Lab 07: Logistic Regression

due Nov 22nd at 11:59p

Getting Started

Packages

You will need the following packages for today's lab:

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --
## v ggplot2 3.3.6      v purrr   0.3.4
## v tibble  3.1.8      v dplyr   1.0.10
## v tidyr   1.2.1      v stringr 1.4.1
## v readr   2.1.2      v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(stringr)
library(skimr)
library(broom)
library(pROC)
```

```
## Type 'citation("pROC")' for a citation.
##
## Attaching package: 'pROC'
##
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(plotROC)
```

```
##
## Attaching package: 'plotROC'
##
## The following object is masked from 'package:pROC':
##
##     ggroc
```

```
library(knitr)
library(performance)
# Fill in other packages as needed
```

Data

The data in this lab is from the Spotify Song Attributes data set in Kaggle. This data set contains song characteristics of 2017 songs played by a single user and whether or not he liked the song. Since this dataset contains the song preferences of a single user, the scope of the analysis is limited to this particular user.

You will use data `spotify.csv` in the `datasets` folder from our GitHub repo.

The Spotify documentation page contains a description of the variables included in this dataset.

```
df <- read.csv("raw_data/spotify.csv")
head(df)
```

```
##   X acousticness danceability duration_ms energy instrumentalness key liveness
## 1 0      0.01020        0.833      204600  0.434         0.021900   2   0.1650
## 2 1      0.19900        0.743      326933  0.359         0.006110   1   0.1370
## 3 2      0.03440        0.838      185707  0.412         0.000234   