# STATS202 Homework3 solutions

**Sample solutions from a student with minor modification**

## Problem 1

This problem relates to the QDA model, in which the observations within each class are drawn from a normal distribution with a class-specific mean vector and a class specific covariance matrix. We consider the simple case where $p = 1$; i.e. there is only one feature. Suppose that we have K classes, and that if an observation belongs to the kth class then X comes from a one-dimensional normal distribution, $X \sim N(\mu_k, \sigma_k^2)$. Recall that the density function for the one-dimensional normal distribution is given in (4.11). Prove that in this case, the Bayes classifier is not linear. Argue that it is in fact quadratic.
**Answer**:
To begin with, we can denote $P(Y = k \mid x = x)$ as:
$P_k(x) = \frac{\pi_k(\frac{1}{\sqrt{2\pi}\sigma_k})\exp(-\frac{(x-\mu_k)^2}{2\sigma_k^2})}{\sum_{i=1}^n \pi_i(\frac{1}{\sqrt{2\pi}\sigma_i})\exp(-\frac{(x-\mu_i)^2}{2\sigma_i^2})}$
where we can denote constant $\sum_{i=1}^n \pi_i(\frac{1}{\sqrt{2\pi}\sigma_i})\exp(-\frac{(x-\mu_i)^2}{2\sigma_i^2}) = c$.
To achieve the max of the equation mentioned above, we can find the max of its log:
$\log(P_k(x)) = \log(\pi_k) + \log(\frac{1}{\sqrt{2\pi}\sigma_k}) - \frac{(x-\mu_k)^2}{2\sigma_k^2} - \log(c)$,
i.e., the discriminant function: $\delta(x) = \log(\pi_k) + \log(\frac{1}{\sqrt{2\pi}\sigma_k}) - \frac{(x-\mu_k)^2}{2\sigma_k^2}$, which is a quadratic function.
In conclusion, the Bayes classifier is not linear when $X \sim N(\mu_k, \sigma_k^2)$.

## Problem 2

(a) If the Bayes decision boundary is linear, do we expect LDA or QDA to perform better on the training set? On the test set?
**Answer**: QDA should perform better on training set while LDA should perform better on test set. 1) QDA gives more flexibility in fitting a model on training set. As a result, QDA may better capture the features of training set and achieve better performance. 2) Given that the true boundary is linear, QDA will face the overfit problem on test set while LDA will achieve better performance.

(b) If the Bayes decision boundary is non-linear, do we expect LDA or QDA to perform better on the training set? On the test set?
**Answer**: Given that the true boundary is non-linear, QDA should perform better on both the training set and the test set.

(c) In general, as the sample size n increases, do we expect the test prediction accuracy of QDA relative to LDA to improve, decline, or be unchanged? Why?
**Answer**: As the sample size n increases, the test prediction accuracy of QDA relative to LDA should improve. As mentioned before, QDA allows more flexibilty when training the model, which may lead to the overfit problem on test set. However, as the sample size n increases, variance of the model may decrease and the overfit problem can be offset.

(d) True or False: Even if the Bayes decision boundary for a given problem is linear, we will probably achieve a superior test error rate using QDA rather than LDA because QDA is flexible enough to model a linear decision boundary. Justify your answer.
**Answer**: False. QDA, which allows higher flexibility, tends to overfit the training set and achieves

high variance model. As a result, LDA usually yields better test error when the true model has a linear decision boundary.

## Problem 3

(a) Estimate the probability that a student who studies for 40 h and has an undergrad GPA of 3.5 gets an A in the class.
**Answer**: The model can be written as $P(x) = \frac{1}{1+\exp(-(-6+0.05x_1+x_2))}$.
A student who studies for 40 h and has an undergrad GPA of 3.5 has a probablity to get an A as: $P(x \mid 40, 3.5) = \frac{1}{1+\exp(-(-6+0.05*40+3.5))} = \frac{1}{1+\exp(0.5)} = 0.38$.

