Week 9 Exercise - Multiple Logistic Regression Model Answers

This week we are learning how to estimate logistic regression models in R. For this example, we will be using data from the US Census and the Southern Poverty Law Center about active hate groups in the United States (source: https://www.splcenter.org/hate-map).

In this practical we are going to explore which state-level variables are associated with higher or lower likelihoods of the state having at least one active Neo-Nazi or anti-immigration hate group in 2020.

```
# Load packages we will be using: remember to install them
# first with install.packages() if you do not have them.
library(tidyverse) # for data tidying and plotting
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.5
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.3      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggeffects) # for partial plots
library(caret) # for testing accuracy
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(performance) # for testing accuracy through k-fold validation
library(broom) # for turning model summaries into tibbles with tidy()

# Read in the data sample
hate_data <- read_rds("hate_group_data.rds")

# Print a preview of the data
hate_data
```

```
## # A tibble: 52 x 10
##    state neo_n~1 anti_~2 pc_ba~3 popul~4 pc_ag~5 pc_bl~6 pc_hh~7 latit~8 longi~9
##    <chr>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>
##  1 Alab~       0       0    25.5   49.0     17.3    26.8    76.4    32.4   -86.3
##  2 Alas~       0       0    29.6    7.32    12.5     3.7    85.5    58.3  -134.
##  3 Ariz~       1       1    29.5   72.8     18       5.2    84.1    33.4  -112.
##  4 Arka~       0       0    23     30.2     17.4    15.7    73      34.7   -92.3
##  5 Cali~       1       1    33.9  395.      14.8     6.5    86.7    38.6  -121.
##  6 Colo~       1       0    40.9   57.6     14.6     4.6    87.6    39.7  -105.
```

```
##  7 Conn~       0       0    39.3   35.7     17.7    12.2    85.5    41.8   -72.7
##  8 Dela~       0       0    32      9.74    19.4    23.2    85      39.2   -75.5
##  9 Dist~       0       1    58.5    7.06    12.4    46      82.6    38.9   -77.0
## 10 Flor~       1       1    29.9  215.      20.9    16.9    83      30.4   -84.3
## # ... with 42 more rows, and abbreviated variable names 1: neo_nazi_hgs,
## #   2: anti_imig_hgs, 3: pc_bachelors_ed, 4: population, 5: pc_age_65_plus,
## #   6: pc_black_pop, 7: pc_hh_with_internet, 8: latitude, 9: longitude
```

Part I

Logistic Regression: Predicting States with Active Neo-Nazi Hate Groups

In this case, the researchers want to see how the percentage of the population with Bachelors or higher levels of education (`pc_bachelors_ed`), the percentage of the population aged 65 plus (`pc_age_65_plus`), the percentage of the population who identify as black in the US census (`pc_black_pop`), the latitude of the state (to see if hate groups are more common in Southern states or Northern States) (`latitude`) and the percentage of households in the state with access to the internet (`pc_hh_with_internet`), change the probabilities of a state having an active hate group. They also want to control for the size of the population (in 100,000s) (`population`).

They create a logistic regression model using the `glm()` function and a `family = binomial(link = "logit")` argument.

```
model_1 <- glm(data = hate_data,
               formula = neo_nazi_hgs ~
                           pc_bachelors_ed + population + pc_age_65_plus +
                           pc_black_pop + pc_hh_with_internet + latitude,
               family = binomial(link = "logit"))

summary(model_1)
```

```
##
## Call:
## glm(formula = neo_nazi_hgs ~ pc_bachelors_ed + population + pc_age_65_plus +
##     pc_black_pop + pc_hh_with_internet + latitude, family = binomial(link = "logit"),
##     data = hate_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.3114  -0.7134  -0.3202   0.6150   1.8313
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)        -30.36398   18.24318  -1.664  0.09603 .
## pc_bachelors_ed     -0.17747    0.12619  -1.406  0.15962
## population           0.03288    0.01256   2.618  0.00885 **
## pc_age_65_plus       0.10634    0.22307   0.477  0.63355
## pc_black_pop        -0.07122    0.06370  -1.118  0.26354
## pc_hh_with_internet  0.41041    0.22950   1.788  0.07374 .
## latitude            -0.03429    0.07323  -0.468  0.63958
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 67.350  on 50  degrees of freedom
## Residual deviance: 45.455  on 44  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 59.455
##
## Number of Fisher Scoring iterations: 6
```

Remember that, in this case we have data for all US states in 2020 - this means we may not be particularly interested in the p-values and are more interested in the effects of each variable (as we may not be trying to generalise to a larger population).

- Using the `exp()` function to create exponentiated odds, describe how changes in the percentage of the population with bachelors or higher education (`pc_bachelors_ed`) is associated with changes in the likelihood of a state having an active Neo-Nazi hate group.

Each additional percentage point of the population with bachelors or higher level education was associated with a 16% reduction in the likelihood of the state having an active Neo-Nazi hate group.

- Using the `exp()` function to create exponentiated odds, describe how changes in the population size (`population`) is associated with changes in the likelihood of a state having an active Neo-Nazi hate group.

Every 100,000 people in the population of a state was associated with a 3.34% increase in the odds of the state having an active Neo-Nazi hate group.

- Using the `exp()` function to create exponentiated odds, describe how changes in the percentage of people aged 65 or over (`pc_age_65_plus`) is associated with changes in the likelihood of a state having an active Neo-Nazi hate group.

Each 1 percentage point increase in the percentage of the population aged 65 or over was associated with a 11.22% increase in the odds of the state having an active Neo-Nazi hate group.

- Using the `exp()` function to create exponentiated odds, describe how changes in the percentage of the population who identify as black in the census (`pc_black_pop`) is associated with changes in the likelihood of a state having an active Neo-Nazi hate group.

Each 1 percentage point increase in the percentage of the population identifying as black in the US census was associated with a 6.9% decrease in the odds of the state having an active Neo-Nazi hate group.

- Using the `exp()` function to create exponentiated odds, describe how changes in the percentage of households with access to the internet (`pc_hh_with_internet`) is associated with changes in the likelihood of a state having an active Neo-Nazi hate group.

Each 1 percentage point increase in the percentage of households with access to the internet was associated with a 50.7% increase in the odds of the state having an active Neo-Nazi hate group.

- Lastly, using the `exp()` function to create exponentiated odds, describe how changes in the latitude of the state's capital (`latitude`) is associated with changes in the likelihood of a state having an active Neo-Nazi hate group. Keep in mind that an increase of 1 degree in latitude means the state is further North.

Each 1 degree increase in the latitude (further North) of a state's capital was associated with a 3.37% reduction in the odds of the state having an active Neo-Nazi hate group.

They decide to tidy their model using `broom::tidy()` so that they can create a new column with the exponentiated odds for each independent variable using the `mutate` function.

```
# Tidy model
model_1_tidy <- broom::tidy(model_1)
```

```
# Add exponentiated odds
model_1_tidy %>%
  mutate(
    exp_odds = exp(estimate)
  )
```