2   0.1590
## 4 3      0.60400        0.494      199413  0.338         0.510000   5   0.0922
## 5 4      0.18000        0.678      392893  0.561         0.512000   5   0.4390
## 6 5      0.00479        0.804      251333  0.560         0.000000   8   0.1640
##   loudness mode speechiness   tempo time_signature valence target
## 1   -8.795    1      0.4310 150.062              4   0.286      1
## 2  -10.401    1      0.0794 160.083              4   0.588      1
## 3   -7.148    1      0.2890  75.044              4   0.173      1
## 4  -15.236    1      0.0261  86.468              4   0.230      1
## 5  -11.648    0      0.0694 174.004              4   0.904      1
## 6   -6.682    1      0.1850  85.023              4   0.264      1
##       song_title           artist
## 1       Mask Off           Future
## 2        Redbone Childish Gambino
## 3   Xanny Family           Future
## 4 Master Of None      Beach House
## 5 Parallel Lines      Junior Boys
## 6       Sneakin'            Drake
```

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.

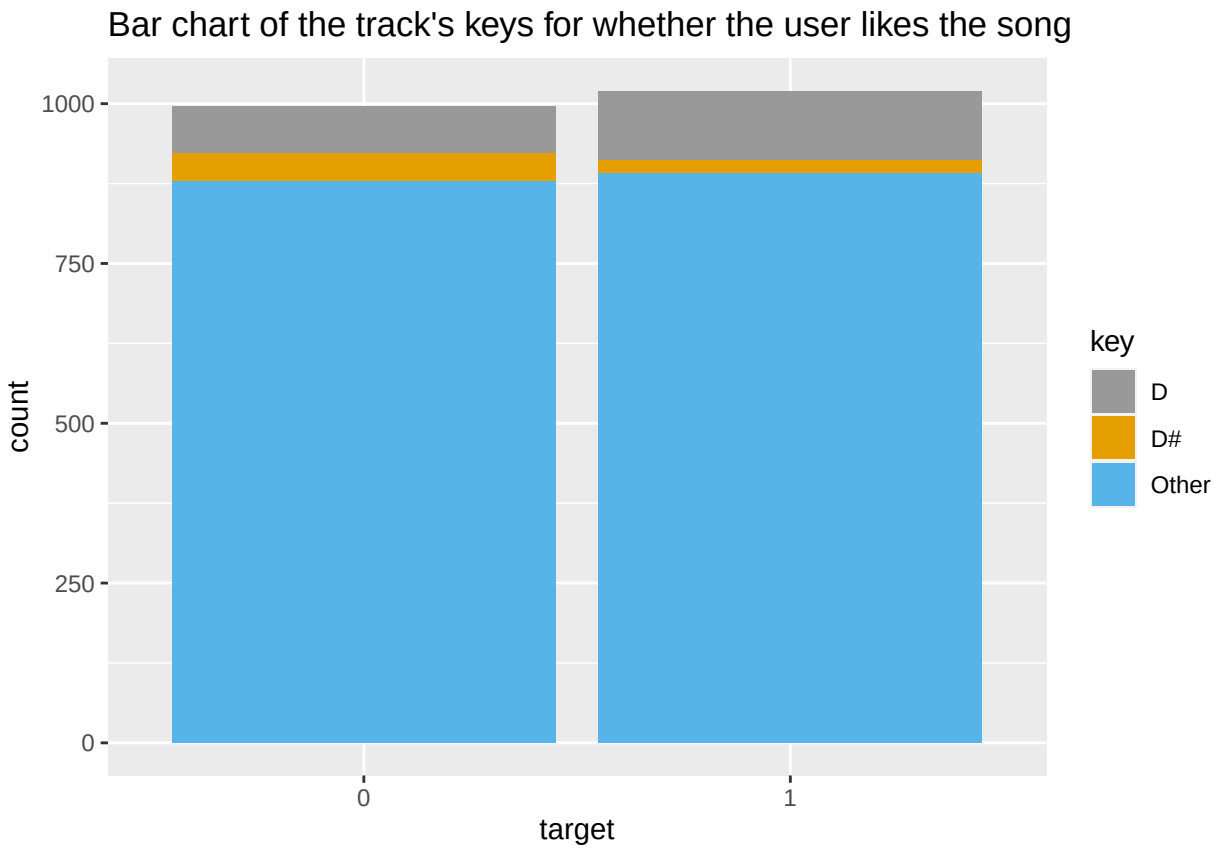
    - If needed, change `target` so that it is factor variable type in R.
    - Change `key` so that it is a factor variable type in R, which takes values "D" if `key==2`, "D#" if `key==3`, and "Other" for all other values.
    - Plot the relationship between `target` and `key`. Briefly describe the relationship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

Answer: For songs that the user likes or not, most of them have other keys and only a tiny portion of them have D# key. More songs that the user doesn't like have D# key, and more songs that the user likes have D key.