(b) How many hours would the student in part (a) need to study to have a 50% chance of getting an A in the class?
**Answer**: To have a 50% chance of getting an A in the class, we can derive an equation as: $P(x \mid x_1, 3.5) = \frac{1}{1+\exp(-(-6+0.05x_1+3.5))} = \frac{1}{1+\exp(2.5-0.05x_1)} = 0.5$, where we can get $x_1 = 50$.
As a result, the student should study 50 hours to have a 50% chance of getting an A in the class.

## Problem 4

(a) On average, what fraction of people with an odds of 0.37 of defaulting on their credit card payment will in fact default?
**Answer**: If odds = 0.37, we have $\frac{p(x)}{1-p(x)} = 0.37$, i.e., $p(x) = \frac{0.37}{1.37} = 27\%$. As a result, 27% people with an odds of 0.37 of defaulting on their credit card payment will in fact default.

(b) Suppose that an individual has a 16% chance of defaulting on her credit card payment. What are the odds that she will default?
**Answer**: If $p(x) = 0.16$, we have odds $= \frac{p(x)}{1-p(x)} = \frac{0.16}{0.84} = 0.19$. As a result, an individual has a 16% chance of defaulting on her credit card payment may default at an odds of 0.19.

## Problem 5

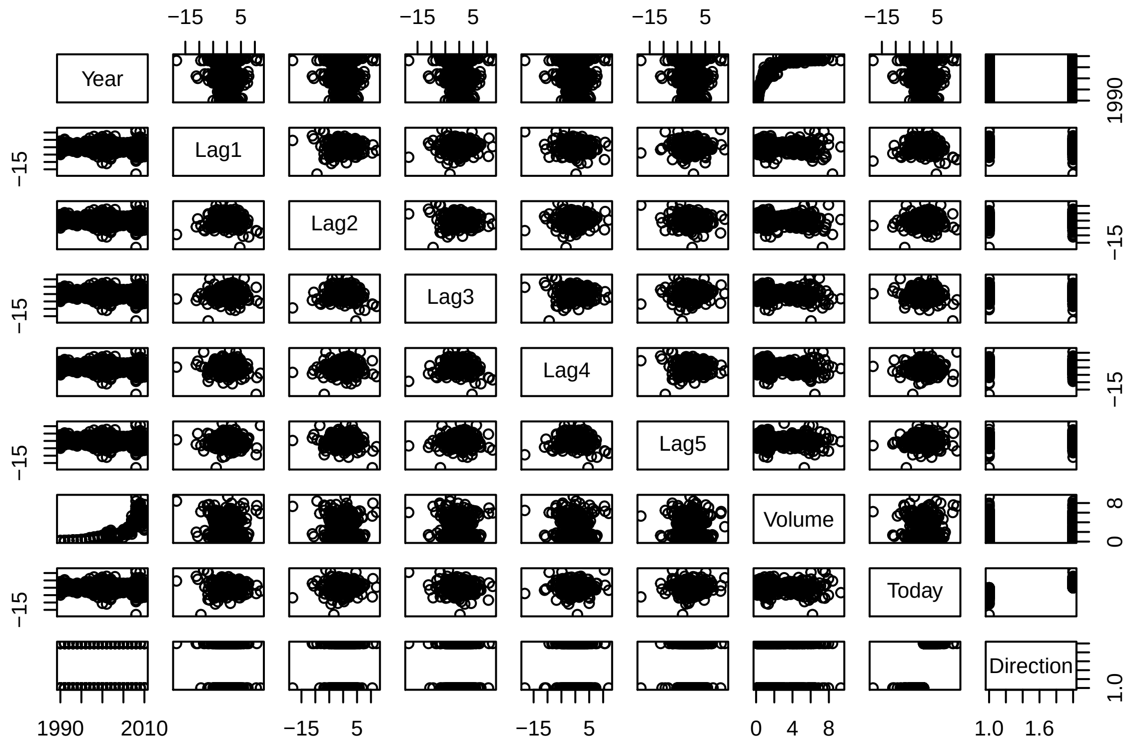
(a) Produce some numerical and graphical summaries of the Weekly data. Do there appear to be any patterns?
**Answer**: As shown in the following chart, we can see clear relationship between volume and year.

```
summary(Weekly)
```

```
##       Year           Lag1               Lag2               Lag3
##  Min.   :1990   Min.   :-18.1950   Min.   :-18.1950   Min.   :-18.1950
##  1st Qu.:1995   1st Qu.: -1.1540   1st Qu.: -1.1540   1st Qu.: -1.1580
##  Median :2000   Median :  0.2410   Median :  0.2410   Median :  0.2410
##  Mean   :2000   Mean   :  0.1506   Mean   :  0.1511   Mean   :  0.1472
##  3rd Qu.:2005   3rd Qu.:  1.4050   3rd Qu.:  1.4090   3rd Qu.:  1.4090
##  Max.   :2010   Max.   : 12.0260   Max.   : 12.0260   Max.   : 12.0260
##       Lag4               Lag5              Volume
##  Min.   :-18.1950   Min.   :-18.1950   Min.   :0.08747
##  1st Qu.: -1.1580   1st Qu.: -1.1660   1st Qu.:0.33202
##  Median :  0.2380   Median :  0.2340   Median :1.00268
##  Mean   :  0.1458   Mean   :  0.1399   Mean   :1.57462
##  3rd Qu.:  1.4090   3rd Qu.:  1.4050   3rd Qu.:2.05373
##  Max.   : 12.0260   Max.   : 12.0260   Max.   :9.32821
##      Today          Direction
```

```
##  Min.   :-18.1950   Down:484
##  1st Qu.: -1.1540   Up  :605
##  Median :  0.2410
##  Mean   :  0.1499
##  3rd Qu.:  1.4050
##  Max.   : 12.0260
```

```r
pairs(Weekly)
```



