PHP 2514 Homework #3

Blain Morin

March 19, 2018

Question 1:

Assume X is in units of $10,000.

First, find y-intercept (β_0):

$$\begin{aligned} \pi &= .27 \\ x &= 0 \\ logit(.27) &= \ln(\frac{.27}{1-.27}) = \beta_0 + \beta_1 X \\ -.9946 &= \beta_0 \end{aligned}$$

Next, find the slope (β_1):

$$\begin{aligned} \pi &= .88 \\ x &= 6 \\ logit(.88) &= \ln(\frac{.88}{1-.88}) = -.9946 + \beta_1 * 6 \\ .4978 &= \beta_1 \end{aligned}$$

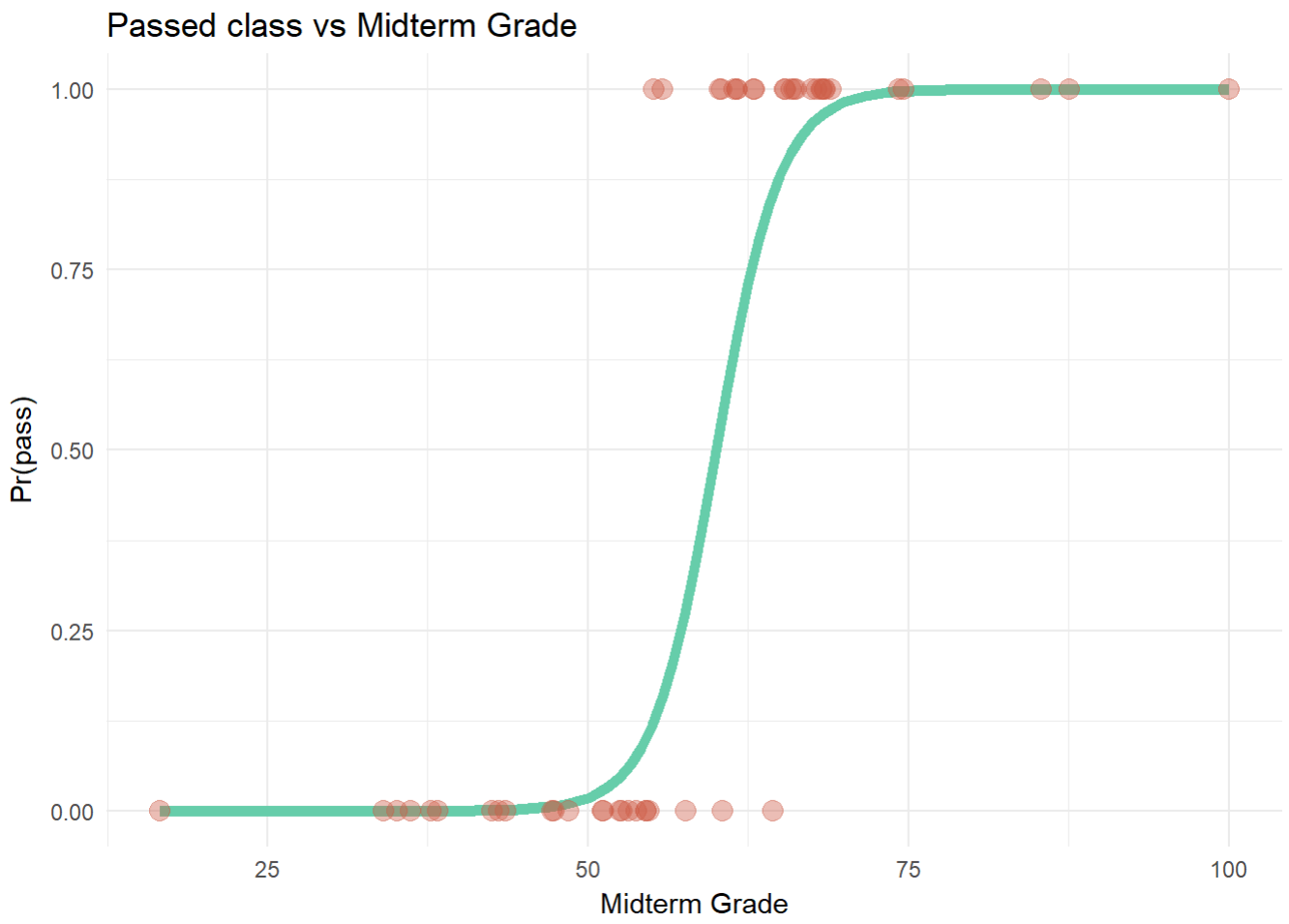
Thus, the equation is:

