STAD29 / STA 1007 assignment 2

Due Tuesday Jan 21 at 11:59pm on Quercus

Hand in the indicated questions. In preparation for the questions you hand in, it is worth your while to work through (or at least read through) the other questions as well.

Hand in your work on Quercus. If you did STAC32 last fall, it's the same procedure. A reminder is here: `https://www.utsc.utoronto.ca/~butler/c32/quercus1.nb.html`

You are reminded that work handed in with your name on it must be *entirely your own work*. It is as if you have signed your name under it. If it was done wholly or partly by someone else, *you have committed an academic offence*, and you can expect to be asked to explain yourself. The same applies if you allow someone else to copy your work. The grader will be watching out for assignments that look suspiciously similar to each other (or to my solutions). Besides which, if you do not do your own assignments, you *will* do badly on the exams, because the struggle to figure things out for yourself is an important part of the learning process.

You will probably need:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.0 --
## v ggplot2 3.2.1     v purrr   0.3.3
## v tibble  2.1.3     v dplyr   0.8.3
## v tidyr   1.0.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.4.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

Hand in question 2 below.

1. Work through, or at least read, chapter 19 of PASIAS.

2. Twenty students all studied for a certain exam. Some of them spend more time studying, and some of them spend less. Did the amount of time spent studying affect whether a student passed the exam? The data are in `http://ritsokiguess.site/STAD29/study_time.txt`.

   (a) (2 marks) Read in and display (at least some of) the data.

   Solution:

   ```
   my_url <- "http://ritsokiguess.site/STAD29/study_time.txt"
   study <- read_delim(my_url, " ")
   ```

   ```
   ## Parsed with column specification:
   ## cols(
   ##   hours = col_double(),
   ##   exam = col_character()
   ## )
   ```

   ```
   study
   ```

   ```
   ## # A tibble: 20 x 2
   ##    hours exam
   ```

```
##    <dbl> <chr>
##  1  0.5  fail
##  2  0.75 fail
##  3  1    fail
##  4  1.25 fail
##  5  1.5  fail
##  6  1.75 fail
##  7  1.75 pass
##  8  2    fail
##  9  2.25 pass
## 10  2.5  fail
## 11  2.75 pass
## 12  3    fail
## 13  3.25 pass
## 14  3.5  fail
## 15  4    pass
## 16  4.25 pass
## 17  4.5  pass
## 18  4.75 pass
## 19  5    pass
## 20  5.5  pass
```

Note, at least for yourself, that you have a column called `hours` that is the number of hours each student studied for the exam, and a column called `exam` that says whether each student passed or failed the exam.

A second thing to note is that each line of the data file refers to one student. What I could have done is to group the students who studied for each different number of hours, and record how many of the students in each group passed or failed the exam. But I didn't do that.

(b) (2 marks) Fit a logistic regression predicting whether a student will pass the exam, as it depends on the number of hours they studied. Your column `exam` is text, and will need to be turned into a `factor`. Display the results using `summary` (or, if you prefer, something from `broom`).

Solution: This is `glm` with `family=binomial`. Using `factor(exam)` right in the `glm` is the easiest way to get the pass/fail variable treated properly:

```
study.1  <- glm(factor(exam)~hours, data=study, family=binomial)
summary(study.1)
```

```
##
## Call:
## glm(formula = factor(exam) ~ hours, family = binomial, data = study)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.70557  -0.57357  -0.04654   0.45470   1.82008
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0777     1.7610  -2.316   0.0206 *
## hours         1.5046     0.6287   2.393   0.0167 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27.726  on 19  degrees of freedom
## Residual deviance: 16.060  on 18  degrees of freedom
## AIC: 20.06
##
## Number of Fisher Scoring iterations: 5
```

A check that this is sensible are the words "deviance" and "Fisher scoring iterations" at the bottom of the output.