(b) Use the full data set to perform a logistic regression with Direction as the response and the five lag variables plus Volume as predictors. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?
**Answer**: The logistic model using Direction as the response and the five lag variables plus Volume as predictors is reported as:

```r
logreg = glm(Direction~Lag1+Lag2+Lag3+Lag4+Lag5+Volume,family=binomial,data=Weekly)
summary(logreg)
```

```
##
## Call:
## glm(formula = Direction ~ Lag1 + Lag2 + Lag3 + Lag4 + Lag5 +
##     Volume, family = binomial, data = Weekly)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6949  -1.2565   0.9913   1.0849   1.4579
##
## Coefficients:
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.26686    0.08593   3.106   0.0019 **
## Lag1        -0.04127    0.02641  -1.563   0.1181
## Lag2         0.05844    0.02686   2.175   0.0296 *
## Lag3        -0.01606    0.02666  -0.602   0.5469
## Lag4        -0.02779    0.02646  -1.050   0.2937
## Lag5        -0.01447    0.02638  -0.549   0.5833
## Volume      -0.02274    0.03690  -0.616   0.5377
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1496.2  on 1088  degrees of freedom
## Residual deviance: 1486.4  on 1082  degrees of freedom
## AIC: 1500.4
##
## Number of Fisher Scoring iterations: 4
```

As shown above, Lag 2 has significantly small p-value, which indicates that Lag 2 is statistically significant in the logistic regression.
(c) Compute the confusion matrix and overall fraction of correct predictions. Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.
**Answer**: Confusin matrix is shown below. According to the matrix, 56% of the predictions are correct. False positve rate of the model is 89% while false negative is 8%.

```
logreg_prob = predict (logreg ,Weekly, type = "response")
logreg_pred = rep ("Down ", nrow(Weekly))
logreg_pred [logreg_prob > 0.5]=" Up"
table(logreg_pred, Weekly$Direction)
```

```
##
## logreg_pred Down  Up
##         Up    430 557
##        Down    54  48
```

```
cat("correct predictions", (557+54)/nrow(Weekly),"; ")
```

```
## correct predictions 0.5610652 ;
```

```
cat("FP", 430/(54+430),"; ")
```

```
## FP 0.8884298 ;
```

```
cat("FN", 48/(48+557))
```

```
## FN 0.07933884
```