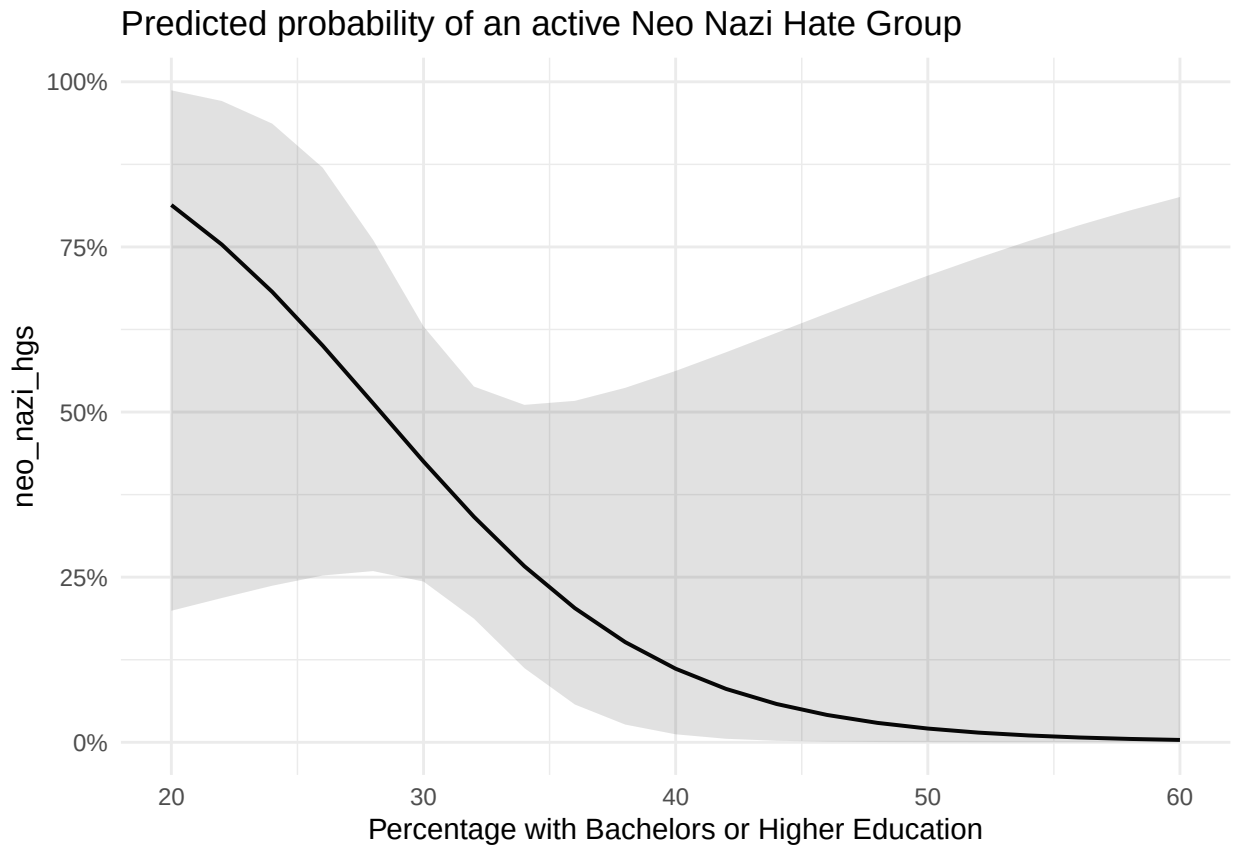
```
## # A tibble: 7 x 6
##   term                estimate std.error statistic p.value exp_odds
##   <chr>                  <dbl>     <dbl>     <dbl>   <dbl>    <dbl>
## 1 (Intercept)         -30.4      18.2       -1.66  0.0960  6.50e-14
## 2 pc_bachelors_ed      -0.177     0.126     -1.41  0.160   8.37e- 1
## 3 population            0.0329    0.0126     2.62  0.00885 1.03e+ 0
## 4 pc_age_65_plus        0.106     0.223      0.477 0.634   1.11e+ 0
## 5 pc_black_pop         -0.0712    0.0637    -1.12  0.264   9.31e- 1
## 6 pc_hh_with_internet   0.410     0.230      1.79  0.0737  1.51e+ 0
## 7 latitude             -0.0343    0.0732    -0.468 0.640   9.66e- 1
```

Note that these figures are written in scientific notation. 8.37e- 1 is equal to 0.837; 1.03e+ 0 is equal to 1.03; 9.31e- 1 is equal to 0.931; and so on.

- Check that your answers above match the exponentiated odds generated above. If they differ, why do they differ? This will also help you practice interpreting scientific notation.

Next, the researchers decide to try and plot the predicted probabilities for a few of their variables: `pc_bachelors_ed`, `population`, `pc_hh_with_internet`, and `latitude`. They use the `ggeffects` package to achieve this.

```
# ggeffects:ggeffect can quickly plot predicted probabilities for both categorical
# and continuous predictors - especially useful for logistic regression
# models! Note that any plot you create with ggeffect can be added to using
# standard ggplot functions, like ggtitle, xlab, or theme_classic!
ggeffect(model_1, terms = "pc_bachelors_ed") %>%
  plot() +
  ggtitle("Predicted probability of an active Neo Nazi Hate Group") +
  xlab("Percentage with Bachelors or Higher Education") +
  theme_minimal()
```



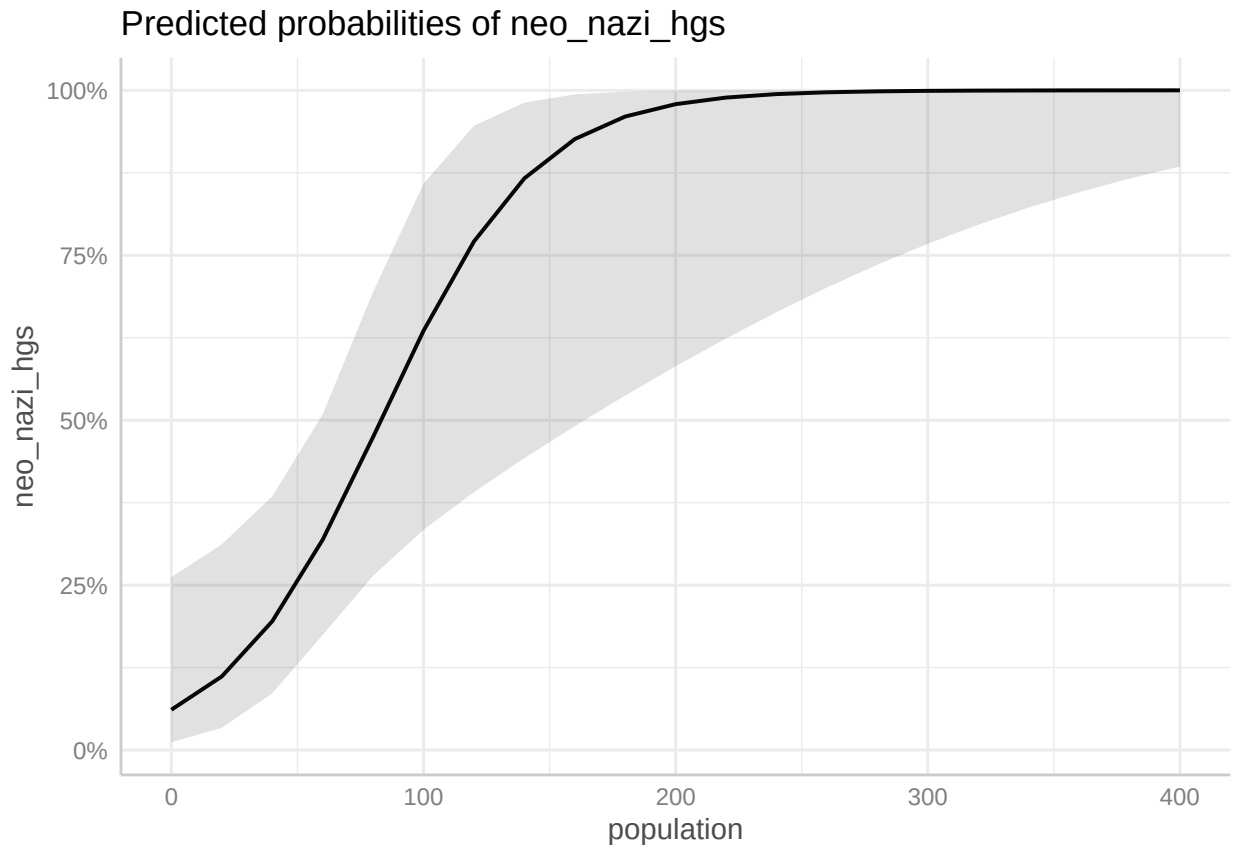
- What is the predicted probability (approximately) of active Neo-Nazi hate groups in states with a bachelors degree/higher education rate of 25%?

Around 62.5%.

- What is the predicted probability (approximately) of a state having active Neo-Nazi hate groups in states with a bachelors degree or higher education rate of 40%?