Another way is to create a new column (or overwrite the old one) creating a factor version of `exam`, for example:

```
study %>% mutate(exam=factor(exam)) -> study
study.1a  <- glm(exam~hours, data=study, family=binomial)
summary(study.1a)
```

```
##
## Call:
## glm(formula = exam ~ hours, family = binomial, data = study)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.70557  -0.57357  -0.04654   0.45470   1.82008
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -4.0777     1.7610  -2.316   0.0206 *
## hours         1.5046     0.6287   2.393   0.0167 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 27.726  on 19  degrees of freedom
## Residual deviance: 16.060  on 18  degrees of freedom
## AIC: 20.06
##
## Number of Fisher Scoring iterations: 5
```

Equally good, and equally full marks.

(c) (2 marks) Is there evidence that the number of hours studying affects the probability of passing the exam? Explain briefly.

Solution: The P-value 0.0167 on the `hours` line is small, so there is indeed evidence that the number of hours studied influences the probability of passing the exam.

To provide a bit more detail: the null hypothesis is that the Estimate for `hours` is zero, which says that the probability of passing the exam is the same no matter how much the student studies. This null hypothesis is rejected, and therefore we conclude that the number of hours studied does indeed have some impact on (probability of) passing the exam.

Extra: the intercept is the *log-odds* of passing the exam for a student who studies no hours (does not study at all). If the intercept is zero (the null hypothesis), this would mean that a student who does not study at all would have a 0.50 probability of passing the exam. This is

also rejected; in fact, the intercept is (very) negative, which means that a student who does not study at all for the exam has a much less than 50–50 chance of passing it.

(d) (2 marks) Is your Estimate for `hours` positive or negative? Does that make sense in the context of the data? Explain briefly.

Solution: Mine is 1.50, positive. This means that a student who studies more hours for the exam has a greater probability of passing it.

To complete the logic, these are the two categories of the response variable (in the order that R knows them):

```
levels(factor(study$exam))
```

```
## [1] "fail" "pass"
```

The first one is the baseline, and we are predicting the probability of the second one, which is indeed the probability of passing the exam.

(e) (4 marks) Obtain predicted probabilities of passing the exam for students who study 0, 1.5, 3, and 4.5 hours. Are these consistent with what you said earlier? Explain briefly.

Solution: First, set up a data frame with the values you want to predict for, in a column called `hours`:

```
new <- tibble(hours=c(0, 1.5, 3, 4.5))
new
```

```
## # A tibble: 4 x 1
##   hours
##   <dbl>
## 1   0
## 2   1.5
## 3   3
## 4   4.5
```

or, if you prefer, this:

```
new <- tribble(
  ~hours,
  0,
  1.5,
  3,
  4.5
)
new
```

```
## # A tibble: 4 x 1
##   hours
##   <dbl>
## 1   0
## 2   1.5
## 3   3
## 4   4.5
```

Then pass your model and this data frame into `predict` with the right `type`:

```
p <- predict(study.1, new, type="response")
```

and then display it side by side with the values these are predictions for:

```
cbind(new, p)
```

```
##   hours          p
## 1   0.0 0.01666378
## 2   1.5 0.13934447
## 3   3.0 0.60735865
## 4   4.5 0.93662366
```

We said earlier that the more a student studies, the better their chances of passing this exam, and the predicted probabilities go clearly up as the number of hours studied increases.

Extra 1: To go back to my note earlier about the intercept, a student who doesn't study at all has a less than 2% chance of passing the exam, which is indeed much less than 50–50.

Extra 2: `cbind` works better than `bind_cols` because sometimes the predictions include other things and the output from `predict` is a `matrix` rather than a data frame (which `bind_cols` will then refuse to combine). In this case, `p` is a vector, so it doesn't matter:[1]

```
bind_cols(new, prob=p)
```

```
## # A tibble: 4 x 2
##   hours   prob
##   <dbl>  <dbl>
## 1   0   0.0167
## 2   1.5 0.139
## 3   3   0.607
## 4   4.5 0.937
```