(d) Now fit the logistic regression model using a training data period from 1990 to 2008, with Lag2 as the only predictor. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2009 and 2010).
**Answer**: Run the model again with training set and only with Lag2. Confusin matrix on test set can be shown below. According to the matrix, 63% of the predictions are correct. False positve rate of the model is 80% while false negative is 8%.

```
train = subset(Weekly, Year<=2008)
test = subset(Weekly, Year>2008)
logreg_t = glm(Direction~Lag2,family=binomial,data=train)
```

```r
logreg_t_prob = predict (logreg_t, test, type = "response")
logreg_t_pred = rep ("Down ", nrow(test))
logreg_t_pred [logreg_t_prob > 0.5]=" Up"
table(logreg_t_pred, test$Direction)
```

```
##
## logreg_t_pred Down Up
##           Up    34 56
##          Down    9  5
```

```r
cat("correct predictions", (9+56)/nrow(test),"; ")
```

```
## correct predictions 0.625 ;
```

```r
cat("FP", 34/(9+34),"; ")
```

```
## FP 0.7906977 ;
```

```r
cat("FN", 5/(5+56))
```

```
## FN 0.08196721
```

(e) Repeat (d) using LDA.
**Answer**: Run the LDA with training set and only with Lag2. Confusin matrix on test set can be shown below. According to the matrix, 63% of the predictions are correct. False positve rate of the model is 80% while false negative is 8%.

```r
LDA_t = lda(Direction~Lag2,data=train)
LDA_t_pred = predict (LDA_t, test)
table(LDA_t_pred$class, test$Direction)
```

```
##
##        Down Up
##   Down    9  5
##   Up     34 56
```

```r
cat("correct predictions", (9+56)/nrow(test),"; ")
```

```
## correct predictions 0.625 ;
```

```r
cat("FP", 34/(9+34),"; ")
```

```
## FP 0.7906977 ;
```

```r
cat("FN", 5/(5+56))
```

```
## FN 0.08196721
```

(f) Repeat (d) using QDA.
**Answer**: Run the QDA with training set and only with Lag2. Confusin matrix on test set can be shown below. According to the matrix, 59% of the predictions are correct. False positve rate of the model is 100% while false negative is 0%. QDA sacrifices "predicting the Down direction accurately" for "predicting the Up direction accurately".

```r
QDA_t = qda(Direction~Lag2,data=train)
QDA_t_pred = predict (QDA_t, test)
table(QDA_t_pred$class, test$Direction)
```

```
##
##        Down Up
##   Down    0  0
```

```
##   Up     43 61
```

```r
cat("correct predictions", (0+61)/nrow(test),"; ")
```

```
## correct predictions 0.5865385 ;
```

```r
cat("FP", 43/(0+43),"; ")
```

```
## FP 1 ;
```

```r
cat("FN", 0/(0+61))
```

```
## FN 0
```

(g) Repeat (d) using KNN with K = 1.
**Answer**: Run the KNN with K = 1 on the training set, considering only Lag2. Confusin matrix on test set can be shown below. According to the matrix, 50% of the predictions are correct. False positve rate of the model is 51% while false negative is 49%.

```r
set.seed (1)
x_train = as.matrix(train$Lag2)
x_test = as.matrix(test$Lag2)
y_train = as.matrix(train$Direction)
y_test = as.matrix(test$Direction)
knn_t_pred=knn (x_train, x_test, y_train ,k=1)
table(knn_t_pred, test$Direction)
```

```
##
## knn_t_pred Down Up
##       Down   21 30
##       Up     22 31
```

```r
cat("correct predictions", (21+31)/nrow(test),"; ")
```

```
## correct predictions 0.5 ;
```

```r
cat("FP", 22/(21+22),"; ")
```

```
## FP 0.5116279 ;
```

```r
cat("FN", 30/(30+31))
```

```
## FN 0.4918033
```

(h) Which of these methods appears to provide the best results on this data?
**Answer**: If we only evaluate the results in terms of correct predictions, Logistic regression and LDA have the best performation (~63% correct predictions).