```
# change `target` and `key` as factor variable
df$target <- as.factor(df$target)

df$key <- case_when(df$key == 2 ~ "D",
                                df$key == 3 ~ "D#",
                                df$key != 2 & df$key != 3 ~ "Other")
df$key <- as.factor(df$key)

# dotplot between target and key
ggplot(df, aes(x = target, fill = key))  +
  geom_bar(position = "stack") + scale_fill_manual(values=c("#999999", "#E69F00", "#56B4E9")) +
  ggtitle("Bar chart of the track's keys for whether the user likes the song")
```



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.

```
logit_reg <- glm(data = df,
                 target ~ acousticness
                 + danceability
                 + duration_ms
```

```
                  + instrumentalness
                  + loudness
                  + speechiness
                  + valence,
                  family = "binomial")

tidy(logit_reg) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-2.955	0.276	-10.693	0
acousticness	-1.722	0.240	-7.182	0
danceability	1.630	0.344	4.737	0
duration_ms	0.000	0.000	4.225	0
instrumentalness	1.353	0.207	6.549	0
loudness	-0.087	0.017	-5.062	0
speechiness	4.072	0.583	6.985	0
valence	0.856	0.223	3.836	0

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.

Conclusion: Since the drop-in-deviance test shows that the new term is statistically significant and we have lower AIC in the model with `key`, we decided to include `key` in the model.

```
model_full <- glm(data = df,
                  target ~ acousticness
                  + danceability
                  + duration_ms
                  + instrumentalness
                  + loudness
                  + speechiness
                  + valence
                  + key,
                  family = "binomial")

anova(logit_reg, model_full, 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
```

```
glance(logit_reg)$AIC
```

```
## [1] 2534.517
```

```
glance(model_full)$AIC
```

```
## [1] 2525.16
```

Use the model you selected in Exercise 3 for the remainder of the lab.

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.

Answer: If all other variables are constant, the odds of the user likes the song is 65.9% lower if the song have D# key then D key, given an odds ratio of exp(-1.073) = 0.341

The odds of a ride exceeding 20 minutes is 37% higher if you are a female compared with a male, if all other variables are constant, given an odds ratio of exp(0.32) = 1.37.

```
tidy(model_full) %>%
  kable(format = "markdown", digits = 3)
```

term	estimate	std.error	statistic	p.value
(Intercept)	-2.509	0.311	-8.068	0.000
acousticness	-1.702	0.241	-7.065	0.000
danceability	1.649	0.345	4.774	0.000
duration_ms	0.000	0.000	4.187	0.000
instrumentalness	1.383	0.207	6.667	0.000
loudness	-0.087	0.017	-5.018	0.000
speechiness	4.034	0.585	6.896	0.000
valence	0.881	0.224	3.927	0.000
keyD#	-1.073	0.335	-3.204	0.001
keyOther	-0.494	0.169	-2.923	0.003

Part II: Checking Assumptions

In the next few questions, we will do an abbreviated analysis of the residuals.