$$\pi = logit^{-1}(-.9946 + .4978X)$$

Question 2

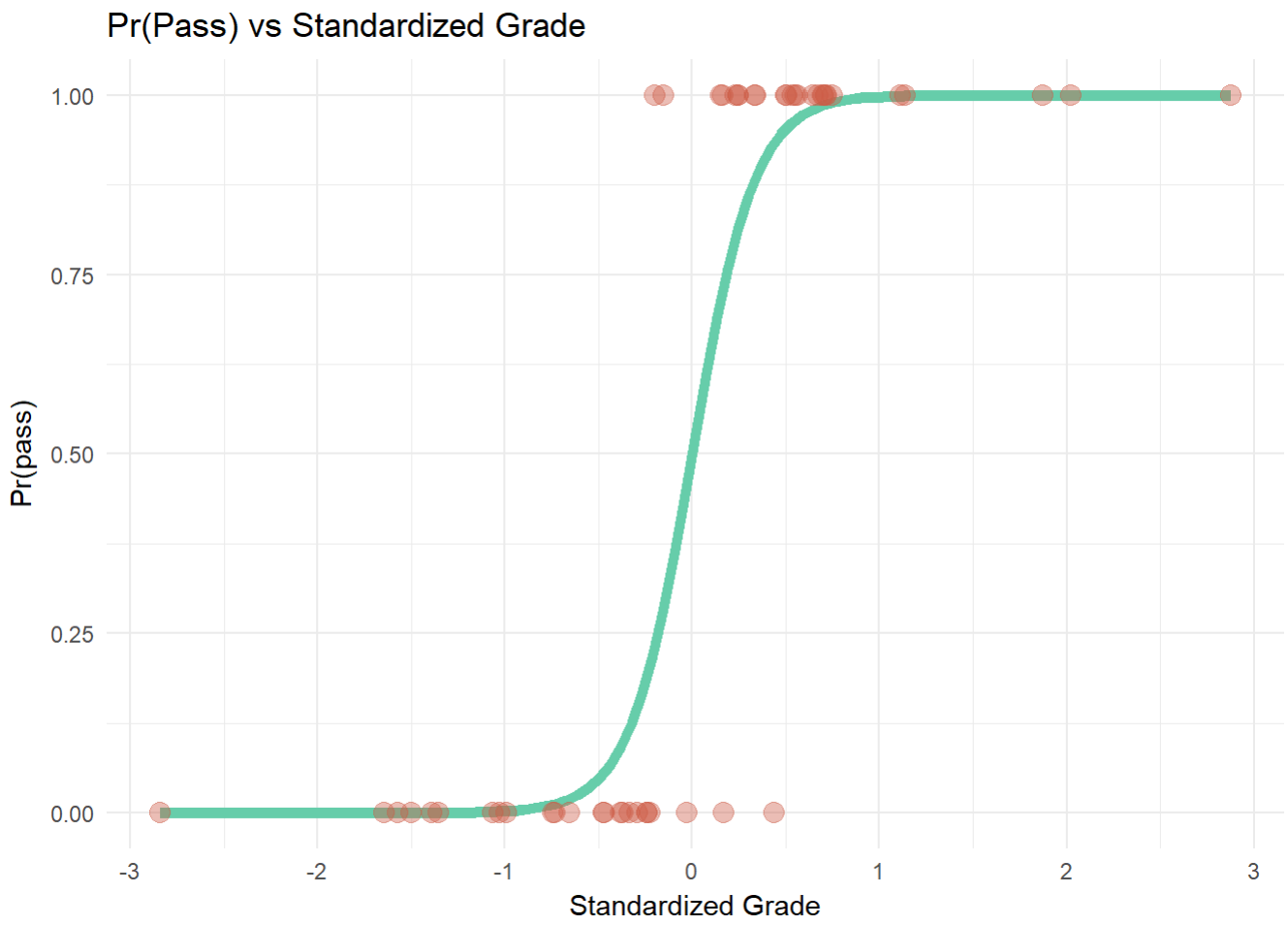
a.)

```
library(tidyverse)
library(arm)

###create data set
studentid = 1:50
set.seed(50)
grades = rnorm(50, 60, 15)

###use the our logit to predict if a student passed using their midterm grade
prob.pass = c()
for (i in 1:50) {

  prob = invlogit(-24 + .4 * grades[i])
  prob.pass[i] = rbinom(1, 1, prob = prob)

}

classdata = data.frame(studentid, grades, prob.pass)



###graph our points with the logit function given by the problem
classdata %>%
  ggplot(aes(x = grades, y = prob.pass)) +
    stat_function(fun = function(x) invlogit(-24 + .4*x), color = "aquamarine3", size = 2
) +
    geom_point(size = 3.5, alpha = .4, color = "coral3") +
    theme_minimal() +
    xlab("Midterm Grade") +
    ylab("Pr(pass)") +
    ggtitle("Passed class vs Midterm Grade")
```



b.)

```
###Add a column with standardized grades
classdata = classdata %>%
  mutate(stand.grade = (grades - mean(grades)) / sd(grades))


###Plot our standardized data with our updated logit curve
###To get our new logit function we multiplied the B0 by 0
###and multiplied B1 by 15
classdata %>%
    ggplot(aes(x = stand.grade, y = prob.pass)) +
    stat_function(fun = function(x) invlogit((-24*0) + (.4*15)*x), color = "aquamarine3",
 size = 2) +
    geom_point(size = 3.5, alpha = .4, color = "coral3") +
    theme_minimal() +
    xlab("Standardized Grade") +
    ylab("Pr(pass)") +
    ggtitle("Pr(Pass) vs Standardized Grade")
```



c.)

```
library(sjPlot)
attach(classdata)

###Add random noise column to our data
noise = rnorm(50, 0 , 1)
classdata$noise = noise

###Run model with no noise
model.no.noise = glm(prob.pass ~ stand.grade, family = binomial(link = "logit"))


###Run model with noise
model.noise = glm(prob.pass ~ stand.grade + noise, family = binomial(link = "logit"))


print(c(model.no.noise$deviance, model.noise$deviance))
```

```
## [1] 20.93671 20.92646
```

```
detach(classdata)
```

