0.5396825
```

```
# accuracy rate when predicting an increase  
116/(116+91)
```

```
## [1] 0.5603865
```

```
# accuracy rate when predicting a decrease  
20/(20+25)
```

```
## [1] 0.4444444
```