Extra 3: you can get a nice picture with the confidence interval for the probability (of all students who study a certain amount) passing the exam. The procedure is in a blog post at `https://www.fromthebottomoftheheap.net/2017/05/01/glm-prediction-intervals-i/`. The idea is that you can often make confidence intervals by going up and down two[2] standard errors from the estimate. The point *here* is that you have to do this on the log-odds scale (the scale of the "linear predictor" in `glm` jargon) and then transform back to probabilities. If you don't do that, you'll end up with CIs for probabilities that go above 1 or below 0. To do the transforming back, it's possible to get R to do most of the work for you:

```
ilink <- family(study.1)$linkinv
ilink
```

```
## function (eta)
## .Call(C_logit_linkinv, eta)
## <environment: namespace:stats>
```

This is a function that when given a log-odds, will return you a probability.[3]

Now, let's do some more predictions, but do them on the log-odds scale this time, and get some standard errors for them too:

```
new <- tibble(hours=seq(0, 6, 0.5))
new
```

```
## # A tibble: 13 x 1
##    hours
##    <dbl>
##  1   0
##  2   0.5
##  3   1
##  4   1.5
##  5   2
##  6   2.5
##  7   3
##  8   3.5
##  9   4
```

```
## 10   4.5
## 11   5
## 12   5.5
## 13   6
```

```
p <- predict(study.1, new, type="link", se.fit=T)
preds <- cbind(new, p)
preds
```

```
##    hours        fit    se.fit residual.scale
## 1    0.0 -4.0777134 1.7609843              1
## 2    0.5 -3.3253907 1.4715288              1
## 3    1.0 -2.5730680 1.1947260              1
## 4    1.5 -1.8207453 0.9417992              1
## 5    2.0 -1.0684226 0.7377287              1
## 6    2.5 -0.3160999 0.6317781              1
## 7    3.0  0.4362229 0.6720757              1
## 8    3.5  1.1885456 0.8377769              1
## 9    4.0  1.9408683 1.0722391              1
## 10   4.5  2.6931910 1.3398379              1
## 11   5.0  3.4455137 1.6242772              1
## 12   5.5  4.1978364 1.9180797              1
## 13   6.0  4.9501591 2.2175268              1
```

A log-odds of zero is a probability of 0.5, and thus, according to this, studying between 2.5 and 3.0 hours will get you a 50% chance of passing the exam. This is consistent with what we saw before.

On this scale, predictions are more accurate near the middle of the data (as we saw in regression).

Next, compute the confidence limits, on the log-odds scale:

```
preds %>%
    mutate(lower_log_odds=fit-2*se.fit, upper_log_odds=fit+2*se.fit)
```

```
##    hours        fit    se.fit residual.scale lower_log_odds upper_log_odds
## 1    0.0 -4.0777134 1.7609843              1    -7.59968205     -0.5557448
## 2    0.5 -3.3253907 1.4715288              1    -6.26844824     -0.3823332
## 3    1.0 -2.5730680 1.1947260              1    -4.96252003     -0.1836160
## 4    1.5 -1.8207453 0.9417992              1    -3.70434377      0.0628532
## 5    2.0 -1.0684226 0.7377287              1    -2.54387990      0.4070348
## 6    2.5 -0.3160999 0.6317781              1    -1.57965604      0.9474563
## 7    3.0  0.4362229 0.6720757              1    -0.90792854      1.7803742
## 8    3.5  1.1885456 0.8377769              1    -0.48700822      2.8640994
## 9    4.0  1.9408683 1.0722391              1    -0.20360995      4.0853465
## 10   4.5  2.6931910 1.3398379              1     0.01351525      5.3728667
## 11   5.0  3.4455137 1.6242772              1     0.19695924      6.6940682
## 12   5.5  4.1978364 1.9180797              1     0.36167702      8.0339958
## 13   6.0  4.9501591 2.2175268              1     0.51510551      9.3852128
```