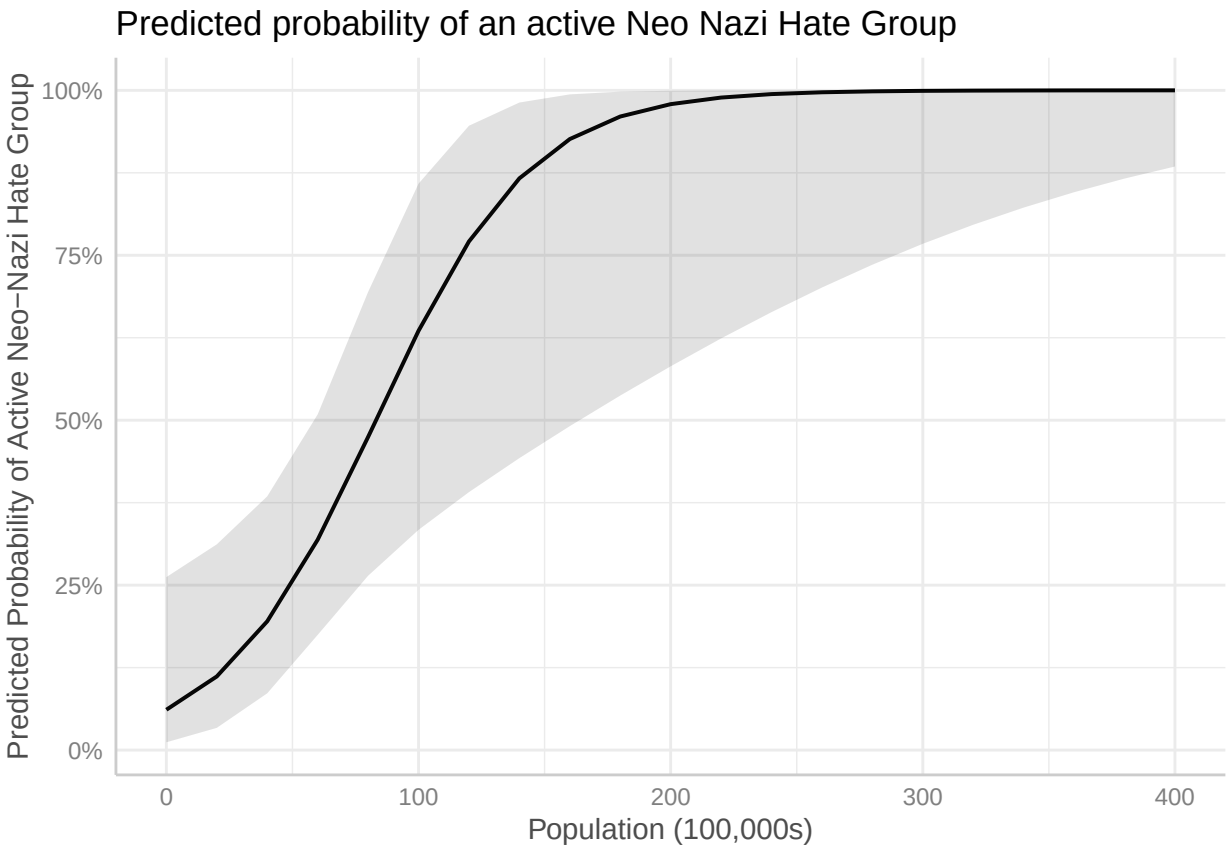
Around 12.5%.

```
ggeffect(model_1, terms = "population") %>%
  plot()
```



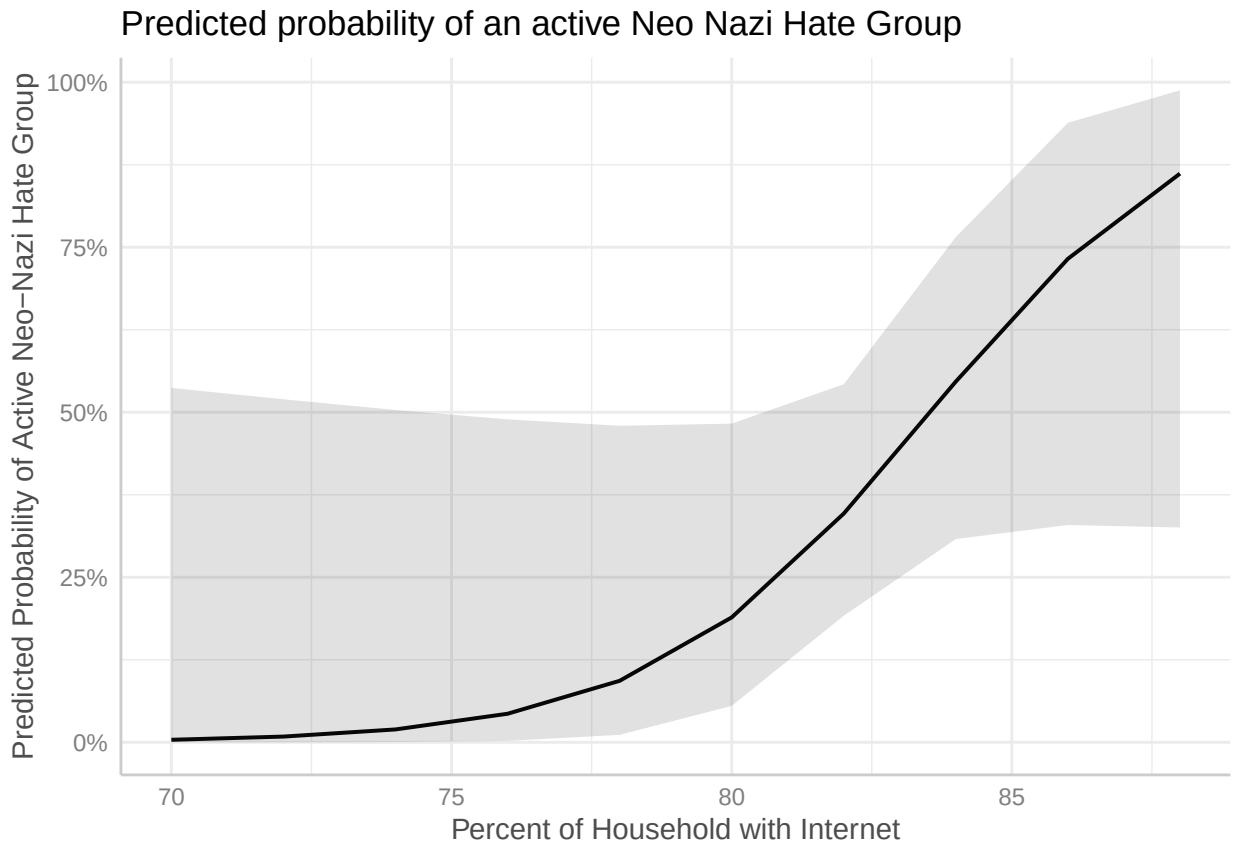
- Add a sensible title and X and Y labels to the plot to make it look nicer, using added ggplot code.

```
ggeffect(model_1, terms = "population") %>%
  plot() +
  ggtitle("Predicted probability of an active Neo Nazi Hate Group") +
  xlab("Population (100,000s)") +
  ylab("Predicted Probability of Active Neo-Nazi Hate Group")
```



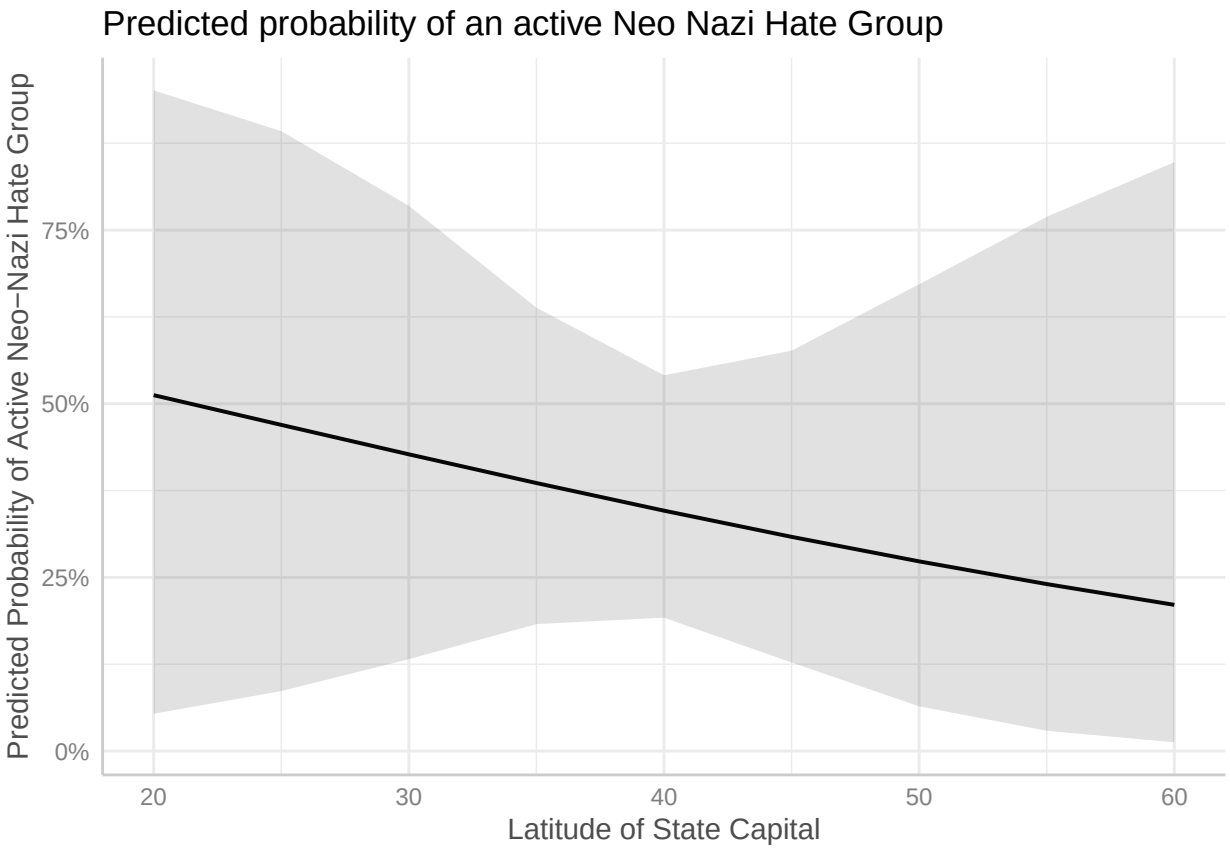
- Create a predicted probabilities plot for the remaining two variables of interest: the percentage of households with internet access (`pc_hh_with_internet`) and the latitude of the state capital (`latitude`).
- The effect of the percentage of households with internet access

```
ggeffect(model_1, terms = "pc_hh_with_internet") %>%
  plot() +
  ggtitle("Predicted probability of an active Neo Nazi Hate Group") +
  xlab("Percent of Household with Internet") +
  ylab("Predicted Probability of Active Neo-Nazi Hate Group")
```