(i) Experiment with different combinations of predictors, including possible transformations and interactions, for each of the methods. Report the variables, method, and associated confusion matrix that appears to provide the best results on the held out data. Note that you should also experiment with values for K in the KNN classifier.
**Answer**:
Logistic regression: If we only evaluate the results in terms of correct predictions, model $Direction = Lag2 + Lag5 : Lag2$ achieves better outcome (~63.4% correct predictions).

```r
bestLog = glm(Direction ~ Lag2 + Lag5:Lag2, family = binomial, data = train)
bestLog_prob = predict (bestLog, test, type = "response")
bestLog_pred = rep ("Down ", nrow(test))
bestLog_pred [bestLog_prob > 0.5]=" Up"
table(bestLog_pred, test$Direction)
```

```
##
## bestLog_pred Down Up
##         Up     34 57
##         Down     9  4
```

```r
cat("correct predictions", (9+57)/nrow(test),"; ")
```

```
## correct predictions 0.6346154 ;
```

```r
cat("FP", 34/(9+34),"; ")
```

```
## FP 0.7906977 ;
```

```r
cat("FN", 4/(4+57))
```

```
## FN 0.06557377
```

LDA: If we only evaluate the results in terms of correct predictions, model $Direction = Lag2 + Lag5 : Lag2$ achieves better outcome (~63.4% correct predictions).

```r
bestLDA = lda(Direction ~ Lag2 + Lag5:Lag2, data = train)
bestLDA_pred = predict(bestLDA, test)
table(bestLDA_pred$class, test$Direction)
```

```
##
##         Down Up
##   Down     9  4
##   Up      34 57
```

```r
cat("correct predictions", (9+57)/nrow(test),"; ")
```

```
## correct predictions 0.6346154 ;
```

```r
cat("FP", 34/(9+34),"; ")
```

```
## FP 0.7906977 ;
```

```r
cat("FN", 4/(4+57))
```

```
## FN 0.06557377
```

QDA: If we only evaluate the results in terms of correct predictions, model $Direction = Lag2 + Lag4 : Lag2$ achieves better outcome (~60% correct predictions).

```r
bestQDA = qda(Direction ~ Lag2+Lag2:Lag4, data = train)
bestQDA_pred = predict(bestQDA, test)
table(bestQDA_pred$class, test$Direction)
```

```
##
##         Down Up
##   Down     7  6
##   Up      36 55
```

```r
cat("correct predictions", (7+55)/nrow(test),"; ")
```

```
## correct predictions 0.5961538 ;
```

```r
cat("FP", 7/(7+36),"; ")
```

```
## FP 0.1627907 ;
```

```r
cat("FN", 55/(6+55))
```

```
## FN 0.9016393
```

KNN: If we only evaluate the results in terms of correct predictions, knn model with $k = 20$ achieves better outcome (~59% correct predictions).

```
knn_pred=knn (x_train, x_test, y_train ,k=20)
table(knn_pred, test$Direction)
```

```
##
## knn_pred Down Up
##     Down   19 20
##     Up     24 41
```

```
cat("correct predictions", (20+41)/nrow(test),"; ")
```

```
## correct predictions 0.5865385 ;
```

```
cat("FP", 23/(23+20),"; ")
```

```
## FP 0.5348837 ;
```

```
cat("FN", 20/(20+41))
```

```
## FN 0.3278689
```

LDA and logistic regression still achieve better results in this context.

## Problem 6

Using the Boston data set, fit classification models in order to predict whether a given suburb has a crime rate above or below the median. Explore logistic regression, LDA, and KNN models using various subsets of the predictors. Describe your findings.
**Answer**:
To begin with, define a column *is.crime* where $is.crime = 1$ if a given suburb has a crime rate above the median. Also, spilt the original data set into training and test set.

```
attach(Boston)
is.crime = rep(0, length(crim))
is.crime[crim > 0.5]= 1
Full = data.frame(Boston, is.crime)
drops = c("crim")
Full = Full[ , !(names(Full) %in% drops)]
set.seed(1)
smp_size = floor(0.8*nrow(Full))
train_ind = sample(seq_len(nrow(Full)), size=smp_size)
train = Full[train_ind, ]
test = Full[-train_ind, ]
```