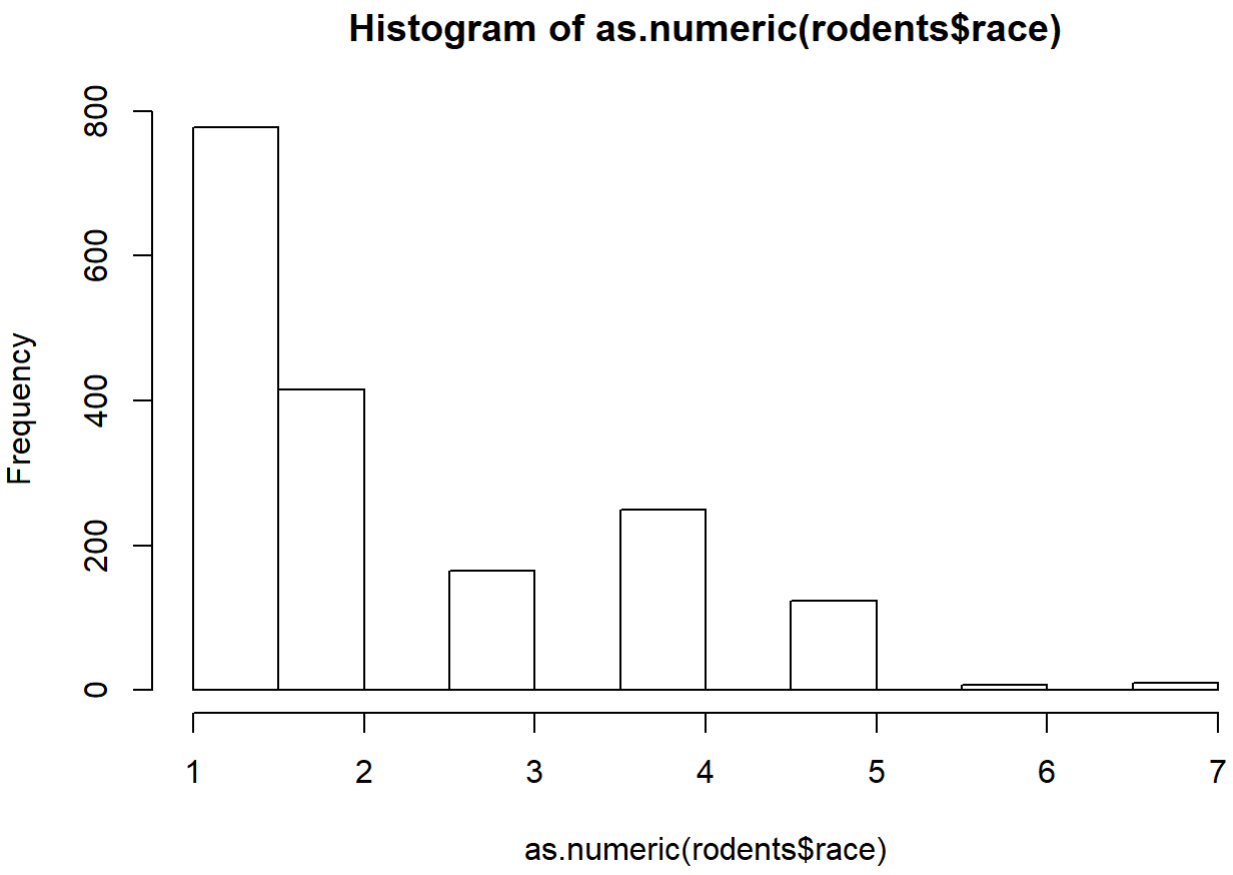
The deviance in our noise model is very similar to the deviance in the first model (20.93 vs 20.74. This makes sense theoretically: since the noise is randomly generated, it shouldn't impact the fit of our model (except for some random spurious correlation). Thus, our residuals will not change much. Therefore, the deviance will not change much.

Question 3

a.)

First we examined the histogram for our race categories.

```
rodents = read.table("rodents.txt")
rodents$rodent2 = as.factor(rodents$rodent2)
rodents$race = as.factor(rodents$race)


hist(as.numeric(rodents$race))
```



We see that race indicators 6 and 7 have a low frequency, so we will bin those together:

```
###Combine race 6 and 7 into category 6.

race.edit = c()

for (i in 1:nrow(rodents)) {
  if (rodents$race[i] == 7) {
    race.edit[i] = 6
  } else {
    race.edit[i] = rodents$race[i]
  }
}

rodents$race.edit = as.factor(race.edit)

base.model = glm(formula = rodent2 ~ race.edit,
                 family = binomial(link = "logit"),
                 data = rodents)
```

	Dependent variable:
	rodent2
race.edit2	1.536***
	(0.169)
race.edit3	1.449***
	(0.214)
race.edit4	1.867***
	(0.187)
race.edit5	0.400
	(0.292)
race.edit6	1.364**
	(0.554)
Constant	-2.152***
	(0.128)
Observations	1,522
Log Likelihood	-760.124
Akaike Inf. Crit.	1,532.248

Note: *p<0.1; **p<0.05;** p<0.01

In the output above, race 1 is our reference category. Raising e to the beta is the multiplicitive change in the odds of having rodents in the apartment. For example, for race category 2:

```
exp(1.536)
```

```
## [1] 4.645969
```

This is interpretted as:

Compared to race category 1, the odds of having rodents in the apartment is 4.6 times higher for those people in race category 2.

b.)

