lab-07

Justin Chan

3/1/2022

Getting Started

Packages

```
library(tidyverse)
library(broom)
library(pROC)
library(plotROC)
library(knitr)
```

Exercises

Part I: Data Prep & Modeling

1. Read through the Spotify documentation page to learn more about the variables in the dataset. The response variable for this analysis is target, where 1 indicates the user likes the song and 0 otherwise. Let's prepare the response and some predictor variables before modeling.

- If needed, change target so that it is factor variable type in R.
- Change key so that it is a factor variable type in R, which takes values "D" if key==2, "D#" if key==3, and "Other" for all other values.
- Plot the relationship between target and key. Briefly describe the relationship between the two variables.

```
spotify_data <- read_csv("~/Desktop/School/STAT108/lab-07/raw_data/spotify.csv")
```

```
## New names:
## * `` -> ...1
```

```
## Rows: 2017 Columns: 17
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): song_title, artist
## dbl (15): ...1, acousticness, danceability, duration_ms, energy, instrumenta...
```

```
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
glimpse(spotify_data)
```

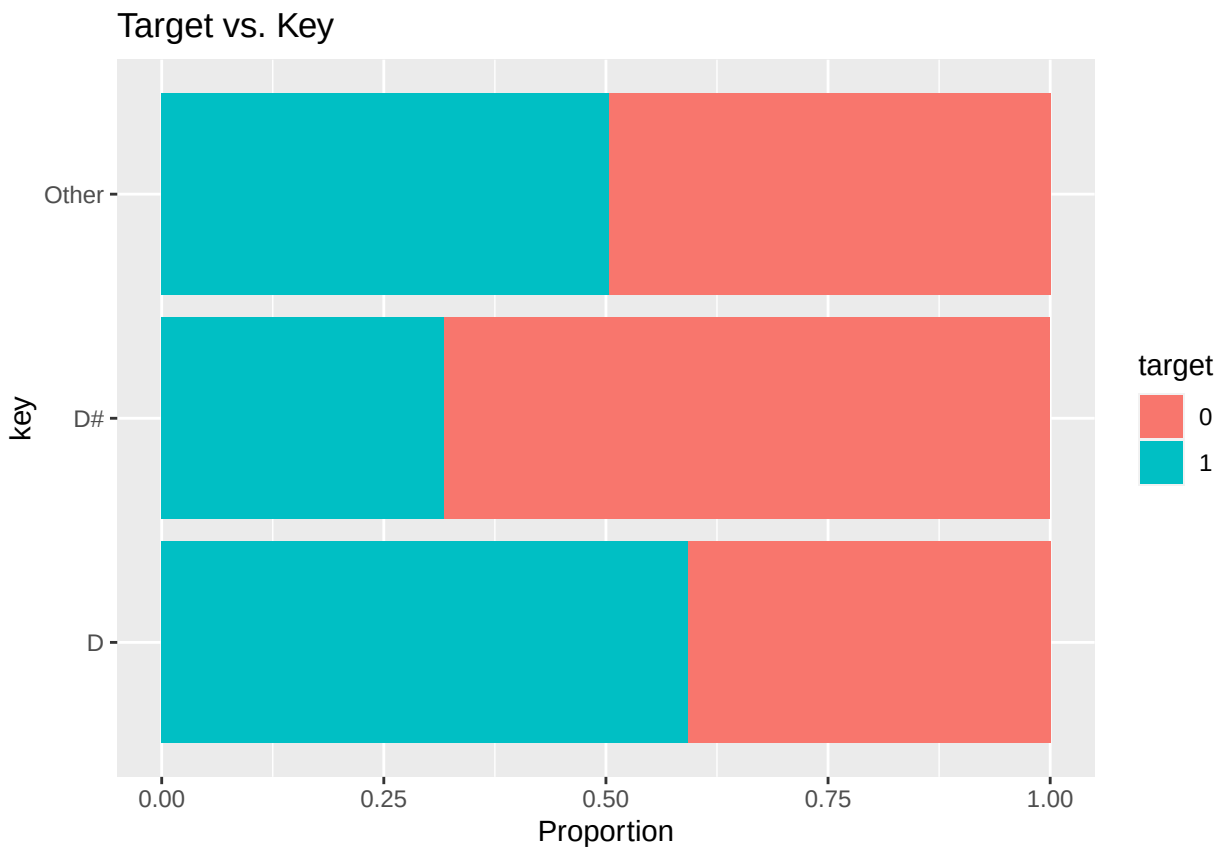
```
## Rows: 2,017
## Columns: 17
## $ ...1             <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,~
## $ acousticness     <dbl> 0.010200, 0.199000, 0.034400, 0.604000, 0.180000, 0.0~
## $ danceability     <dbl> 0.833, 0.743, 0.838, 0.494, 0.678, 0.804, 0.739, 0.26~
## $ duration_ms      <dbl> 204600, 326933, 185707, 199413, 392893, 251333, 24140~
## $ energy           <dbl> 0.434, 0.359, 0.412, 0.338, 0.561, 0.560, 0.472, 0.34~
## $ instrumentalness <dbl> 2.19e-02, 6.11e-03, 2.34e-04, 5.10e-01, 5.12e-01, 0.0~
## $ key              <dbl> 2, 1, 2, 5, 5, 8, 1, 10, 11, 7, 5, 10, 0, 0, 9, 6, 1,~
## $ liveness         <dbl> 0.1650, 0.1370, 0.1590, 0.0922, 0.4390, 0.1640, 0.207~
## $ loudness         <dbl> -8.795, -10.401, -7.148, -15.236, -11.648, -6.682, -1~
## $ mode             <dbl> 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0,~
## $ speechiness      <dbl> 0.4310, 0.0794, 0.2890, 0.0261, 0.0694, 0.1850, 0.156~
## $ tempo            <dbl> 150.062, 160.083, 75.044, 86.468, 174.004, 85.023, 