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

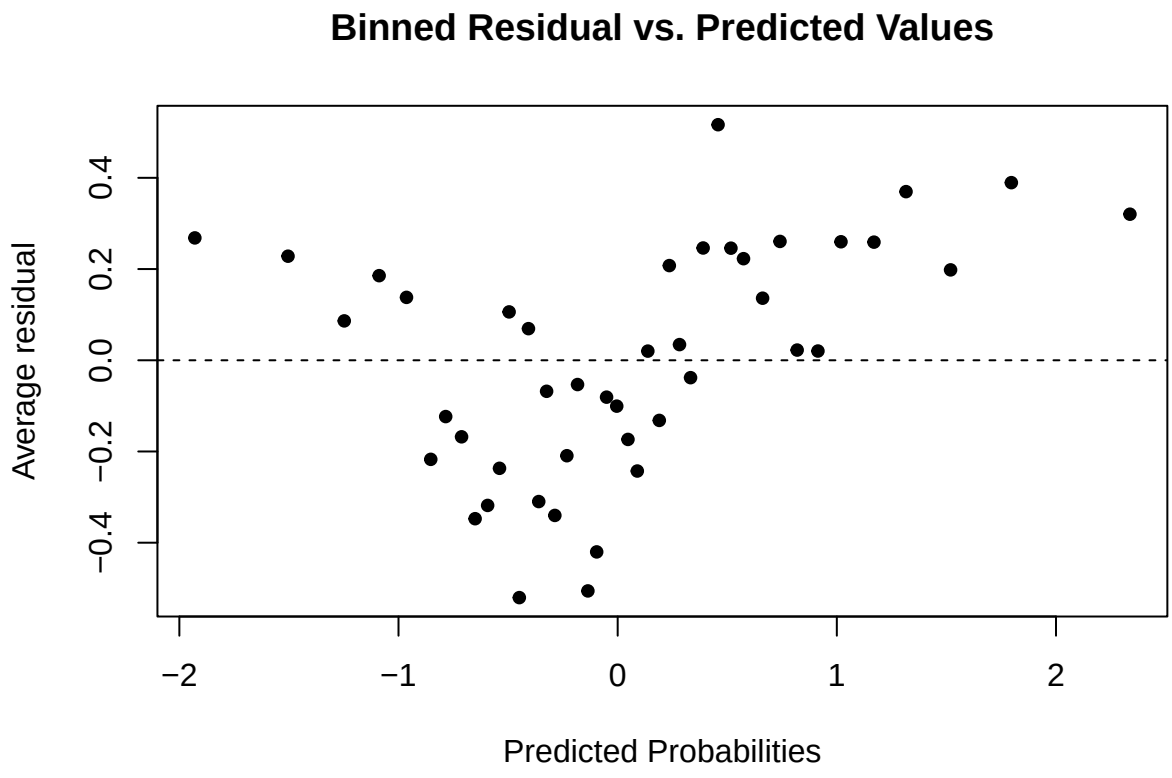
```
(model_aug <- augment(model_full))
```

```
## # A tibble: 2,017 x 15
##    target acoust~1 dance~2 durat~3 instr~4 loudn~5 speec~6 valence key   .fitted
##    <fct>     <dbl>   <dbl>   <int>   <dbl>   <dbl>   <dbl>   <dbl> <fct>   <dbl>
##  1 1       0.0102    0.833  204600 2.19e-2   -8.80  0.431    0.286 D       2.22
##  2 1       0.199     0.743  326933 6.11e-3  -10.4   0.0794   0.588 Other   0.567
##  3 1       0.0344    0.838  185707 2.34e-4   -7.15  0.289    0.173 D       1.28
```

```
##  4 1       0.604     0.494  199413 5.1 e-1  -15.2   0.0261   0.23  Other  -0.313
##  5 1       0.18      0.678  392893 5.12e-1  -11.6   0.0694   0.904 Other   1.73
##  6 1       0.00479   0.804  251333 0         -6.68  0.185    0.264 Other   0.591
##  7 1       0.0145    0.739  241400 7.27e-6  -11.2   0.156    0.308 Other   0.753
##  8 1       0.0202    0.266  349667 6.64e-1  -11.6   0.0371   0.393 Other   0.822
##  9 1       0.0481    0.603  202853 0         -3.63  0.347    0.398 Other   0.554
## 10 1       0.00208   0.836  226840 0         -7.79  0.237    0.386 Other   0.992
## # ... with 2,007 more rows, 5 more variables: .resid <dbl>, .std.resid <dbl>,
## #   .hat <dbl>, .sigma <dbl>, .cooksd <dbl>, and abbreviated variable names
## #   1: acousticness, 2: danceability, 3: duration_ms, 4: instrumentalness,
## #   5: loudness, 6: speechiness
```

