Classification

Kelly Trinh

*The data set used in this assignment can be accessed **here (https://archive.ics.uci.edu/ml/datasets/bank+marketing#)***

What is logistic regression?

There are multiple ways to create linear models through classification rather than regression. These classification linear models essentially find a decision boundary between classes. One way we do classification is through logistic regression. In logistic regression, the target variable is most commonly a binary output, allowing us to classify into one class or the other class. We then calculate the log odds of the positive class, and try to directly classify the data. Another way we can create a linear model through classification is using the Naive Bayes model, which uses Bayesian probability to classify data into one class or another.

This analysis uses a bank marketing data set and goes through both the logistic regression model and the Naive Bayes model to predict whether or not a certain client of the bank subscribed a term deposit using the classes balance, which is the amount of money in a client's checking account, and loan, which is whether or not a client has a personal loan.

Importing the data

First, we import the data from the csv file downloaded from the UCI Machine Learning Repository. I named the data set **df** for easier reference.

```
df <- read.csv('/Users/kellytrinh/Desktop/school/Regression/bank-full.csv', na.strings
="NA", header=TRUE)
```

Data cleaning

Next, we clean the data. I first checked if there were any NA values within the columns, in which it showed that there are none, so removing the NAs is not necessary. Next, since only the balance, loan, and y columns are of importance to me, I assigned the data frame to the subset of those columns. In this assignment, y represents whether or not a client subscribed a term deposit. Furthermore, the data frame had "yes" and "no" values in the loan and y categories, so I changed those to factors, such that 1 represents yes and 0 represents no.

```
sapply(df, function(x) sum(is.na(x)==TRUE))
```

```
##       age       job   marital education   default   balance   housing      loan
##         0         0         0         0         0         0         0         0
##   contact       day     month  duration  campaign     pdays  previous  poutcome
##         0         0         0         0         0         0         0         0
##         y
##         0
```

```
df <- df[,c(6, 8, 17)]

df$loan <- as.factor(ifelse (df$loan=='yes',1,0))

df$y <- as.factor(ifelse (df$y=="yes",1,0))
```

A. Dividing the data set into 80/20 train and test sets

The below code block portrays how I divided the data set into an 80/20 train/test set. The rows are randomly sampled to vector *i* with row indices. These are used to divide the data set.

```
set.seed(1234)
i <- sample(1:nrow(df), 0.80*nrow(df), replace=FALSE)
train <- df[i,]
test <- df[-i,]
```

B. Data exploration using the train set

After completing the train and test section, we can explore the data using different R commands. The data exploration of this data set is shown below.

```
dim(train) # Show the dimensions of the data frame
```

```
## [1] 36168     3
```

```
names(train) # Names of the columns in the data
```

```
## [1] "balance" "loan"    "y"
```

```
str(train) # Show information about the structure of the data frame
```

```
## 'data.frame':    36168 obs. of  3 variables:
##  $ balance: int  3857 631 1780 8016 749 1794 -59 246 578 1940 ...
##  $ loan   : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 1 2 ...
##  $ y      : Factor w/ 2 levels "0","1": 1 2 2 1 1 1 1 1 1 1 ...
```

```
summary(train) # Show the statistics of each numeric column in the data set
```

```
##     balance       loan      y
##  Min.   : -6847   0:30377   0:31918
##  1st Qu.:    74   1: 5791   1: 4250
##  Median :   451
##  Mean   :  1358
##  3rd Qu.:  1428
##  Max.   :102127
```

```
head(train) # Show the first six instances of each column in the data set
```

	balance <int>	loan <fct>	y <fct>
40784	3857	0	0
40854	631	0	1
41964	1780	0	1
15241	8016	0	0
33702	749	0	0
35716	1794	1	0