- The effect of the latitude of the state capital

```
ggeffect(model_1, terms = "latitude") %>%
  plot() +
  ggtitle("Predicted probability of an active Neo Nazi Hate Group") +
  xlab("Latitude of State Capital") +
  ylab("Predicted Probability of Active Neo-Nazi Hate Group")
```



Next, the researchers decide they want to find out how accurate their model is. In other words, if they had information about all of their independent variables, but not about whether there were any active Neo-Nazi hate groups in the state, how accurately would they be able to guess whether there were any? This is one way of assessing model fit when we cannot use something like the R-squared value.

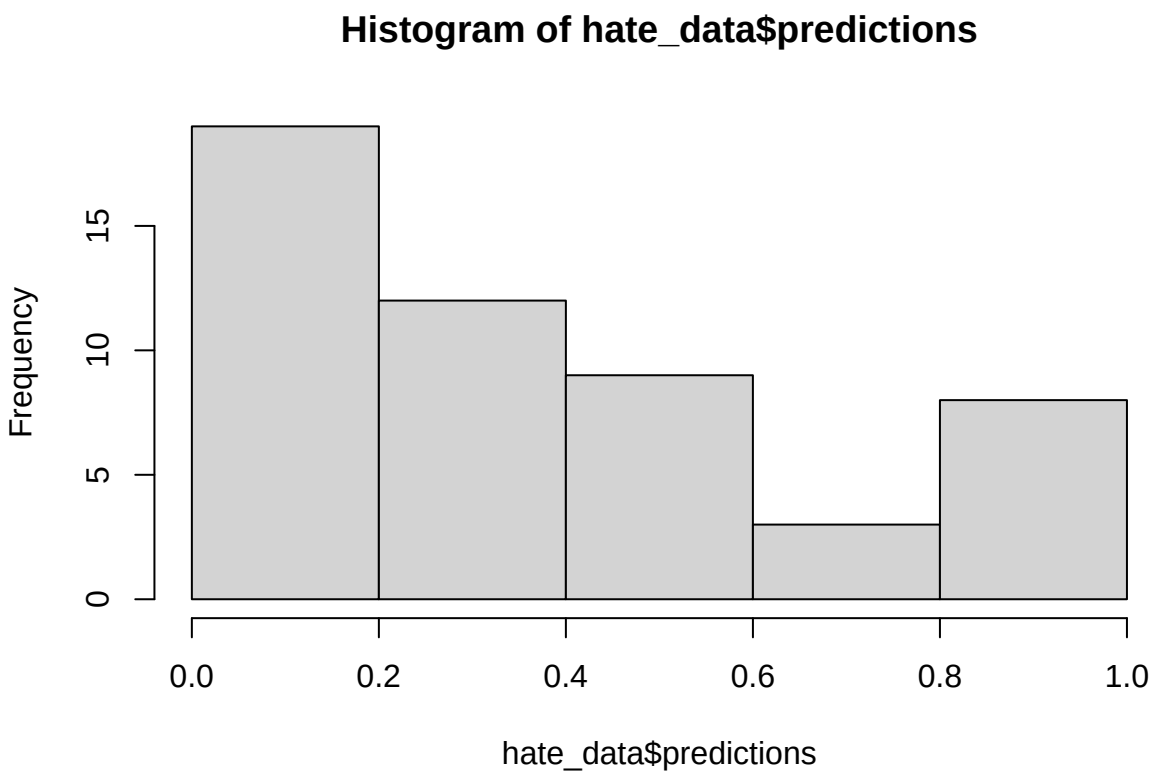
First, they want to use a conventional Confusion Matrix and measurement of accuracy. In order to do this, they first need to calculate the predictions from their model.

The `predict` function can be used to generate the predicted probability for all cases in the sample, based on the model. This is added as a new column to the data using the mutate function

```
hate_data <- hate_data %>%
  mutate(
    predictions = predict(model_1, type = "response", newdata = hate_data)
  )
```

You can plot a simple histogram of the predictions to see how they look. They should look like a spread of proportions (between 0 and 1).

```
hist(hate_data$predictions)
```



We want to test how many predictions would have been correct - to do this we are going to say that predictions equal to 50% or greater will be our model predicting that the state *does* have an active Neo-Nazi group (1), whereas predictions less than 50% will be our model predicting that the state *does not* have an active Neo-Nazi group (0)

```
hate_data <- hate_data %>%
  mutate(
    # Overwrite the predictions variable where prediction >= 0.5 is 1 and
    # a prediction less than 0.5 is 0. Keep missing as missing
    predictions = case_when(is.na(predictions) ~ NA_real_,
                            predictions >= 0.5 ~ 1,
                            TRUE ~ 0
                            )
  )
```

We can use the 'confusionMatrix()' function from the `caret` package to calculate the accuracy of our model predictions.

```
caret::confusionMatrix(data = factor(hate_data$predictions),
                       reference = factor(hate_data$neo_nazi_hgs))
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  0  1
##          0 28  7
##          1  4 12
##
```

```
##                Accuracy : 0.7843
##                  95% CI : (0.6468, 0.8871)
##     No Information Rate : 0.6275
##     P-Value [Acc > NIR] : 0.0126
##
##                   Kappa : 0.5234
##
##  Mcnemar's Test P-Value : 0.5465
##
##             Sensitivity : 0.8750
##             Specificity : 0.6316
##          Pos Pred Value : 0.8000
##          Neg Pred Value : 0.7500
##              Prevalence : 0.6275
##          Detection Rate : 0.5490
##    Detection Prevalence : 0.6863
##       Balanced Accuracy : 0.7533
##
##        'Positive' Class : 0
##
```

The caret confusion matrix gives us quite a lot of information, but the important parts for our purposes are: the two-by-two table at the top, the Accuracy value, the No Information Rate, and the probability that the model's accuracy is better than the No Information Rate (P-Value [Acc > NIR]).

The Confusion Matrix The 2x2 table tells us how our predictions (rows) matched up with or deviated from our real values. In this case, our model was able to correctly predict 12 of the 19 states with active Neo-Nazi hate groups. It was also able to correctly identify 28 of the 32 states that did not have active Neo-Nazi hate groups.

Accuracy The accuracy value gives us the proportion of cases in our data that were correctly classified by the logistic regression model. Here, our model correctly classified around 78.4% of states as having or not having active Neo-Nazi hate groups.

No Information Rate The No Information Rate tells us the proportion of the largest group (0 or 1) we would expect to be able to get correct just by knowing the proportion of states that had or did not have Neo-Nazi hate groups, on average, and then randomly guessing for each state. In this case, if we had just randomly guessed we would be expected to have correctly classified around 62.75% of the largest group (states without active Neo-Nazi hate groups).

P-Value [Acc > NIR] This is the p-value for whether our model performs significantly better than what we would expect from the random range of correct classifications we would expect to get with no information. If it is less than 0.05, we could reject the idea that our model is no better than just randomly guessing.

A confusion matrix is not the only way we can assess accuracy. We can also use something called k-fold cross-validation. K-fold cross validation provides a measure of accuracy by randomly splitting the sample into a subset of mutually exclusive groups (k subsets) before checking how well the model performs on all of those different groups on average.