Next, we try to find a good model using other other house-hold level variables. The variables that I'm choosing to examine are:

- Rotted / Broken Windows
- Missing or Worn Flooring
- Cracks in Walls
- Holes in the flooring
- If the building was built before 1947
- If there is a regular exterminator
- Total household income
- Housing type (rent, public housing, owned)

```
attach(rodents)

b.data = data.frame(
  as.factor(rodent2),
  as.factor(extwin4_2),
  as.factor(extflr5_2),
  as.factor(intcrack2),
  as.factor(inthole2),
  as.factor(old),
  as.factor(race.edit),
  as.factor(regext),
  totincom2
)

detach(rodents)

attach(b.data)

###change income units to thousand
b.data$totincom2 = b.data$totincom2/10000

###remove NAs
b.data = na.omit(b.data)

b.model = glm(as.factor.rodent2. ~ ., family = binomial(link="logit"), data = b.data)

b.step.model = step(b.model, direction = "both")
```

as.factor.rodent2.

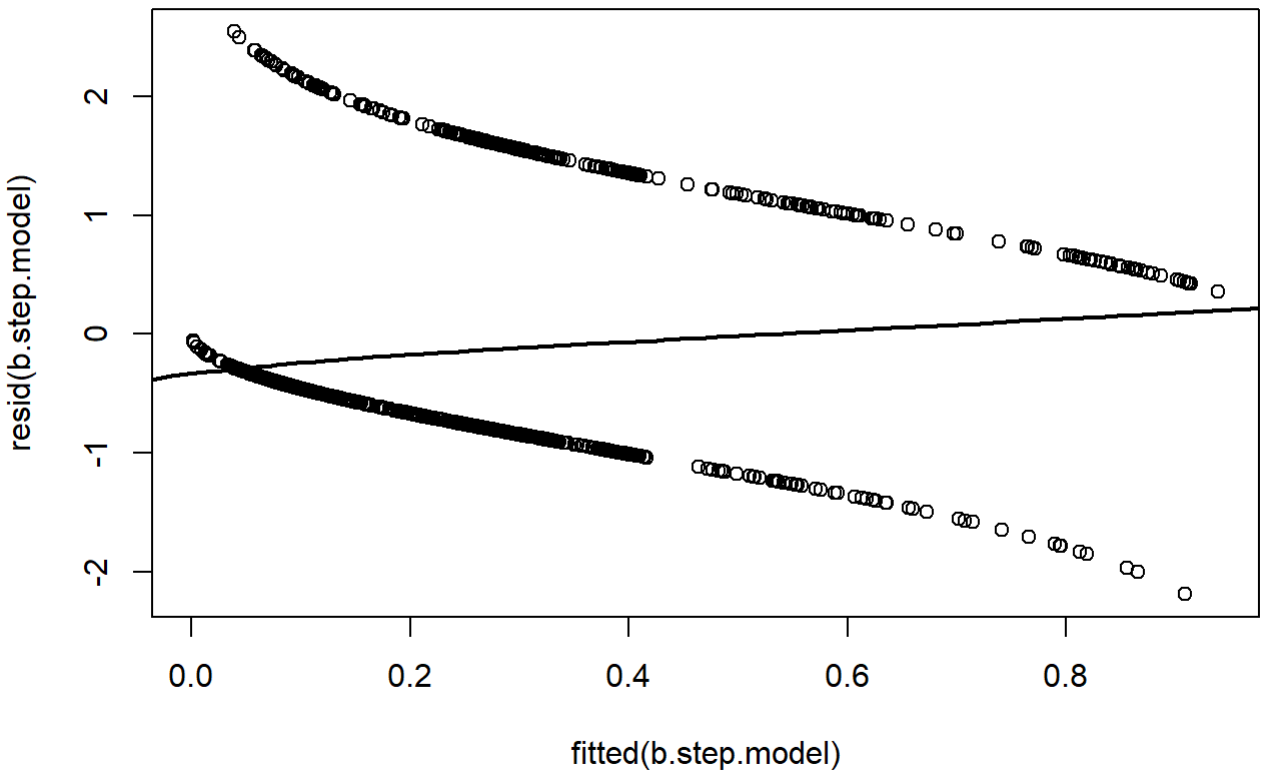
	Odds Ratio	*CI*	*p*
(Intercept)	0.11	0.07 – 0.16	<.001
as.factor.extflr5_2. (1)	2.19	1.10 – 4.41	.026
as.factor.intcrack2. (1)	3.50	2.31 – 5.31	<.001
as.factor.inthole2. (1)	2.88	1.65 – 5.11	<.001
as.factor.old. (1)	1.31	0.95 – 1.81	.097
as.factor.race.edit.			
2	3.44	2.33 – 5.12	<.001
3	3.15	1.92 – 5.17	<.001
4	5.13	3.32 – 8.00	<.001
5	1.53	0.77 – 2.89	.201
6	4.26	1.25 – 12.76	.012
totincom2	0.95	0.92 – 0.98	.006
Observations		1224	

The odds ratios for all of the variables are given in the above table. The interpretation of these is similar to part(a). For example, having a hole in the floor of the apartment means that there is 2.88 times the odds of having rodents than in an apartment with no holes in the floor. The interpretation is similar for income: a household earning 1000 dollars more than another similar household has .95 times the odds of having rodents.