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

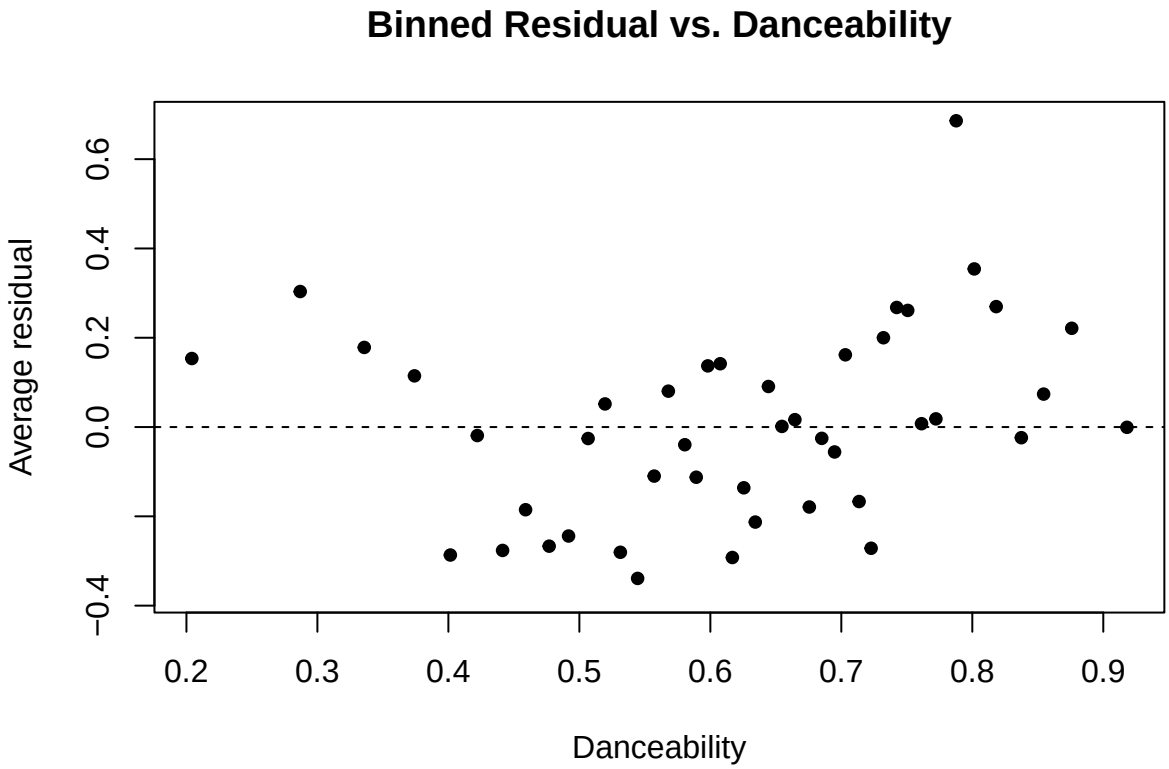
```
arm::binnedplot(x = model_aug$.fitted, y = model_aug$.resid,
                xlab = "Predicted Probabilities",
                main = "Binned Residual vs. Predicted Values",
                col.int = FALSE)
```



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 = model_aug$danceability,
                y = model_aug$.resid,
```

```
                col.int = FALSE,
                xlab = "Danceability",
                main = "Binned Residual vs. Danceability")
```



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