Because we have such a small sample size it wouldn't be wise for us to choose a high number of 'folds', but we can still check accuracy with the k-fold cross-validation algorithm using the `performance` R package.

```
# Here I set the random number generator using set.seed() so that the whole class gets the same results
# because the k-folds will be chosen in the same way
set.seed(10000)
```

```
# Then I run the performance_accuracy function from the performance package
performance::performance_accuracy(model_1, k = 3)
```

```
## # Accuracy of Model Predictions
##
## Accuracy: 78.73%
##       SE: 7.38%-points
##   Method: Area under Curve
```

The k-fold cross-validation seems to be telling us approximately the same thing as our conventional Confusion Matrix accuracy test - the average accuracy across folds was around 78.73%. However, we probably can't be too sure in this case because our population is so small — generally a good number of k-folds is 10, but if we tried to do this with our data it would mean only around 5 cases would be in each fold.

Finally, the researchers checked for any issues of multicollinearity in their independent variables. The `performance` package includes a function to check this in much the same way as the `car` package.

```
performance::multicollinearity(model_1)
```

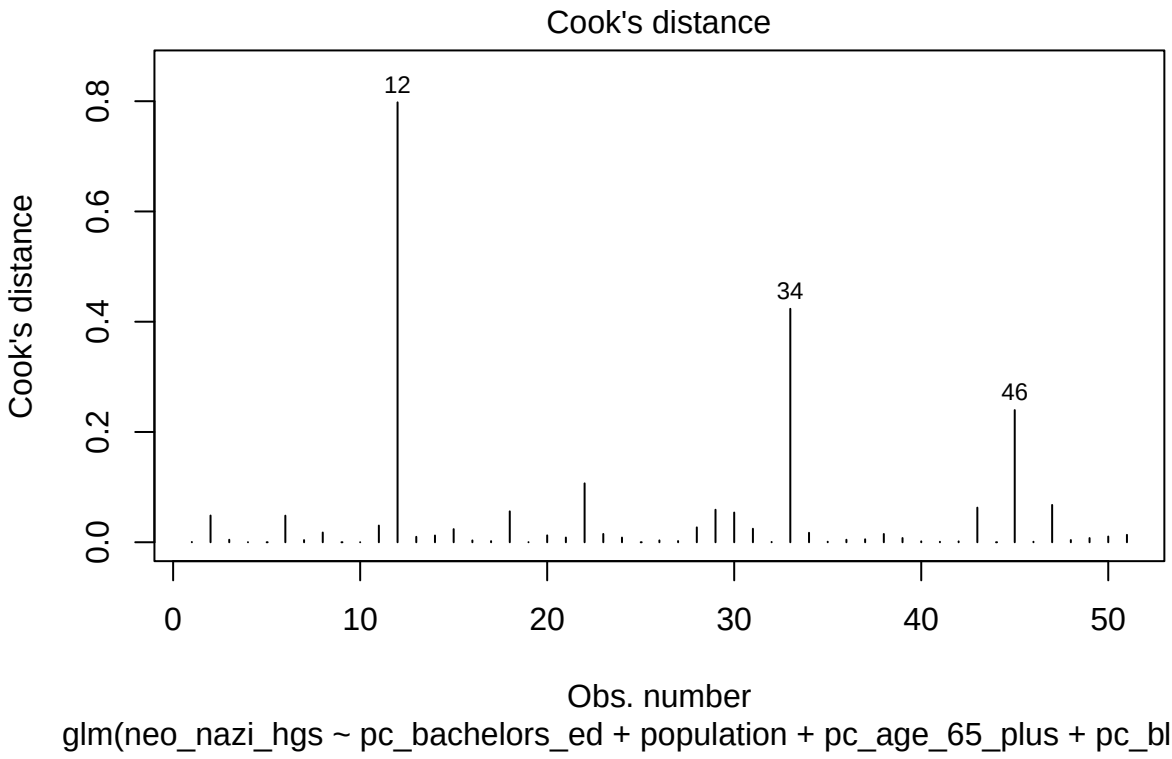
```
## # Check for Multicollinearity
##
## Low Correlation
##
##                 Term  VIF   VIF 95% CI Increased SE Tolerance Tolerance 95% CI
##      pc_bachelors_ed 3.13 [2.22, 4.72]         1.77      0.32     [0.21, 0.45]
##           population 1.69 [1.32, 2.50]         1.30      0.59     [0.40, 0.76]
##       pc_age_65_plus 1.26 [1.07, 1.97]         1.12      0.79     [0.51, 0.93]
##         pc_black_pop 1.93 [1.47, 2.86]         1.39      0.52     [0.35, 0.68]
##  pc_hh_with_internet 3.51 [2.46, 5.30]         1.87      0.28     [0.19, 0.41]
##             latitude 1.20 [1.04, 1.97]         1.10      0.83     [0.51, 0.96]
```

- Was there any evidence of problematic multicollinearity in this model, based on the VIF statistics?

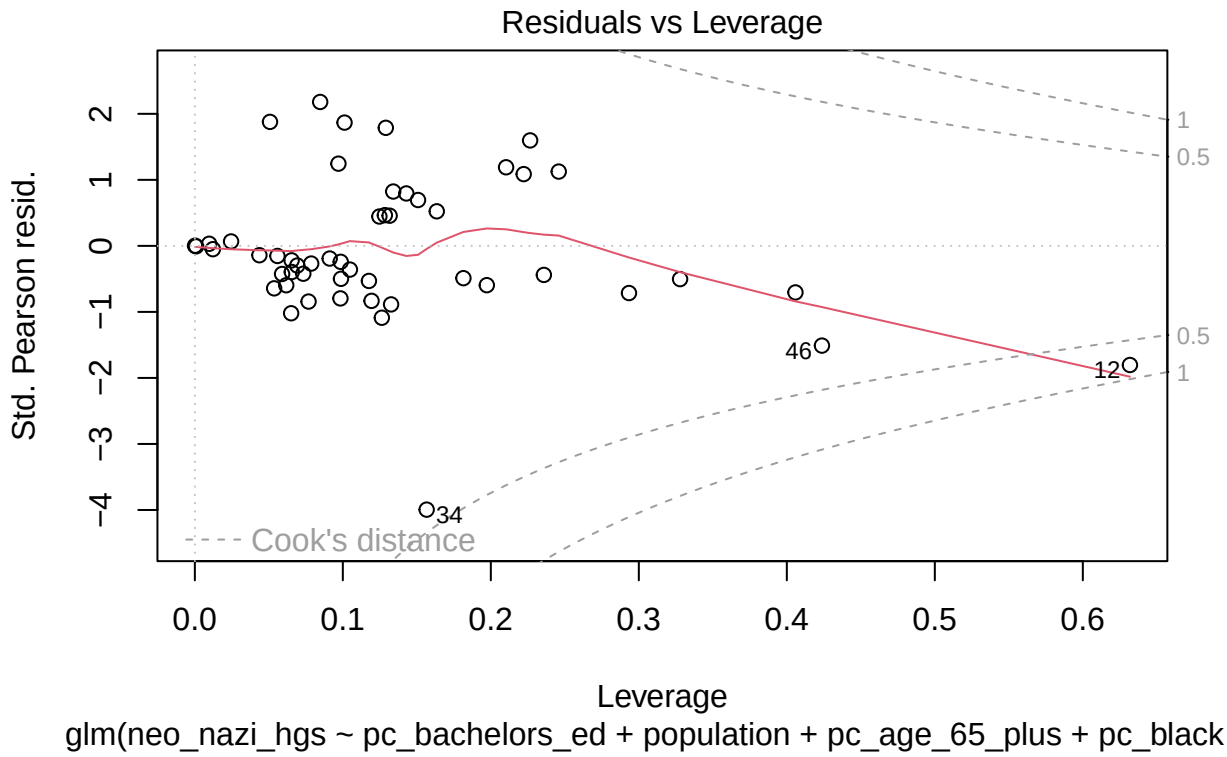
The VIF statistics were all below 5, which is generally considered a cut off for problematic multicollinearity.

We can also check for outliers using the same plot function we used in multiple linear regression. The `performance` package also has a function to check for outliers, but it is not always very informative.

```
plot(model_1, which = 4)
```




```
plot(model_1, which = 5)
```




```
performance::check_outliers(model_1)
```

```
## OK: No outliers detected.
## - Based on the following method and threshold: cook (0.92).
## - For variable: (Whole model)
```

- Was there any evidence of outliers or leverage points? If so, which states may be outliers and what would you recommend doing to ensure the model estimates are not bias?

The plots seem to suggest that state 12 (Hawaii), state 34 (Minnesota), and state 46 (Utah) may be substantial leverage points or outliers, however, the performance method of checking for outliers does not suggest any problematic outliers. It may be beneficial to try re-running the model with the outliers removed in order to check that they are not disproportionately impacting the results.