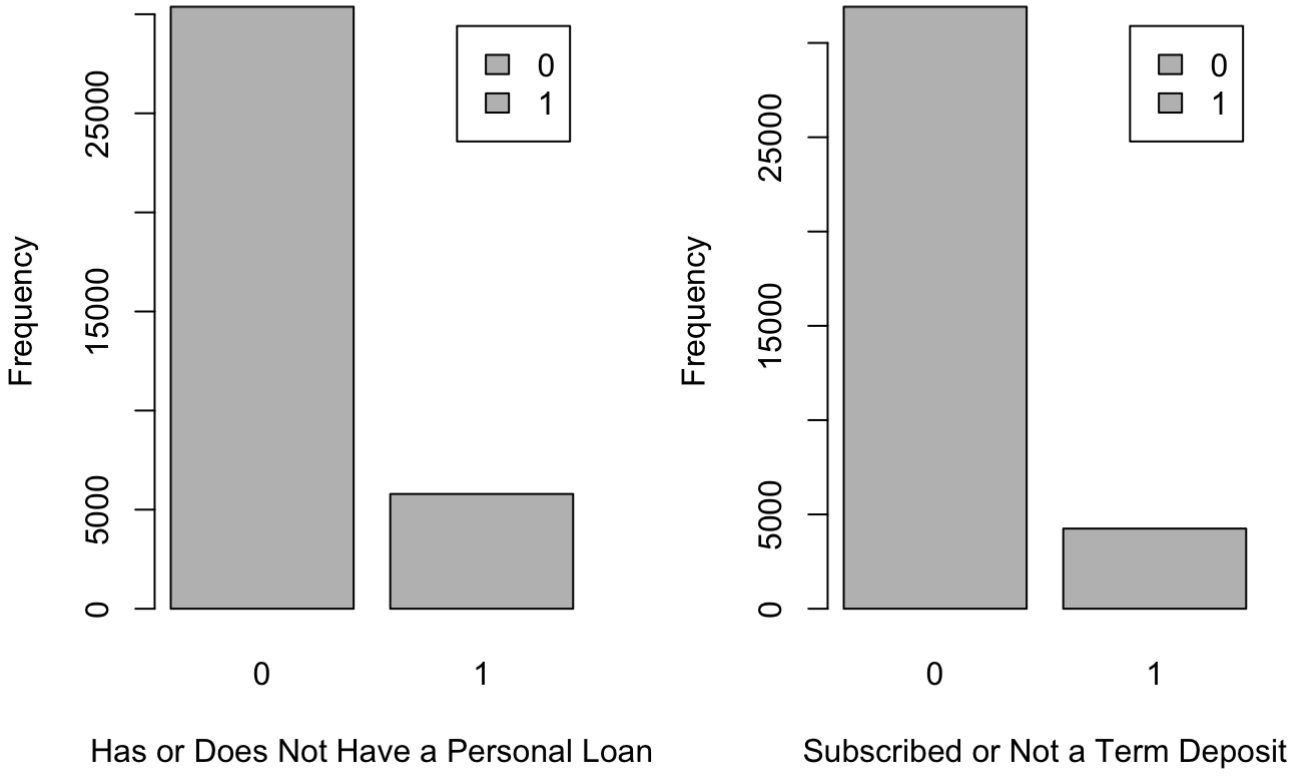
6 rows

C. Plotting the data based on the data exploration, using the training data

To plot the training data, I created two barplots using loan and y, which is whether or not the client has a personal loan and whether the client subscribed a term deposit. Using the two bar plots, we can see that there may be an association between the two variables, such that the loan may be able to predict y. I was going to create a box plot for the data as well, but there was no set of data in which a box plot would be able to nicely represent the data.

Note: 1 represents yes, and 0 represents no in the bar plots.

```
par(mfrow=c(1,2))
loan_graph <- table(train$loan)
term_graph <- table(train$y)
barplot(loan_graph, legend.text = TRUE, ylab="Frequency", xlab="Has or Does Not Have a P
ersonal Loan")
barplot(term_graph, legend.text = TRUE, ylab="Frequency", xlab="Subscribed or Not a Term
Deposit")
```



D. Building a logistic regression model

Now, a logistic regression model is built. I used both the loan and balance columns against the y column to create this logistic regression. The summary is also output.

```
glm1 <- glm(y~., data=train, family="binomial")
summary(glm1)
```

```
##
## Call:
## glm(formula = y ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.9361  -0.5150  -0.5082  -0.3845   2.3716
##
## Coefficients:
##               Estimate Std. Error  z value Pr(>|z|)
## (Intercept) -1.987e+00  1.882e-02 -105.593   <2e-16 ***
## balance      3.618e-05  4.327e-06    8.361   <2e-16 ***
## loan1       -6.523e-01  5.492e-02  -11.877   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 26180  on 36167  degrees of freedom
## Residual deviance: 25936  on 36165  degrees of freedom
## AIC: 25942
##
## Number of Fisher Scoring iterations: 5
```

The summary of the logistic regression model is similar to the summary of a linear regression model. We can see that for each one-unit change in balance, the log odds of whether or not a client subscribed a loan deposit increases by 0.000036. By comparing the null deviance and residual deviance, we can see that the model may be a good fit because the difference between the two is quite large. We can also see that the standard errors are quite low, which gives us good confidence in the estimates given in the summary. The balance standard error is lower, which gives us a higher confidence for balance than loan. The p-values for both balance and loan are quite small, which gives us good confidence that these two are good estimates for predicting y.