80~
## $ time_signature   <dbl> 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 4, 4, 4, 4, 4,~
## $ valence          <dbl> 0.286, 0.588, 0.173, 0.230, 0.904, 0.264, 0.308, 0.39~
## $ target           <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ song_title       <chr> "Mask Off", "Redbone", "Xanny Family", "Master Of Non~
## $ artist           <chr> "Future", "Childish Gambino", "Future", "Beach House"~
```

```
spotify_data <- spotify_data %>%
  mutate(key = if_else(key == 2, "D", if_else(key == 3, "D#", "Other")),
         target = as.factor(target))

glimpse(spotify_data)
```

```
## Rows: 2,017
## Columns: 17
## $ ...1             <dbl> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,~
## $ acousticness     <dbl> 0.010200, 0.199000, 0.034400, 0.604000, 0.180000, 0.0~
## $ danceability     <dbl> 0.833, 0.743, 0.838, 0.494, 0.678, 0.804, 0.739, 0.26~
## $ duration_ms      <dbl> 204600, 326933, 185707, 199413, 392893, 251333, 24140~
## $ energy           <dbl> 0.434, 0.359, 0.412, 0.338, 0.561, 0.560, 0.472, 0.34~
## $ instrumentalness <dbl> 2.19e-02, 6.11e-03, 2.34e-04, 5.10e-01, 5.12e-01, 0.0~
## $ key              <chr> "D", "Other", "D", "Other", "Other", "Other", "Other"~
## $ liveness         <dbl> 0.1650, 0.1370, 0.1590, 0.0922, 0.4390, 0.1640, 0.207~
## $ loudness         <dbl> -8.795, -10.401, -7.148, -15.236, -11.648, -6.682, -1~
## $ mode             <dbl> 1, 1, 1, 1, 0, 1, 1, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0,~
## $ speechiness      <dbl> 0.4310, 0.0794, 0.2890, 0.0261, 0.0694, 0.1850, 0.156~
## $ tempo            <dbl> 150.062, 160.083, 75.044, 86.468, 174.004, 85.023, 80~
## $ time_signature   <dbl> 4, 4, 4, 4, 4, 4, 4, 4, 4, 4, 3, 4, 4, 4, 4, 4, 4, 4,~
## $ valence          <dbl> 0.286, 0.588, 0.173, 0.230, 0.904, 0.264, 0.308, 0.39~
## $ target           <fct> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,~
## $ song_title       <chr> "Mask Off", "Redbone", "Xanny Family", "Master Of Non~
## $ artist           <chr> "Future", "Childish Gambino", "Future", "Beach House"~
```

```
ggplot(data = spotify_data, aes(x = key, fill = target)) +
  geom_bar(position = "fill") +
  labs(y = "Proportion",
       title = "Target vs. Key") +
  coord_flip()
```