Overall, most of the variables in our model are significant at the 5 percent level. The signs of the household factors are consistent with what one would expect (ie holes in the floor is associated with higher odds of having rats). The variables that the model dropped were having a regular exterminator and having a crack in the exterior windows.

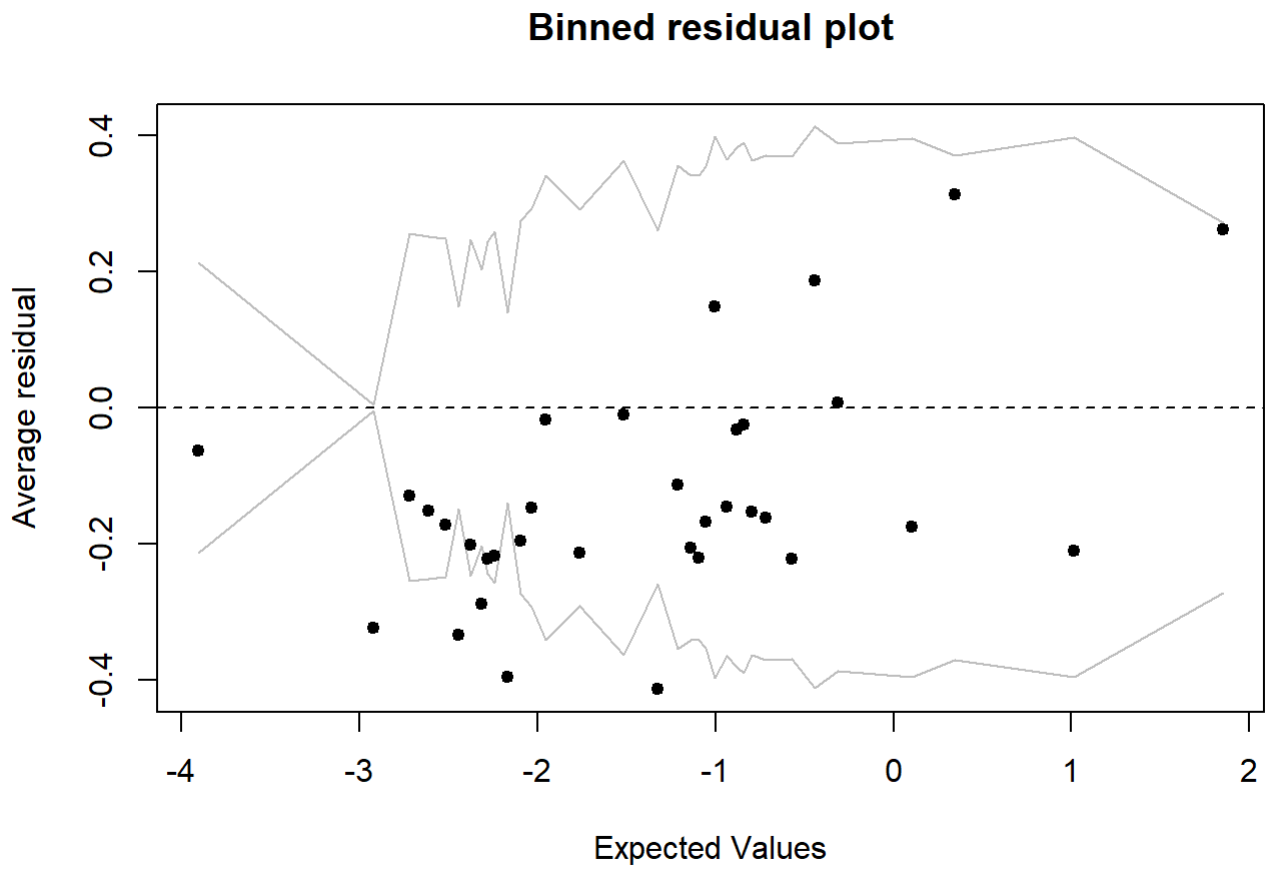
Next we checked the fit using residual diagnostics. First, we compared the residuals and fitted values:

```
plot(fitted(b.step.model), resid(b.step.model))
lines(lowess(predict(b.step.model),residuals(b.step.model)),col="black",lwd=2)
```



Since this is a logistic regression, we see two groups of points. The black line is a regression on the points. We see that this line is fairly close to the 0 line, which means that there is no pattern in our residuals. We can examine the residuals further using a bin plot:

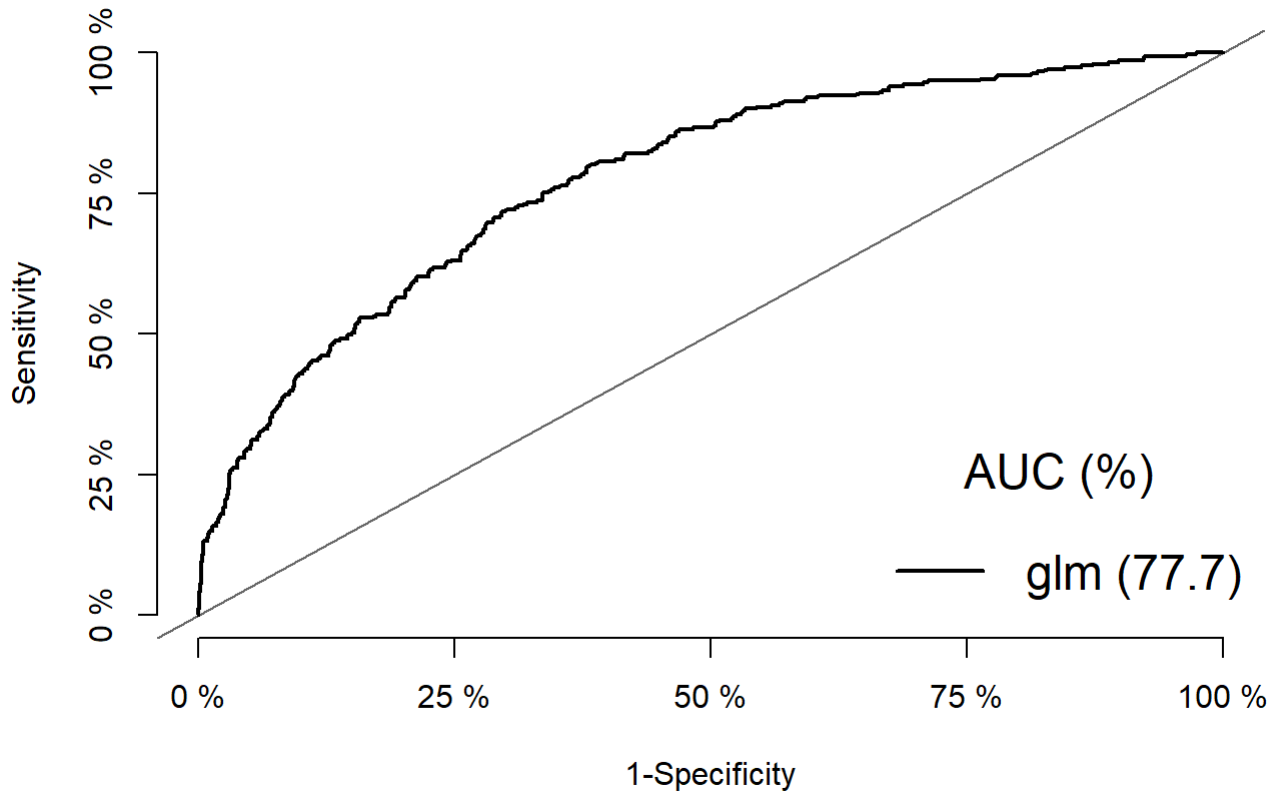
```
library(arm)
binnedplot(predict.glm(b.step.model), resid(b.step.model))
```



The grey lines represent the 95% confidence interval for the residuals assuming that the model was true. We see that the model is mostly okay, but that there are a few point in the lower expected values that fall outside of the confidence interval. This could be a problem with the model. We would maybe need to add a quadratic term or use a log transformation.

We also looked at the ROC curve:

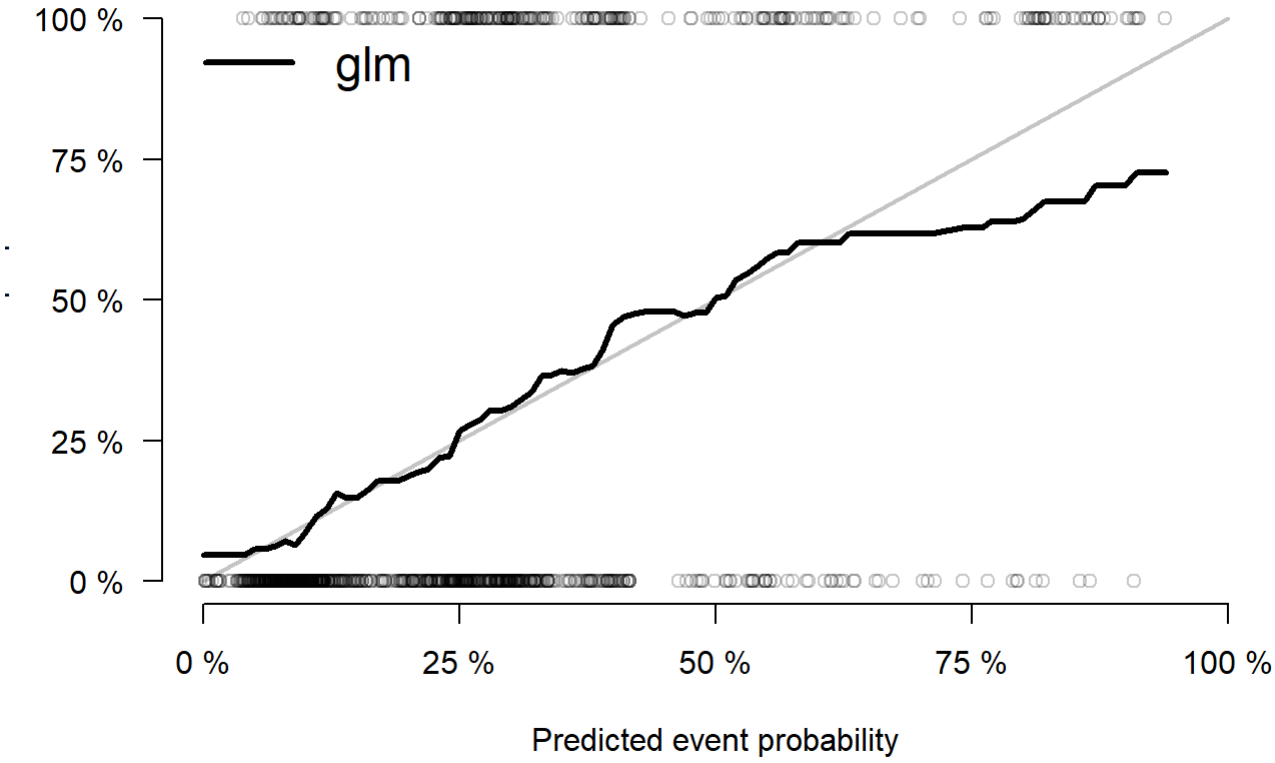
```
library(ModelGood)
plot(Roc(b.step.model), auc = TRUE)
```



As seen above, the area under our ROC curve is .77. This means that our model does a decent job of discriminating between apartments with and without rodents.

Next we looked at a calibration curve for our model:

```
calPlot2(b.step.model)
```



We can see from the calibration plot that our model predicts the observed probability closely up to probabilities of .6. Above .6, our model underestimates.

Question 4

For our first model, we use only the clinical variables. Since our outcome (dehydration) is an ordinal factor, we use a cumulative logit model.

```
library(readr)
library(nnet)
library(ordinal)
library(sjPlot)

###Select clinical only columns
dhaka = read_csv("dhaka.csv")
clinicaldata = dhaka[, 1:13]

###Change columns to factors
clinicaldata = lapply(clinicaldata, as.factor)
clinicaldata = as.data.frame(clinicaldata)

###Remove NA
clinicaldata = na.omit(clinicaldata)

###Run regression for only clinical predictors
model.clinical = clm(ordered(dehyd) ~ capref +
                       extrem +
                       eyes +
                       genapp +
                       mucous +
                       pulse +
                       resp +
                       skin +
                       tears +
                       thirst +
                       urine +
                       heart,
                     data = clinicaldata)

summary(model.clinical)
```

```
## formula:
## ordered(dehyd) ~ capref + extrem + eyes + genapp + mucous + pulse + resp + skin + tear
s + thirst + urine + heart
## data:    clinicaldata
##
##  link  threshold nobs logLik  AIC    niter max.grad cond.H
##  logit flexible  380  -301.86 651.73 17(0) 3.54e-09 1.9e+11
##
## Coefficients:
##          Estimate Std. Error z value Pr(>|z|)
## capref1  -0.29819         NA      NA       NA
## extrem1   0.36116         NA      NA       NA
## eyes1     0.25018         NA      NA       NA
## eyes2     1.06348         NA      NA       NA
## genapp1   0.49429         NA      NA       NA
## genapp2   1.12650         NA      NA       NA
## mucous1  -0.01163         NA      NA       NA
## mucous2 -20.78348         NA      NA       NA
## pulse1    0.53693         NA      NA       NA
## pulse2    0.47221         NA      NA       NA
## resp1     0.04421         NA      NA       NA
## resp2     0.27152         NA      NA       NA
## skin1     0.84665         NA      NA       NA
## skin2     0.79642         NA      NA       NA
## tears1    0.24017         NA      NA       NA
## tears2    0.77238         NA      NA       NA
## thirst1  -0.52307         NA      NA       NA
## thirst2  -0.36865         NA      NA       NA
## urine1    0.34005         NA      NA       NA
## urine2    0.01425         NA      NA       NA
## heart1    0.33680         NA      NA       NA
## heart2   21.02869         NA      NA       NA
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 0|1   0.7313         NA      NA
## 1|2   3.8836         NA      NA
```

We can see from the above output that there is something peculiar going on with our standard errors and p values (they are all NA). Examining the data further sheds light onto whats happening:

```
summary(clinicaldata)
```

```
##  dehyd   capref  extrem  eyes    genapp  heart   mucous  pulse   resp
##  0:156   0:379   0:359   0: 56   0:199   0:251   0:136   0:265   0:264
##  1:174   1:  1   1: 21   1:276   1: 83   1:126   1:242   1: 78   1:109
##  2: 50                   2: 48   2: 98   2:  3   2:  2   2: 37   2:  7
##  skin    tears   thirst  urine
##  0:195   0:185   0:  5   0:121
##  1:162   1:155   1:344   1:202
##  2: 23   2: 40   2: 31   2: 57
```