Lastly, the researchers check whether there is any evidence of autocorrelation using a Durbin-Watson test.

```
car::durbinWatsonTest(model_1)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.1346422      1.707809    0.35
##  Alternative hypothesis: rho != 0
```

- Was there any evidence of significant autocorrelation within the residuals from the model?

There was no significant evidence of autocorrelation among the residuals in the model. The Durbin Watson test statistic had an associated p-value that was greater than 0.05, which means we cannot reject the null hypothesis that the residuals are independent.

Part II

Logistic Regression: Predicting States with Active Anti-Immigrant Hate Groups

- Using the above code as a template, create a similar logistic regression model predicting whether states have an active anti-immigrant hate group below.

```
model_2 <- glm(data = hate_data,
                  formula = anti_imig_hgs ~
                              pc_bachelors_ed + population + pc_age_65_plus +
                              pc_black_pop + pc_hh_with_internet + latitude,
                  family = binomial(link = "logit"))

summary(model_2)
```

```
##
## Call:
## glm(formula = anti_imig_hgs ~ pc_bachelors_ed + population +
##     pc_age_65_plus + pc_black_pop + pc_hh_with_internet + latitude,
##     family = binomial(link = "logit"), data = hate_data)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -1.02446  -0.16115  -0.06712  -0.03111   2.64932
##
## Coefficients:
##                      Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -3.494718  43.614672  -0.080   0.9361
## pc_bachelors_ed      0.259433   0.191031   1.358   0.1744
## population           0.026463   0.013065   2.025   0.0428 *
## pc_age_65_plus       0.064850   0.532167   0.122   0.9030
## pc_black_pop         0.016504   0.121549   0.136   0.8920
## pc_hh_with_internet -0.006085   0.544099  -0.011   0.9911
## latitude            -0.308094   0.151555  -2.033   0.0421 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 32.717  on 50  degrees of freedom
## Residual deviance: 11.406  on 44  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 25.406
##
## Number of Fisher Scoring iterations: 8
```

- Use the `broom::tidy()` function to create a tidy tibble version of your model results, and then create a new column in this tibble for the exponentiated (`exp()`) coefficients of the changes in log odds.

```
model_2_tidy <- broom::tidy(model_2)

model_2_tidy <- model_2_tidy %>%
  mutate(
    exp_odds = exp(estimate)
  )
```

```
model_2_tidy
```

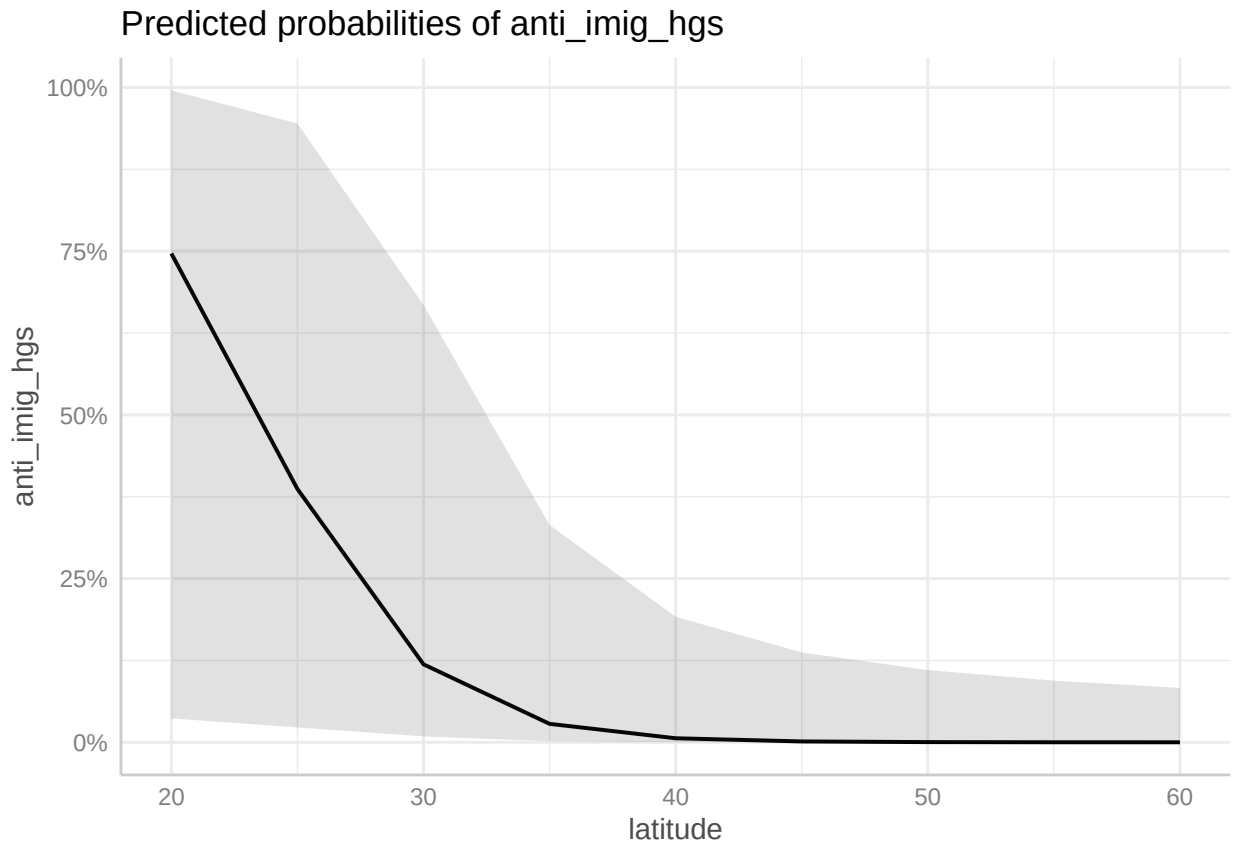
```
## # A tibble: 7 x 6
##   term                estimate std.error statistic p.value exp_odds
##   <chr>                  <dbl>     <dbl>     <dbl>   <dbl>    <dbl>
## 1 (Intercept)         -3.49      43.6      -0.0801  0.936    0.0304
## 2 pc_bachelors_ed      0.259      0.191     1.36    0.174    1.30
## 3 population           0.0265     0.0131    2.03    0.0428   1.03
## 4 pc_age_65_plus       0.0648     0.532     0.122   0.903    1.07
## 5 pc_black_pop         0.0165     0.122     0.136   0.892    1.02
## 6 pc_hh_with_internet -0.00608    0.544    -0.0112  0.991    0.994
## 7 latitude            -0.308      0.152    -2.03    0.0421   0.735
```

- Interpret the coefficients in your model: how are changes in each independent variable associated with changes in the log odds of a state having an active anti-immigrant hate group?