2. Fit a logistic regression model with target as the response variable and the following as predictors: acousticness, danceability, duration_ms, instrumentalness, loudness, speechiness, and valence. Display the model output.

```
target_spotify_log_model <- glm(target ~ acousticness + danceability + duration_ms + instrumentalness +
              data = spotify_data, family = binomial)
tidy(target_spotify_log_model, conf.int = TRUE, exponentiate = FALSE) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value	conf.low	conf.high
(Intercept)	-2.955	0.276	-10.693	0	-3.504	-2.420
acousticness	-1.722	0.240	-7.182	0	-2.197	-1.257
danceability	1.630	0.344	4.737	0	0.958	2.308
duration_ms	0.000	0.000	4.225	0	0.000	0.000
instrumentalness	1.353	0.207	6.549	0	0.952	1.763
loudness	-0.087	0.017	-5.062	0	-0.122	-0.054
speechiness	4.072	0.583	6.985	0	2.947	5.234

term	estimate	std.error	statistic	p.value	conf.low	conf.high
valence	0.856	0.223	3.836	0	0.420	1.296

3. We consider adding key to the model. Conduct the appropriate test to determine if key should be included in the model. Display the output from the test and write your conclusion in the context of the data.

```
full_spotify_log_model <- glm(target ~ acousticness + danceability + duration_ms + instrumentalness + lo
              data = spotify_data, family = binomial)

tidy(target_spotify_log_model)
```

```
## # A tibble: 8 x 5
##   term                estimate   std.error statistic  p.value
##   <chr>                  <dbl>       <dbl>     <dbl>    <dbl>
## 1 (Intercept)      -2.96       0.276          -10.7  1.10e-26
## 2 acousticness     -1.72       0.240           -7.18 6.89e-13
## 3 danceability      1.63       0.344            4.74 2.17e- 6
## 4 duration_ms       0.00000287 0.000000680      4.23 2.39e- 5
## 5 instrumentalness  1.35       0.207            6.55 5.80e-11
## 6 loudness         -0.0874     0.0173          -5.06 4.14e- 7
## 7 speechiness       4.07       0.583            6.98 2.85e-12
## 8 valence           0.856      0.223            3.84 1.25e- 4
```

```
tidy(full_spotify_log_model)
```

```
## # A tibble: 10 x 5
##    term                estimate   std.error statistic  p.value
##    <chr>                  <dbl>       <dbl>     <dbl>    <dbl>
##  1 (Intercept)      -2.51       0.311           -8.07 7.14e-16
##  2 acousticness     -1.70       0.241           -7.07 1.60e-12
##  3 danceability      1.65       0.345            4.77 1.80e- 6
##  4 duration_ms       0.00000286 0.000000684      4.19 2.82e- 5
##  5 instrumentalness  1.38       0.207            6.67 2.60e-11
##  6 loudness         -0.0866     0.0173          -5.02 5.21e- 7
##  7 speechiness       4.03       0.585            6.90 5.33e-12
##  8 valence           0.881      0.224            3.93 8.61e- 5
##  9 keyD#            -1.07       0.335           -3.20 1.36e- 3
## 10 keyOther         -0.494      0.169           -2.92 3.47e- 3
```

```
anova(target_spotify_log_model, full_spotify_log_model, test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model 1: target ~ acousticness + danceability + duration_ms + instrumentalness +
##     loudness + speechiness + valence
## Model 2: target ~ acousticness + danceability + duration_ms + instrumentalness +
##     loudness + speechiness + valence + key
##   Resid. Df Resid. Dev Df Deviance Pr(>Chi)
## 1      2009     2518.5
```

```
## 2      2007     2505.2  2   13.357 0.001258 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Since the p-value for the chi-square test against the full and reduced model is 0.001258, we reject the null hypothesis that adding the key into the logistic model is statistically significant. In other words, the p-value seems too little when comparing the accuracy of the reduced and full models meaning there is a difference between them, insinuating that key adds to the accuracy of the full model. We can say that there is at least one new term in the table that is statistically significant.