We can see from the data summary above the the variables capref, heart, and mucous have small responses. For example, there is only one person with capref = 1. This causes our standard errors to approach infinity and causes R to return NA. We can fix this by removing the offending variables:

```
###Run regression omitting low response variables
model.clinical2 = clm(ordered(dehyd) ~
                        extrem +
                        eyes +
                        genapp +
                        pulse +
                        resp +
                        skin +
                        tears +
                        thirst +
                        urine,
                      data = clinicaldata)

summary(model.clinical2)
```

```
## formula:
## ordered(dehyd) ~ extrem + eyes + genapp + pulse + resp + skin + tears + thirst + urine
## data:    clinicaldata
##
##  link  threshold nobs logLik  AIC    niter max.grad cond.H
##  logit flexible  380  -304.23 646.46 6(0)  2.97e-12 8.7e+02
##
## Coefficients:
##         Estimate Std. Error z value Pr(>|z|)
## extrem1  0.42926    0.49432   0.868 0.385180
## eyes1    0.30908    0.35064   0.881 0.378069
## eyes2    1.08978    0.51131   2.131 0.033061 *
## genapp1  0.56383    0.28210   1.999 0.045642 *
## genapp2  1.23614    0.33505   3.689 0.000225 ***
## pulse1   0.48374    0.30987   1.561 0.118496
## pulse2   0.47070    0.44799   1.051 0.293404
## resp1    0.12998    0.29739   0.437 0.662060
## resp2    0.34674    0.84438   0.411 0.681333
## skin1    0.82618    0.27294   3.027 0.002470 **
## skin2    0.73455    0.57116   1.286 0.198419
## tears1   0.29532    0.24600   1.200 0.229951
## tears2   0.84218    0.44865   1.877 0.060498 .
## thirst1 -0.56537    0.85895  -0.658 0.510399
## thirst2 -0.39690    0.94256  -0.421 0.673691
## urine1   0.37931    0.25361   1.496 0.134746
## urine2   0.02899    0.37217   0.078 0.937913
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 0|1   0.7313     0.9060   0.807
## 1|2   3.8561     0.9326   4.135
```

We see that the NA problem is fixed. We see that eyes, general appearance, skin and tears are significant at the 5% level. Except for thirst, the other variables have the expected sign. Thus, the only variable we will exclude from our next model is thirst.

Next we will add the individual level variables to the model:

```
library(dplyr)

###Merge factor data with nonfactor variables
clinicaldata = dhaka[, 1:13]
clinicaldata = lapply(clinicaldata, as.factor)
clinicaldata = as.data.frame(clinicaldata)
nonclinical = dhaka[,14:20]
nonclinical$female = as.factor(nonclinical$female)
dhaka.clean = cbind(clinicaldata, nonclinical)

###Remove heart, mucous, capref columns (due to nonresponse)

dhaka.clean = dhaka.clean %>%
  dplyr::select(-heart, -capref, -mucous, -thirst)

###Remove NAs
dhaka.clean = na.omit(dhaka.clean)

###Specify cumulative formula
allmodel = clm(ordered(dehyd) ~ .,
                      data = dhaka.clean)

###Use step algorithm to find best model
best.dhaka = step(allmodel, direction = "both")
```

Here is the summary of the model with the best AIC score:

```
summary(best.dhaka)
```

```
## formula:
## ordered(dehyd) ~ genapp + skin + tears + agemonths + episodes + muac
## data:    dhaka.clean
##
##  link  threshold nobs logLik  AIC    niter max.grad cond.H
##  logit flexible  383  -303.37 628.74 6(0)  2.91e-11 4.8e+06
##
## Coefficients:
##            Estimate Std. Error z value Pr(>|z|)
## genapp1    0.799496   0.274890   2.908 0.003633 **
## genapp2    1.379095   0.318757   4.326 1.52e-05 ***
## skin1      0.907155   0.249038   3.643 0.000270 ***
## skin2      1.366925   0.501011   2.728 0.006366 **
## tears1     0.358366   0.232320   1.543 0.122939
## tears2     1.152279   0.405174   2.844 0.004456 **
## agemonths  0.031462   0.008480   3.710 0.000207 ***
## episodes   0.035282   0.013081   2.697 0.006992 **
## muac      -0.019770   0.008927  -2.215 0.026786 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Threshold coefficients:
##     Estimate Std. Error z value
## 0|1  -0.6716     1.1713  -0.573
## 1|2   2.4474     1.1844   2.066
```

Our final model uses general apearance, skin pinch, tears, age, episodes, and mid upper arm circumfrence. The sign on all the coefficients are consistent with what we would expect. The reference catagory is dehyd = 0.

The betas for the factor variables can be interpretted in terms of odds ratios as follows:

When a child has a restless appearance (genapp1 = 1), the odds of some dehydration (dehyd = 1) is 2.2 times higher (exp(.7995)) than that of a normal appearance child.

Moreover, when a child has a restless appearance (genapp = 1), the odds of severe dehydration (dehyd = 2) is 3.97 times higher (exp(1.3791)) than that of a normal appearance child.

The betas for the nominal variables can be interpretted as follows:

An additional month of age increase the odds of being dehydrated by a factor of 1.03 (exp(.0314)).