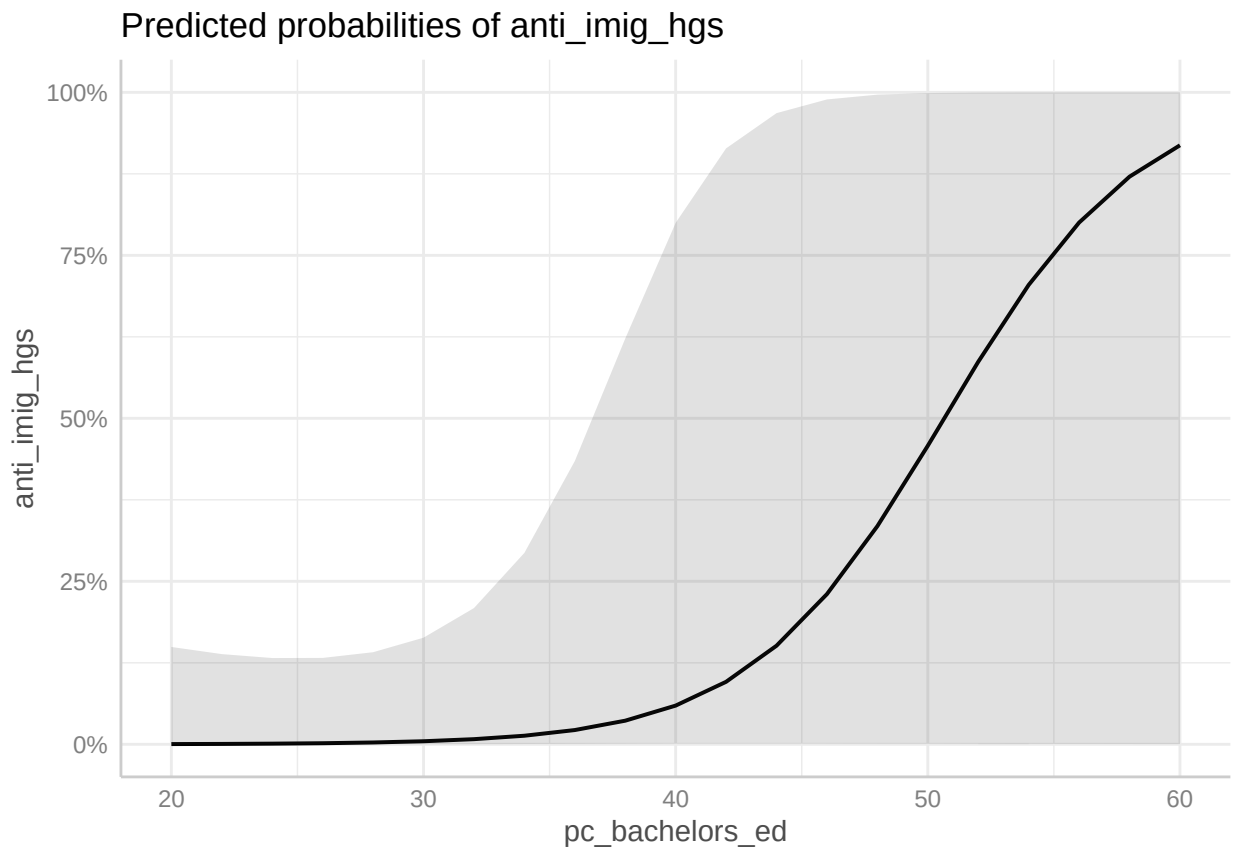
An increase of 1 percentage point in the percentage of the population with Bachelors or higher level education was associated with a 1.3 times increase in the odds of a state having an active anti-immigrant hate group. An increase in the population size of a state of 100,000 was associated with a further increase of around 3%. An increase of 1 percentage point in the percentage of the population who are aged 65+ was associated with a 7% increase in the odds of an active anti-immigration hate group and increases of 1 percentage point in the percentage of the population identifying as black in the population was associated with a 2% increase in the likelihood of a state having an active anti-immigration hate group. Higher percentages of households with access to the internet were associated with only slighly reduced odds of a state having an active anti-immigration hate group (around a 0.6% decrease per 1 percentage point increase in the percentage of households with access to the internet). Lastly, an increase of 1 degree in state capital latitude (indicating a more Northern state) was associated with a 26.5 per cent decrease in the likelihood of a state having an active anti-immigrant hate group.

- Pick two of the results that you find particularly interesting and create partial plots showing the predicted probabilities of a state having an active anti-immigrant hate group across values of these variables.

```
ggeffect(model_2, terms = "latitude") %>%
  plot()
```



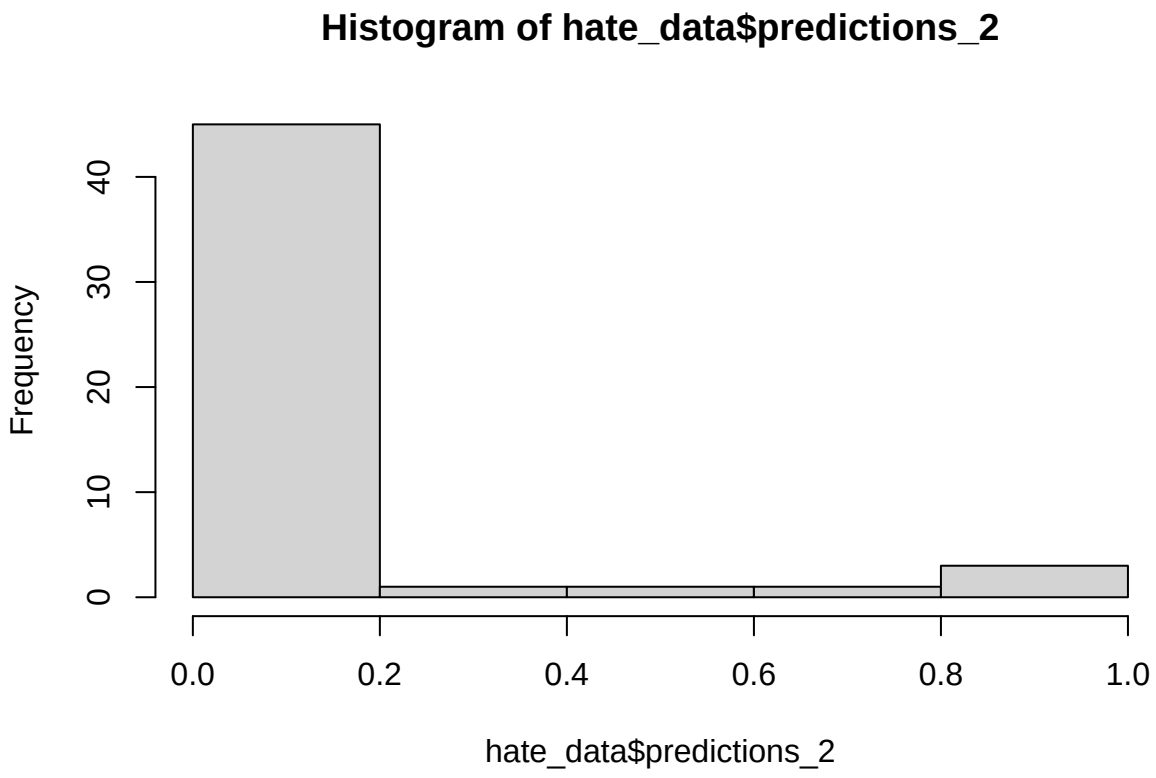

```
ggeffect(model_2, terms = "pc_bachelors_ed") %>%
  plot()
```



Mutate the predictions required to run the `confusionMatrix()` function from the `caret` package, and then run this function.

```
hate_data <- hate_data %>%
  mutate(
    predictions_2 = predict(model_2, type = "response", newdata = hate_data)
  )

hist(hate_data$predictions_2)
```




```
hate_data <- hate_data %>%
  mutate(
    # Overwrite the predictions variable where prediction >= 0.5 is 1 and
    # a prediction less than 0.5 is 0. Keep missing as missing
    predictions_2 = case_when(is.na(predictions) ~ NA_real_,
                              predictions >= 0.5 ~ 1,
                              TRUE ~ 0
                              )
  )

caret::confusionMatrix(data = factor(hate_data$predictions_2),
                       reference = factor(hate_data$anti_imig_hgs))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction  0  1
##          0 34  1
##          1 12  4
##
##                Accuracy : 0.7451
##                  95% CI : (0.6037, 0.8567)
##     No Information Rate : 0.902
##     P-Value [Acc > NIR] : 0.999709
##
##                   Kappa : 0.2722
```

```
##
##  Mcnemar's Test P-Value : 0.005546
##
##             Sensitivity : 0.7391
##             Specificity : 0.8000
##          Pos Pred Value : 0.9714
##          Neg Pred Value : 0.2500
##              Prevalence : 0.9020
##          Detection Rate : 0.6667
##    Detection Prevalence : 0.6863
##       Balanced Accuracy : 0.7696
##
##        'Positive' Class : 0
##
```

- Report the results of the confusion matrix, the accuracy rate, the no information rate, and the p-value associated with the difference between your model's accuracy and the no information accuracy rate.

The confusion matrix for this model shows that the logistic regression produced a large number of false-positives, it predicted that 12 of the 46 states without active anti-immigrant hate groups actually did have active anti-immigration hate groups. However, it did successfully identify 4 of the 5 states that did have active anti-immigrant hate groups. Overall, this resulted in an accuracy of 74.51%, which was significantly worse than the no information rate (90.2%, p = 0.9997). None of the independent variables in the model therefore appear to be very effective at consistently identifying which states have active anti-immigration hate groups.

Try assessing your model's accuracy using k-fold validation with the `performance::performance_accuracy()` function from the `performance` package.

```
set.seed(10000)
performance::performance_accuracy(model_2, k = 3)
```

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

## Warning: glm.fit: algorithm did not converge

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

## # Accuracy of Model Predictions
##
## Accuracy: 76.67%
##       SE: 27.28%-points
##   Method: Area under Curve
```

We also need to check our assumptions. First, check whether there are any issues with multicollinearity in your model.