4. Display the model you chose in Exercise 3. If appropriate, interpret the coefficient for keyD# in the context of the data. Otherwise, state why it's not appropriate to interpret this coefficient.

```
tidy(full_spotify_log_model)
```

```
## # A tibble: 10 x 5
##    term                estimate   std.error statistic  p.value
##    <chr>                  <dbl>       <dbl>     <dbl>    <dbl>
##  1 (Intercept)      -2.51       0.311           -8.07 7.14e-16
##  2 acousticness     -1.70       0.241           -7.07 1.60e-12
##  3 danceability      1.65       0.345            4.77 1.80e- 6
##  4 duration_ms       0.00000286 0.000000684      4.19 2.82e- 5
##  5 instrumentalness  1.38       0.207            6.67 2.60e-11
##  6 loudness         -0.0866     0.0173          -5.02 5.21e- 7
##  7 speechiness       4.03       0.585            6.90 5.33e-12
##  8 valence           0.881      0.224            3.93 8.61e- 5
##  9 keyD#            -1.07       0.335           -3.20 1.36e- 3
## 10 keyOther         -0.494      0.169           -2.92 3.47e- 3
```

The coefficient of keyD# would be -1.07. It would not be appropriate to interpret this coefficient to compare the direct relationship between the predictor variable, keyD# and the base line for the model since this is logistic model. However, it is appropriate to interpret the coefficient, -1.07 for keyD# as the odds of target for the group, keyD# are exp{-1.07} times the odds of target of the baseline group.

Part II: Checking Assumptions

5. Use the augment function to calculate the predicted probabilities and corresponding residuals.

```
spotify_aug <- augment(full_spotify_log_model, type.predict = "response",
                       type.residuals = "deviance")
spotify_aug
```

```
## # A tibble: 2,017 x 15
##    target acousticness danceability duration_ms instrumentalness loudness
##    <fct>         <dbl>        <dbl>       <dbl>            <dbl>    <dbl>
##  1 1           0.0102         0.833      204600       0.0219        -8.80
##  2 1           0.199          0.743      326933       0.00611      -10.4
##  3 1           0.0344         0.838      185707       0.000234      -7.15
##  4 1           0.604          0.494      199413       0.51         -15.2
##  5 1           0.18           0.678      392893       0.512        -11.6
##  6 1           0.00479        0.804      251333       0             -6.68
```

```
##  7 1           0.0145         0.739      241400       0.00000727   -11.2
##  8 1           0.0202         0.266      349667       0.664        -11.6
##  9 1           0.0481         0.603      202853       0             -3.63
## 10 1           0.00208        0.836      226840       0             -7.79
## # ... with 2,007 more rows, and 9 more variables: speechiness <dbl>,
## #   valence <dbl>, key <chr>, .fitted <dbl>, .resid <dbl>, .std.resid <dbl>,
## #   .hat <dbl>, .sigma <dbl>, .cooksd <dbl>
```