Then, run logistic regression on the training set.
Findings: Certain featurs are insiginificant in prediction. Abandon the features with high p-value (tax and chas) to get an improved model $is.crime = . - tax - chas$. Confusin matrix on test set can be shown below. According to the matrix, 86% of the predictions are correct. False positve rate of the model is 9% while false negative is 24%.

```
baseline = glm(is.crime ~ ., data = train, family = binomial)
```

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

```r
summary(baseline)
```

```
##
## Call:
## glm(formula = is.crime ~ ., family = binomial, data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.7994  -0.0390   0.0000   0.0001   3.2010
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -59.135429  13.415449  -4.408 1.04e-05 ***
## zn           -0.131564   0.073526  -1.789  0.07356 .
## indus        -0.431239   0.140513  -3.069  0.00215 **
## chas          0.969027   1.134603   0.854  0.39307
## nox         110.802628  22.697793   4.882 1.05e-06 ***
## rm           -2.764966   1.089782  -2.537  0.01118 *
## age           0.128713   0.032596   3.949 7.86e-05 ***
## dis           1.920852   0.476268   4.033 5.50e-05 ***
## rad           0.743166   0.257985   2.881  0.00397 **
## tax          -0.008349   0.009658  -0.864  0.38736
## ptratio       0.310972   0.176283   1.764  0.07772 .
## black        -0.018686   0.006303  -2.965  0.00303 **
## lstat        -0.225931   0.104936  -2.153  0.03132 *
## medv          0.257068   0.107664   2.388  0.01695 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 554.814  on 403  degrees of freedom
## Residual deviance:  96.383  on 390  degrees of freedom
## AIC: 124.38
##
## Number of Fisher Scoring iterations: 10
```

```r
logreg = glm(is.crime ~ . - tax-chas, data = train, family = binomial)
```

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

```r
logreg_prob = predict(logreg, test, type = "response")
logreg_pred = rep(0, length(logreg_prob))
logreg_pred[logreg_prob > 0.5] = 1
table(logreg_pred,test$is.crime)
```

```
##
## logreg_pred  0  1
##           0 63  8
##           1  6 25
```

```r
cat("correct predictions", (63+25)/nrow(test),"; ")
```

```
## correct predictions 0.8627451 ;
```

```r
cat("FP", 6/(6+63),"; ")
```

```
## FP 0.08695652 ;
```

```r
cat("FN", 8/(8+25))
```

```
## FN 0.2424242
```

Then, run LDA on the training set, with all features included.
Findings: I found that removing 2~3 covariates wouldn't have meaningful impact on the classification result, and decided to use the full model for analysis. Confusin matrix on test set can be shown below. According to the matrix, 90% of the predictions are correct. False positve rate of the model is 0% while false negative is 30%.

```r
baseline = lda(is.crime ~ ., data = train)
baseline_pred = predict(baseline, test)
table(baseline_pred$class, test$is.crime)
```

```
##
##      0  1
##   0 69 10
##   1  0 23
```

```r
cat("correct predictions", (69+23)/nrow(test),"; ")
```

```
## correct predictions 0.9019608 ;
```

```r
cat("FP", 0/69,"; ")
```

```
## FP 0 ;
```

```r
cat("FN", 10/(10+23))
```

```
## FN 0.3030303
```

Finally, run KNN on the training set, with K = 1,2,3,5,10,20, repectively.
Findings: I found that the model has best result at K = 2. Confusin matrix on test set can be shown below. According to the matrix, 94% of the predictions are correct. False positve rate of the model is 0% while false negative is 18%.

```r
set.seed (1)
drops = c("is.crime")
x_train = train[ , !(names(train) %in% drops)]
x_test = test[ , !(names(test) %in% drops)]
y_train = as.matrix(train$is.crime)
y_test = as.matrix(test$is.crime)
knn_pred=knn (x_train, x_test, y_train ,k=2)
table(knn_pred, test$is.crime)
```

```
##
## knn_pred  0  1
##        0 69  6
##        1  0 27
```

```r
cat("correct predictions", (69+27)/nrow(test),"; ")
```

```
## correct predictions 0.9411765 ;
```

```r
cat("FP", 0/69,"; ")
```

```
## FP 0 ;
```

```r
cat("FN", 6/(6+27))
```

```
## FN 0.1818182
```