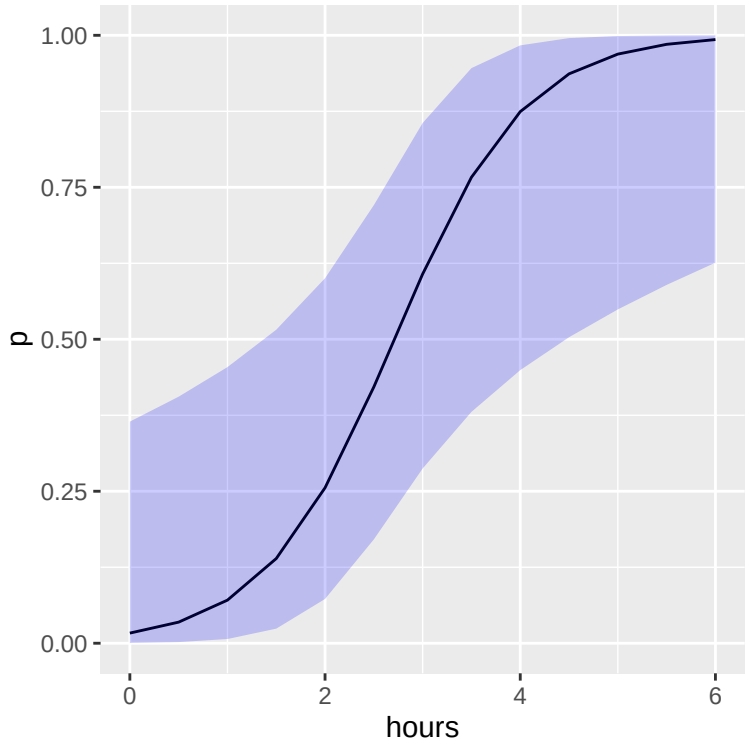
Now, turn them back into probabilities and save what we need. This is where that `ilink` function comes in:

```
preds %>%
    mutate(lower_log_odds=fit-2*se.fit, upper_log_odds=fit+2*se.fit) %>%
    mutate(lower=ilink(lower_log_odds),
           p=ilink(fit),
           upper=ilink(upper_log_odds)) %>%
```

```
    select(hours, lower, p, upper) -> preds2
preds2
##    hours        lower          p     upper
## 1    0.0 0.0005003601 0.01666378 0.3645326
## 2    0.5 0.0018915823 0.03471034 0.4055643
## 3    1.0 0.0069466833 0.07089196 0.4542245
## 4    1.5 0.0240249588 0.13934447 0.5157081
## 5    2.0 0.0728387163 0.25570318 0.6003767
## 6    2.5 0.1708442000 0.42162653 0.7206033
## 7    3.0 0.2874239099 0.60735865 0.8557431
## 8    3.5 0.3805986064 0.76648084 0.9460429
## 9    4.0 0.4492726419 0.87444750 0.9834608
## 10   4.5 0.5033787617 0.93662366 0.9953806
## 11   5.0 0.5490812464 0.96909707 0.9987633
## 12   5.5 0.5894463327 0.98519444 0.9996759
## 13   6.0 0.6260025614 0.99296752 0.9999161
```

Now to plot these. This uses a new thing called `geom_ribbon` which I explain below:

```
ggplot(preds2, aes(x=hours, y=p, ymin=lower, ymax=upper)) +
    geom_line() + geom_ribbon(fill="blue", alpha=0.2)
```



`geom_ribbon` needs a `ymin` and a `ymax`, and draws a coloured stripe between `ymin` and `ymax` all the way up. The colour is controlled by `fill`; I also used a small value of `alpha`[4] to make the stripe see-through.

The black curve shows the predicted probabilities of passing for each number of hours studied, with the upward trend we saw before. The blue stripe shows how certain we are about those predicted probabilities; with only 20 students, the answer is "pretty uncertain" and the stripe is wide. Note that the predicted probability is not necessarily near the middle of the stripe,

particularly at the top and bottom. It was in the middle on the log-odds scale, but the back-transformation is non-linear. For example, with 6 hours of study, we predict a student to be almost certain to pass, but the confidence interval goes all the way down to 0.675.

You see that the stripe never goes below 0 or above 1, so the confidence intervals always contain reasonable probabilities.

This stripe is about the same width all the way across. You'd expect it to be narrower in the middle and wider at the ends, but probabilities close to 0 and 1 don't have far to vary, and probabilities in the middle can vary more, so this is fighting the "more nearby data" thing, and it's not clear which will win.