E. Building a Naive Bayes model

```
library(e1071)
nb1 <- naiveBayes(y~., data=train)
nb1
```

```
##
## Naive Bayes Classifier for Discrete Predictors
##
## Call:
## naiveBayes.default(x = X, y = Y, laplace = laplace)
##
## A-priori probabilities:
## Y
##         0         1
## 0.8824928 0.1175072
##
## Conditional probabilities:
##    balance
## Y       [,1]     [,2]
##   0 1299.318 2916.282
##   1 1798.174 3623.894
##
##    loan
## Y            0          1
##   0 0.83100445 0.16899555
##   1 0.90658824 0.09341176
```

The prior for whether a client has subscribed a loan deposit is 0.882 for if they haven't and 0.117 for if they have.

The probabilities that the client subscribed a loan deposit are 0.91% if they have a personal loan, and 0.09% if they do not have a personal loan. This tells us that clients are more likely to have subscribed a loan deposit if they have a personal loan because the difference between the two probabilities is extremely great.

The balance attribute is not categorical, but we can see that the mean balance for clients that have not subscribed a loan deposit is 1798, and the mean for if they have is 3624. These values are fairly different from each other, so we can guess that clients with a higher balance in their checking account are more likely to have subscribed a loan deposit.

F. Predict and Evaluate on the test data using the models

```
# Logistic Regression
prob1 <- predict(glm1, newdata=test, type="response")
pred1 <- ifelse(prob1>0.5, 1, 0)
acc1 <- mean(pred1==test$y)
print(paste("accuracy of logistic regression: ", acc1))
```

```
## [1] "accuracy of logistic regression:  0.884662169633971"
```

```
table(pred1, test$y)
```

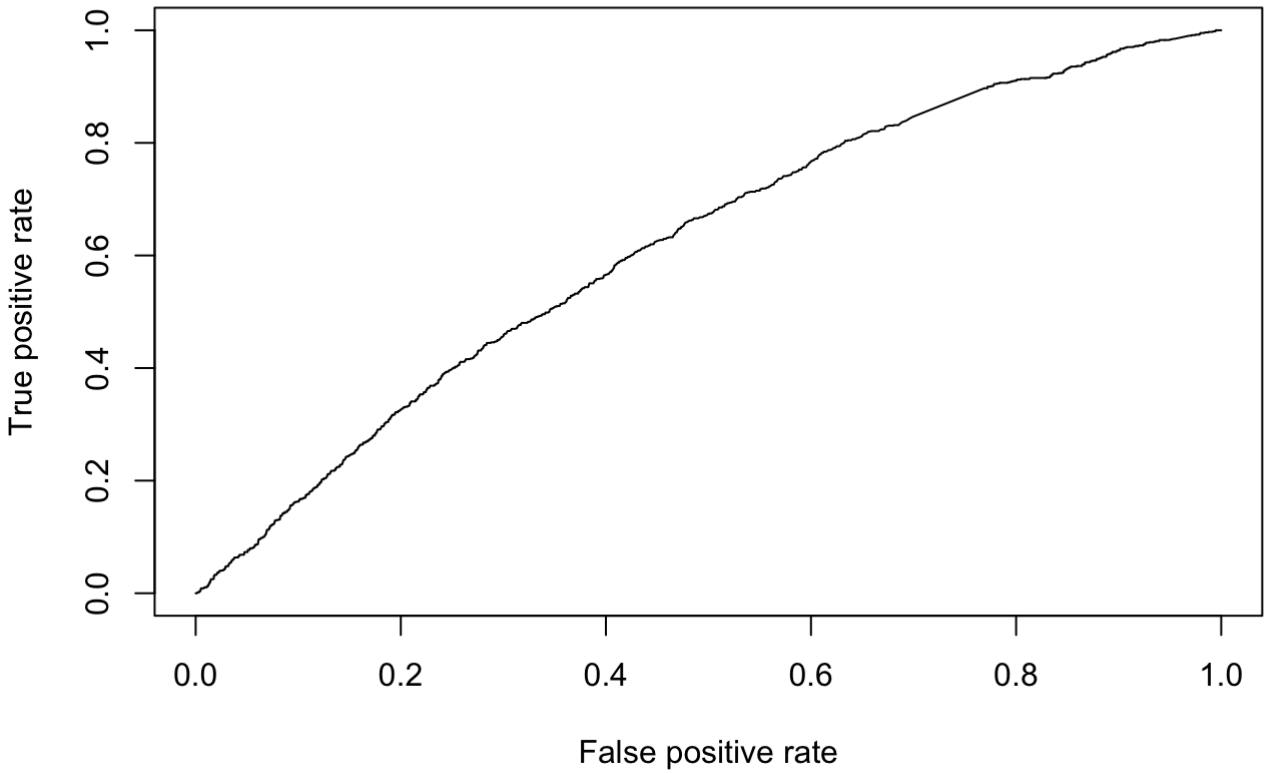
```
##
## pred1    0    1
##     0 8000 1039
##     1    4    0
```

```
# ROC for Logistic Regression
library(ROCR)
pr1 <- prediction(prob1, test$y)
prf1 <- performance(pr1, measure = "tpr", x.measure = "fpr")
plot(prf1)

# Sensitivity and Specificity for Log. Reg.
library(caret)
```

```
## Loading required package: ggplot2
```

```
## Loading required package: lattice
```




```
sensitivity(table(pred1, test$y))
```

```
## [1] 0.9995002
```

```
specificity(table(pred1, test$y))
```

```
## [1] 0
```

```
# Naive Bayes
prob2 <- predict(nb1, newdata=test, type="class")
table(prob2, test$y)
```

```
##
## prob2    0    1
##     0 7882 1018
##     1  122   21
```

```
acc2 <- mean(prob2==test$y)
print(paste("accuracy of the first Naive Bayes model: ", acc2))
```

```
## [1] "accuracy of the first Naive Bayes model:  0.873935640827159"
```

```
# Sensitivity and Specificity for Naive Bayes
sensitivity(table(prob2, test$y))
```

```
## [1] 0.9847576
```

```
specificity(table(prob2, test$y))
```

```
## [1] 0.02021174
```

