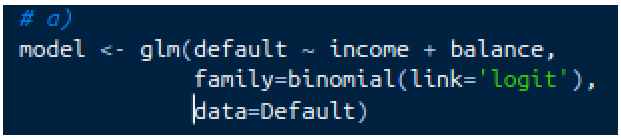
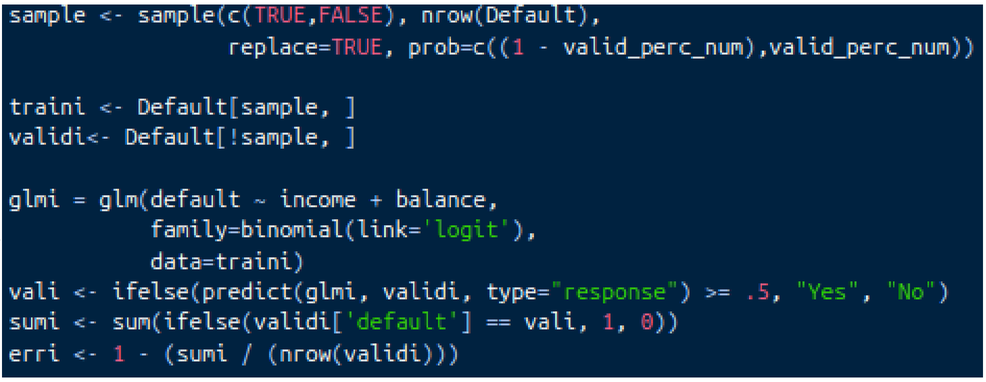
Question 5:

a) I fit a standard GLM with family = binomial(link=”logit”) in order to create a glm with a logit probability of the output being assigned to 1, that is the record defaulted on their loan



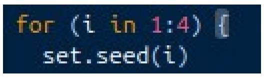
```
# a)
model <- glm(default ~ income + balance,
             family=binomial(link='logit'),
             data=Default)
```

b) I first set a variable valid_perc_num for the percentage of the data I wanted to allocate to validation data. I set .3 or 30% of the data for this test to be held back for validation data. My code splits into train and validate data, performs a model fitting and interprets the output, giving me a percentage of data that was classified incorrectly according to the validate data

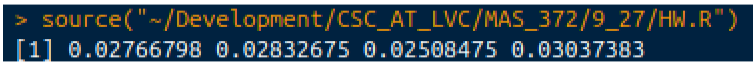


```
sample <- sample(c(TRUE,FALSE), nrow(Default),
                 replace=TRUE, prob=c((1 - valid_perc_num),valid_perc_num))

traini <- Default[sample, ]
validi<- Default[!sample, ]

glmi = glm(default ~ income + balance,
           family=binomial(link='logit'),
           data=traini)
vali <- ifelse(predict(glmi, validi, type="response") >= .5, "Yes", "No")
sumi <- sum(ifelse(validi['default'] == vali, 1, 0))
erri <- 1 - (sumi / (nrow(validi)))
```

c) I wrote a for loop to do this 4 different times, setting the seed differently each time. This resulted in an array of 4 different seeded values. Overall, the approximate test error (from the validation data) was roughly around 2.8%, which is really good for a simple glm!

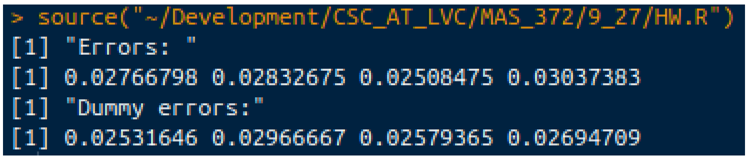


```
for (i in 1:4) {
  set.seed(i)
```



```
> source("~/Development/CSC_AT_LVC/MAS_372/9_27/HW.R")
[1] 0.02766798 0.02832675 0.02508475 0.03037383
```

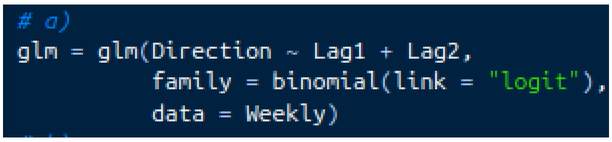
d) After repeating b and c using a dummy variable for student (all the values set to 0), the results of running this dummy model were not all that different. It improved the error slightly in the majority of cases, but visually not by any amount that I would consider noteworthy



```
> source("~/Development/CSC_AT_LVC/MAS_372/9_27/HW.R")
[1] "Errors: "
[1] 0.02766798 0.02832675 0.02508475 0.03037383
[1] "Dummy errors:"
[1] 0.02531646 0.02966667 0.02579365 0.02694709
```

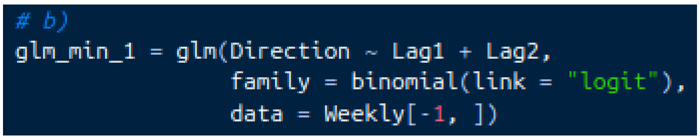
Question 7

a) I fit a simple glm logistic regression model



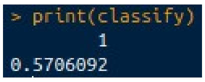
```
# a)
glm = glm(Direction ~ Lag1 + Lag2,
          family = binomial(link = "logit"),
          data = Weekly)
```

b) Ditto, but with leaving out the first row

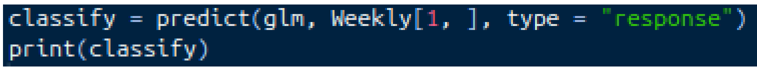


```
# b)
glm_min_1 = glm(Direction ~ Lag1 + Lag2,
                family = binomial(link = "logit"),
                data = Weekly[-1, ])
```

c) Just using the first row, the result is classified as up, with p = .5706



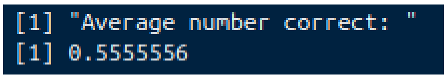
```
> print(classify)
        1
0.5706092
```



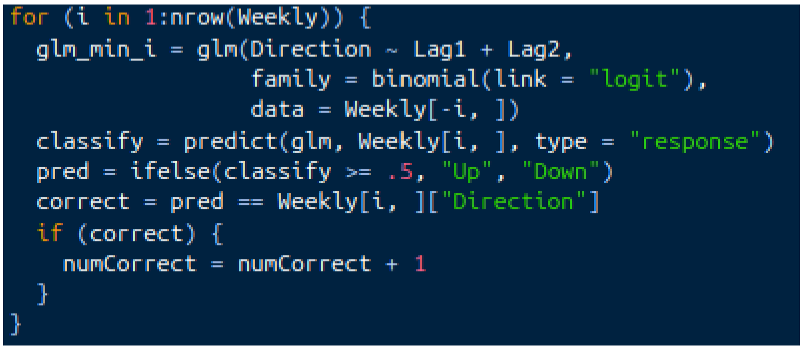
```
classify = predict(glm, Weekly[1, ], type = "response")
print(classify)
```

d) I wrote a basic for-loop to make a model and then classify each point according to the model without that point. I then kept a running tally of correct responses

e) Overall, the average of correct was not great, but it is better than just guessing (slightly...)



```
[1] "Average number correct: "
[1] 0.5555556
```



```
for (i in 1:nrow(Weekly)) {
  glm_min_i = glm(Direction ~ Lag1 + Lag2,
                  family = binomial(link = "logit"),
                  data = Weekly[-i, ])
  classify = predict(glm, Weekly[i, ], type = "response")
  pred = ifelse(classify >= .5, "Up", "Down")
  correct = pred == Weekly[i, ]["Direction"]
  if (correct) {
    numCorrect = numCorrect + 1
  }
}
```