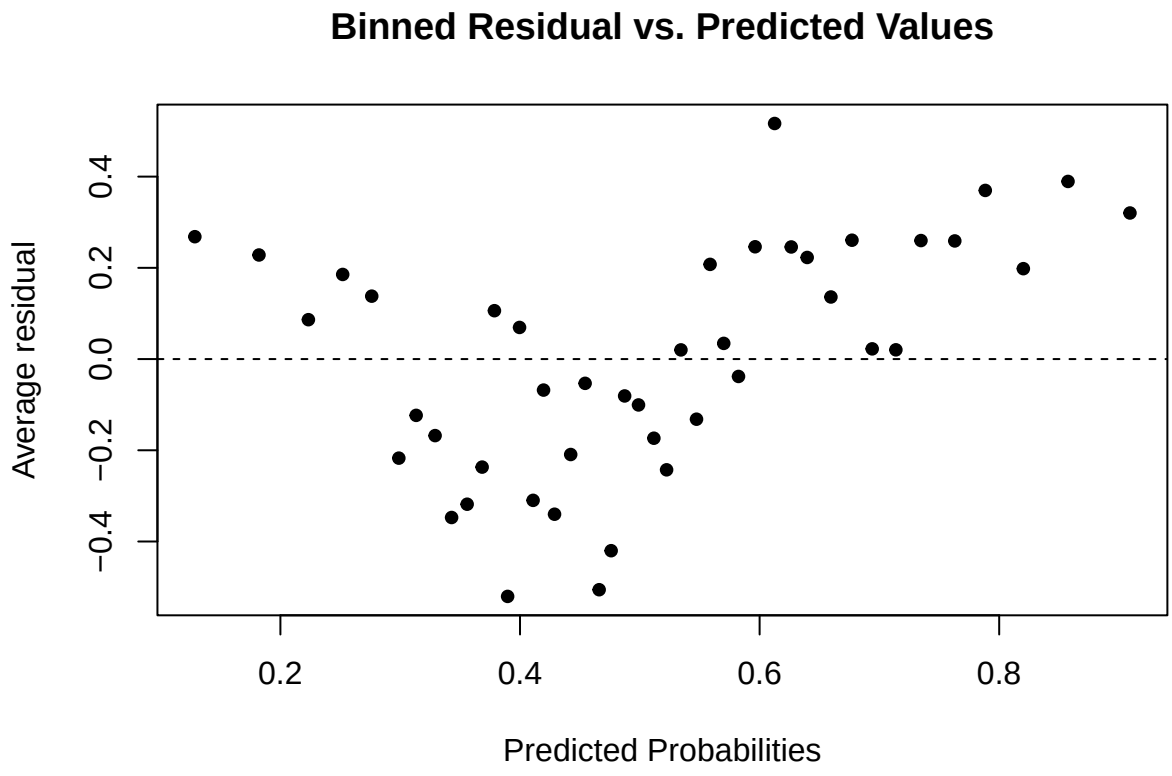
6. Create a binned plot of the residuals versus the predicted probabilities.

```
nbins <- sqrt(nrow(spotify_aug))

spotify_aug %>%
  arrange(.fitted) %>%
  summarise(mean_resid = mean(.resid), #y axis
            mean_pred = mean(.fitted)) #x axis
```

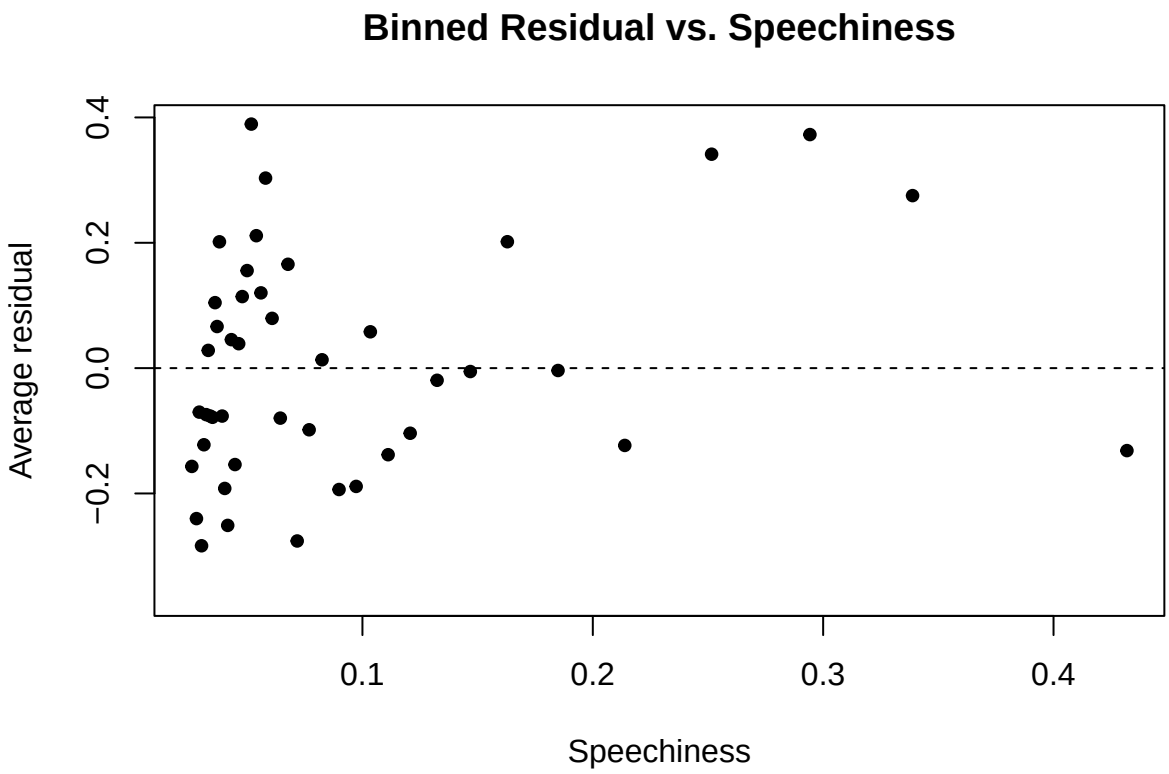
```
## # A tibble: 1 x 2
##   mean_resid mean_pred
##        <dbl>     <dbl>
## 1    0.00461     0.506
```

```
arm::binnedplot(x = spotify_aug$.fitted, y = spotify_aug$.resid,
                xlab = "Predicted Probabilities",
                main = "Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



7. Choose a quantitative predictor in the final model. Make the appropriate table or plot to examine the residuals versus this predictor variable.