We can see that the accuracy for the logistic regression model is slightly higher than that for the Naive Bayes model. Logistic regression models may perform better on larger data sets, and since this data set has approximately 45,000 observations, this may be way the accuracy for logistic regression is higher.

Furthermore, I build an ROC curve for the Logistic Regression model. Through the curve, we can see that the classifier does not directly shoot up, but is more of a diagonal line through the graph. This indicates that the logistic regression model may have not had a predictive value.

Lastly, I calculated the sensitivity and specificity for both the logistic regression and Naive Bayes model. For both models, we can see that the sensitivity is quite high, but the specificity is quite low. This indicates that the models are good at predicting clients that have subscribed a loan deposit, but the models are quite bad at predicting clients that haev not. One potential reason for this is the fact that only two attributes were used. If more attributes were added to create a multiclass model, then the algorithm could've been better trained to predict negative values.

G. Strengths and weaknesses of Naive Bayes and Logistic Regression

One weakness that both logistic regression and the Naive Bayes model has is that they are both high-bias models, meaning that prediction values may be extremely different in comparison to the training data. Furthermore, another weakness of the Naive Bayes model is that it is not the most efficient algorithm for large

data sets, but it can work for smaller data sets. On the other hand, a strength of the logisic regression model is that it is a good algorithm for large data sets, but not as good for smaller data sets. A strength of both logistic regression and Naive Bayes is that multiple classes can be implemented into the algorithms.

H. Benefits and drawbacks of the classification metrics used

The three classification metrics that I used in this assignment were accuracy, an ROC curve for the logistic regression model, and sensitivity & specificity. The benefits of using accuracy is that it is simple and easy to understand in regards to understanding the data. However, a drawback to using accuracy is that it may not say much about the precision of the data, and it may be bad on data that is quite unbalanced. For sensitivity & specificity, a significant drawback is that it is extremely difficult to have high rates of both sensitivity and specificity. The closer both are to 1, then the better the models are, but getting both to be high is difficult because they can be opposites of each other. However, a benefit of using sensitivity & specificity is we are able to tell what the model is better at predicting: positives or negatives. This means that we can create more models based off of what we now know from the sensitivity and specificity values. Lastly, an ROC curve is beneficial because it allows us to visualize the discrimination between the abnormal and normal test results. he drawbacks of using an ROC curve is that it does not display the exact cut-off value, and the number of samples is not displayed either.