```
performance::multicollinearity(model_2)
```

```
## # Check for Multicollinearity
##
## Low Correlation
##
##                 Term  VIF   VIF 95% CI Increased SE Tolerance Tolerance 95% CI
##      pc_bachelors_ed 4.67 [3.20, 7.12]         2.16      0.21     [0.14, 0.31]
##           population 1.72 [1.33, 2.54]         1.31      0.58     [0.39, 0.75]
```

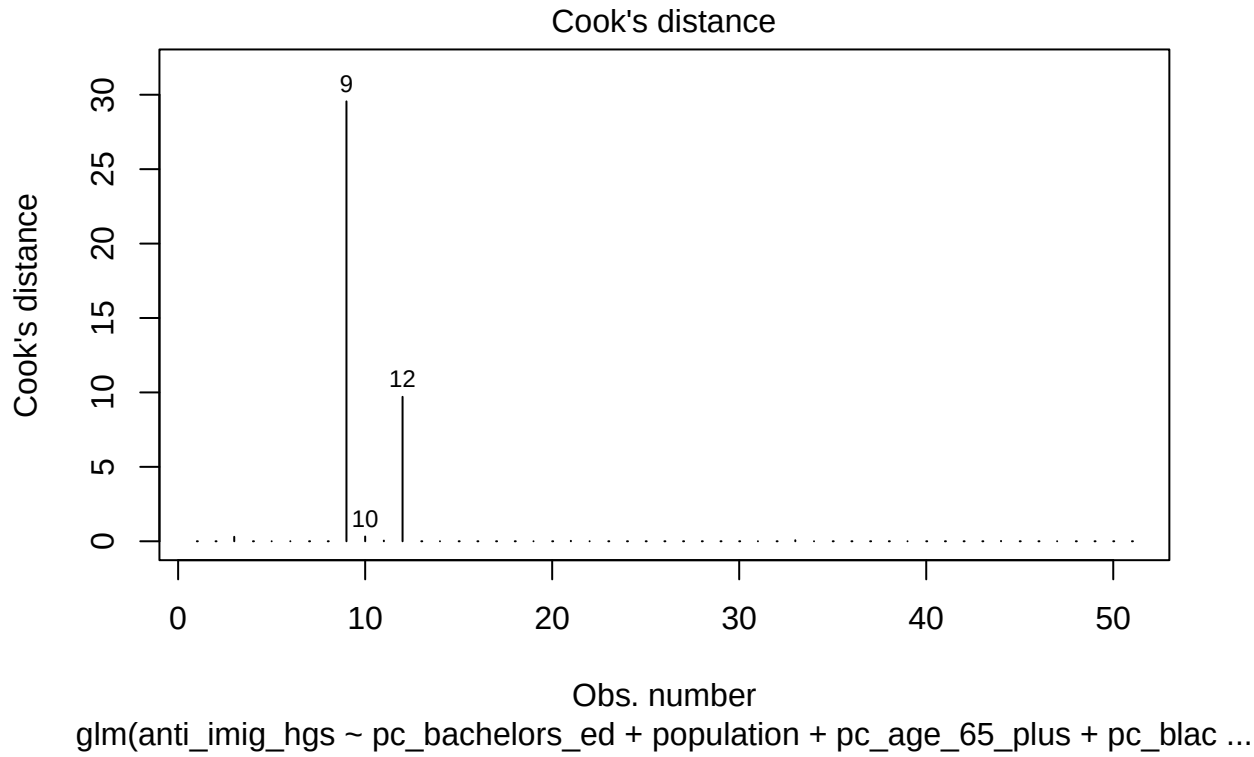
```
##       pc_age_65_plus 2.60 [1.89, 3.89]         1.61      0.38     [0.26, 0.53]
##         pc_black_pop 4.24 [2.93, 6.44]         2.06      0.24     [0.16, 0.34]
##  pc_hh_with_internet 2.24 [1.66, 3.33]         1.50      0.45     [0.30, 0.60]
##             latitude 1.76 [1.36, 2.61]         1.33      0.57     [0.38, 0.73]
```

- Report the VIF results below. Are there any problems with multicollinearity in the model? If so, why? If not, why not?

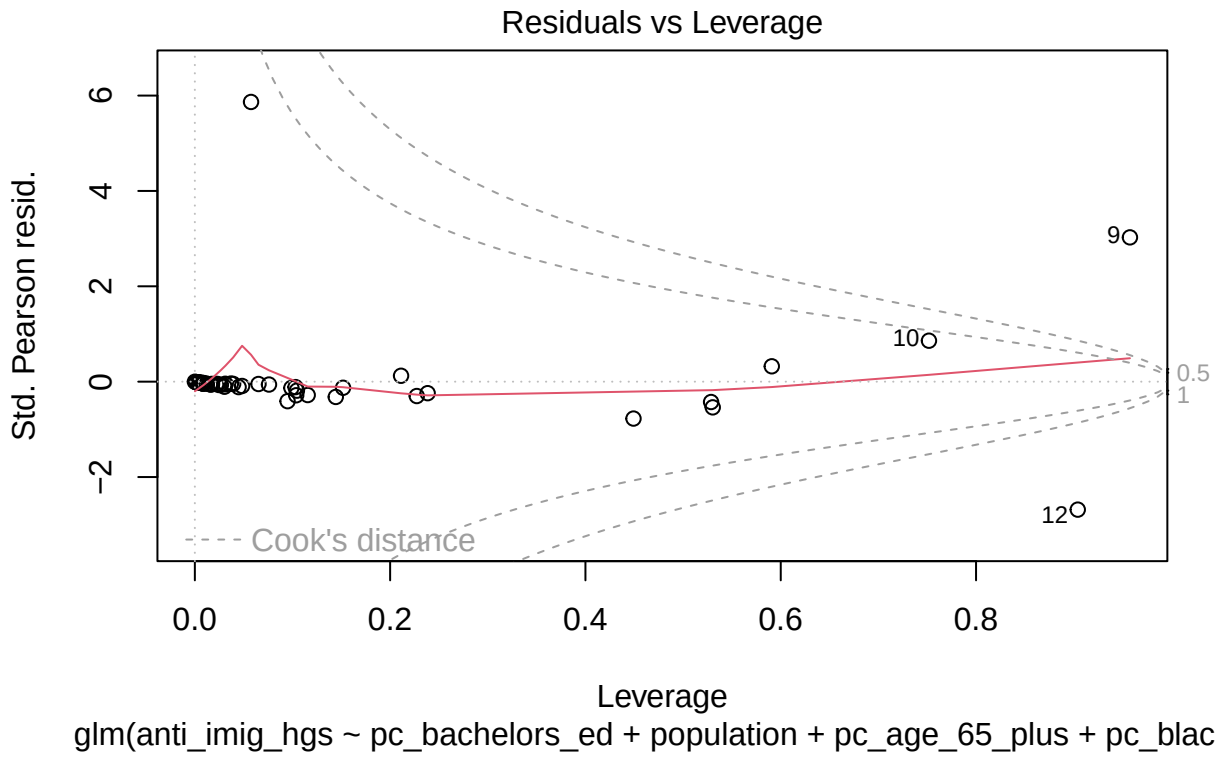
While none of the independent variables had a VIF value greater than 5, two variables had VIF values that were approaching 5 (pc_bachelors_ed & pc_black_pop). It may be worthwhile removing one of these variables and re-running the model to see if estimates change substantially or if model fit improves.

Next, write the code to check whether there are any outliers using both the `performance` package and the base `plot()` package.

```
plot(model_2, which = 4)
```




```
plot(model_2, which = 5)
```




```
performance::check_outliers(model_2)
```

```
## 2 outliers detected: cases 9, 12.
## - Based on the following method and threshold: cook (0.92).
## - For variable: (Whole model).
```

- Are there any states you might consider outliers? If there are potential outliers, what would you recommend doing to check if these are unduly influencing the estimates of some independent variables?

The Cook's Distance plots seem to suggest that observation 9 (District of Columbia (DC)), observation 10 (Florida), and observation 12 (Hawaii) may be significant outliers that are biasing model estimates. The output from the performance package also identifies DC and Hawaii as significant outliers. It may be beneficial to attempt to re-estimate the model with these outliers removed to observe how estimates change.

Lastly, check whether the residuals from this model are independent (that there is no autocorrelation). You can do this using either the `car` or the `performance` packages.

```
performance::check_autocorrelation(model_2)
```

```
## OK: Residuals appear to be independent and not autocorrelated (p = 0.436).
```

```
car::durbinWatsonTest(model_2)
```

```
##  lag Autocorrelation D-W Statistic p-value
##    1       0.1088105      1.780314    0.41
##  Alternative hypothesis: rho != 0
```

The Durbin Watson test statistic indicates that there is no significant evidence of autocorrelation (D-W Statistic = 1.78, p = 0.41).

Week 9 Challenge

- For more practice, create a new binary categorical variable where 1 is equal to a state having neither an active Neo Nazi hate group, nor an active anti-immigration hate group, and 0 is equal to the state having either an active Neo-Nazi hate group, or an active anti-immigration hate group, or both. Then, create a logistic regression model predicting states that have neither active anti-immigration nor active Neo-Nazi hate groups and report the results.

However, if you feel like you'd prefer to challenge yourself by looking at some data you might use for assessment two feel free to do that instead!