```
arm::binnedplot(x = spotify_aug$speechiness,
                y = spotify_aug$.resid,
                col.int = FALSE,
                xlab = "Speechiness",
                main = "Binned Residual vs. Speechiness")
```



8. Choose a categorical predictor in the final model. Make the appropriate table or plot to examine the residuals versus this predictor variable.

```
spotify_aug %>%
  group_by(key) %>%
  summarise(mean_resid = mean(.resid)) %>%
  kable(format="markdown")
```

key	mean_resid
D	0.0541533
D#	-0.0991901
Other	0.0031569

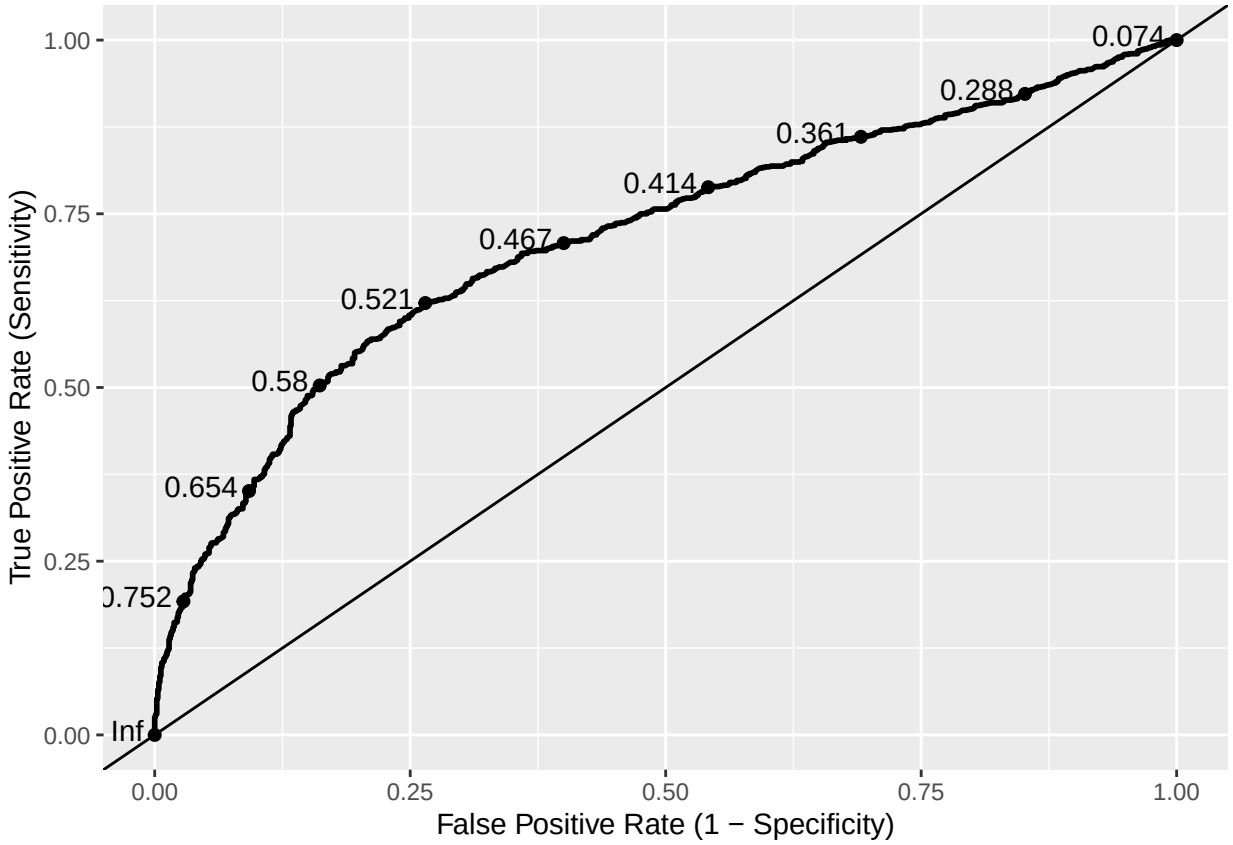
9. Based on the residuals plots from Exercises 6 - 8, is the linearity assumption satisfied? Briefly explain why or why not.

Based on the residual plots from Exercises 6-8, the linearity assumption is not satisfied due to the binned residuals versus predicted values having a V pattern. Although the other graphs and table seem to have no pattern or have residuals far from zero, the exercise 6 plot has a pattern so it ultimately does not pass the assumption. Every variable in the model should be tested to check if there are no patterns in the binned plots.

Part III: Model Assessment & Prediction

10. Plot the ROC curve and calculate the area under the curve (AUC). Display at least 5 thresholds (n.cut = 5) on the ROC.