```
model_aug %>%
  group_by(key) %>%
  summarise(mean_resid = mean(.resid))
```

```
## # A tibble: 3 x 2
##   key   mean_resid
##   <fct>      <dbl>
## 1 D        0.0542
## 2 D#      -0.0992
## 3 Other    0.00316
```

In practice, you should examine plots of residuals versus every predictor variable to make a complete assessment of the model fit. For the sake of time on the lab, you will use these three plots to help make the assessment about the model fit.

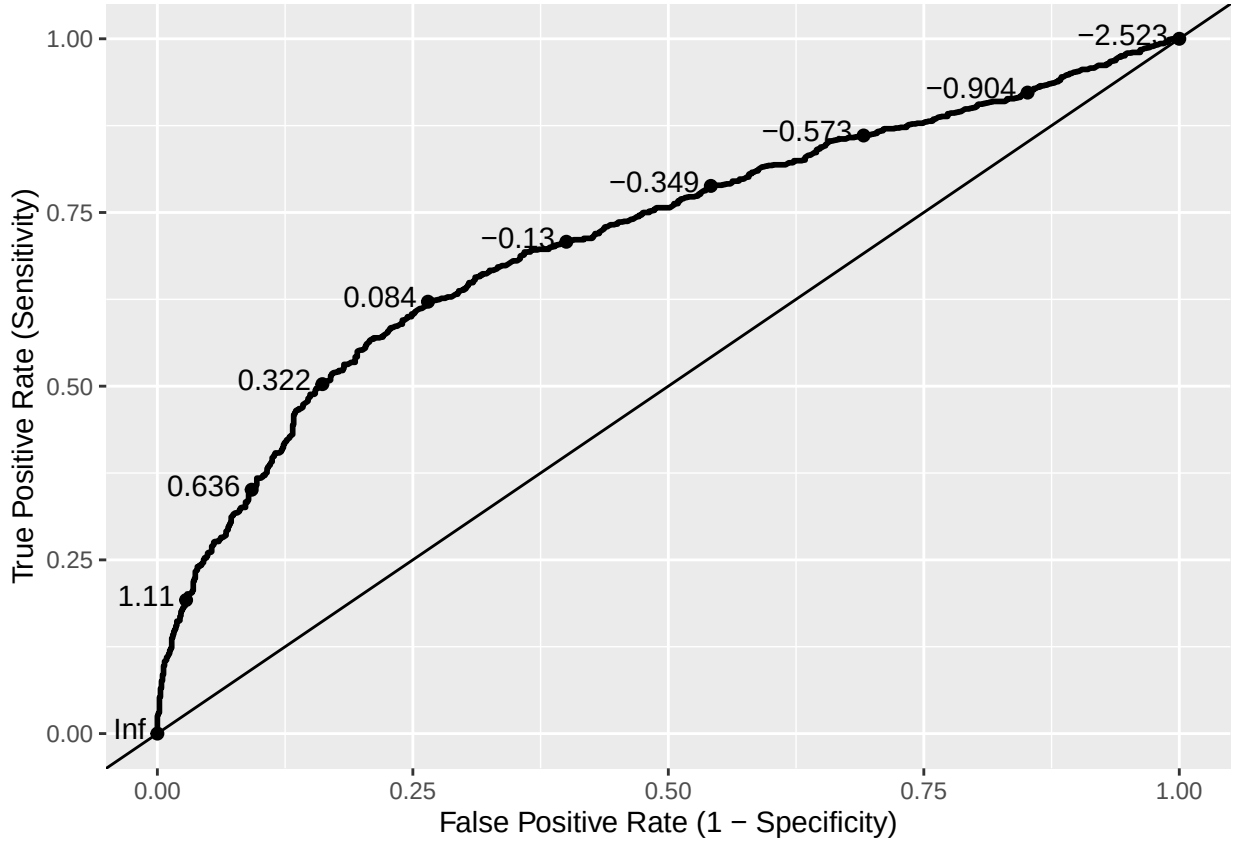
9. Based on the residuals plots from Exercises 6 - 8, is the linearity assumption satisfied? Briefly explain why or why not.

Answer: By the first two residual plots, there are no strong pattern and variance are similar. From the third residual plots, the mean residuals for each type of keys are very small. Therefore the linearity assumption is satisfied.

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(model_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?

Answer: Since the ROC curve is not very close to the top-left corner and relatively not too high AUC, we won't say the model differentiates between the songs the user likes versus those he doesn't very well but it's acceptable.

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.

Answer: my threshold value would be 0.084. From the ROC curve, up to the point of 0.084, true positive rate is much larger than false positive rate. But after 0.084, the rate of increase in true positive rate starts to slow down.

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

```
threshold <- 0.084
model_aug %>%
  mutate(risk_predict = if_else(.fitted > threshold, "Yes", "No")) %>%
  group_by(target, risk_predict) %>%
  summarise(n = n()) %>%
  kable(format="markdown")
```

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

target	risk_predict	n
0	No	731
0	Yes	266
1	No	386
1	Yes	634

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

- What is the proportion of true positives (sensitivity)?
- What is the proportion of false positives (1 - specificity)?
- What is the misclassification rate?

Answer: The proportion of true positive is 0.31. The proportion of false positives is 0.13. The misclassification rate is 0.32.

Submitting the Assignment

As before, what the instructor is going to check is your repo. Make sure to produce a pdf, and include it in your repo with the name Lab07.pdf. Also, include a folder called raw_data where the original data should be stored, and another folder called mod_data where the final version of your data table should be stored. Finally, a Readme.md should be created with a short description of this lab and the data.

```
write.csv(df, "mod_data/mod_spotify.csv")
```

Grading

Part I	20
Part II	20
Document neatly organized with appropriate headers	5
Commit messages and repo	5
Total	**50**