```
(roc_curve <- ggplot(spotify_aug,
                     aes(d = as.numeric(target) - 1,
                         m = .fitted)) +
  geom_roc(n.cuts = 10, labelround = 3) +
  geom_abline(intercept = 0) +
  labs(x = "False Positive Rate (1 - Specificity)",
       y = "True Positive Rate (Sensitivity)") )
```




```
calc_auc(roc_curve)$AUC
```

```
## [1] 0.7137869
```

11. Based on the ROC curve and AUC in the previous exercise, do you think this model effectively differentiates between the songs the user likes versus those he doesn't?

Based on the ROC curve and AUC in the previous exercise, I would say this model effectively differentiates between the songs the user likes versus those he doesn't to an extent. The ideal model would have an AUC of 1 but, with the AUC value being 0.714 which is greater than 0.5, the model can effectively predict the songs the user likes versus those he doesn't. The ROC curve further support this conclusion since the curve is above the y = x line meaning it can be useful for logistic regression when looking at terms of false and true positive rates.

12. You are part of the data science team at Spotify, and your model will be used to make song recommendations to users. The goal is to recommend songs the user has a high probability of liking.

Choose a threshold value to distinguish between songs the user will like and those the user won't like. What is your threshold value? Use the ROC curve to help justify your choice.

The threshold value I would choose to distinguish between songs the user will like and those the user won't like would be around 0.55. I chose that number since the displayed values on the ROC Curve by n.cuts = 10 have 0.58 and 0.521 being the best values. I went with choosing the value between the two values since both points seem almost the same distance from the top left point of the graph just by glancing.

13. Make the confusion matrix using the threshold chosen in the previous question.

```
threshold <- 0.55

spotify_aug %>%
  mutate(predict = if_else(.fitted > threshold, "1: Yes", "0: No")) %>%
  group_by(target, predict) %>%
  summarise(n = n()) %>%
  kable(format="markdown")
```

```
## `summarise()` has grouped output by 'target'. You can override using the `.groups` argument.
```

target	predict	n
0	0: No	784
0	1: Yes	213
1	0: No	439
1	1: Yes	581

14. Use the confusion matrix from the previous question to answer the following:

- What is the proportion of true positives (sensitivity)? The proportion of true positive also known as sensitivity is the proportion of observations that y = 1 and have a predicted probability above the particular threshold or predicted as 1. This can be calculated by:

$$sensitvity = \frac{pred_{p=1,y=1}}{count_{y=1}} = \frac{581}{581+439} = 0.570$$

- What is the proportion of false positives (1 - specificity)? The proportion of false positives (1 - specificity) is the proportion of observations that y = 0 and have a predicted probability above the particular threshold or predicted as 1. This can be calculated by:

$$1 - specificity = 1 - \frac{pred_{p=0,y=0}}{count_{y=0}} = 1 - \frac{784}{784+213} = 1 - 0.786 = 0.214$$

- What is the misclassification rate? The misclassification rate is the number of observations that were wrongly predicted over the total number of observations. The misclassification rate can be calculated by:

$$miscalculationrate = \frac{pred_{p=1,y=0} + pred_{p=0,y=1}}{count_{observations}} = 1 - \frac{213+439}{784+581+213+439} = 0.323$$