STA303/1002: Mixed assessment 2

Spotify data

Yukun Gao; 1006112564

Part 1: Data manipulation

1. Load your data by putting your ID in `get_my_songs()`.

```
#devtools::install_github("sta303-bolton/sta303mixed2")
library(sta303mixed2)
library(lme4)
library(mgcv)
```

```
get_my_songs(1006112564)
```

2. Add a new variable called `is_kpop` to your `training_data` set. This variable should take the value 1 if the track is from the 'k-pop' genre and 0 if it is from the jazz genre.

```
training_data <- training_data %>% mutate(is_kpop = if_else(genre=="k-pop",1,0))
```

3. Add a new variable called `likely_live` to your `training_data` set. This variable should take the value 1 if the track has a `liveness` score of greater than 0.8.

```
training_data <- training_data %>% mutate(likely_live = if_else(liveness>0.8,1,0))
```

Part 2: Tables and calculations

1. Using the `training_data`, create a table of `genre` by `mode_name`. I.e., have `genre` on the rows and `mode_name` on the columns. You can use `table` or `xtabs`.

```
training.table=table(training_data$genre,training_data$mode_name)
training.table
```

```
##
##         major minor
##   jazz   5187  3021
##   k-pop   804   666
```

2. Using the `training_data`, calculate the probability that a song is in a minor mode, given that it is a K-pop song.

```
666/(666+804)
```

```
## [1] 0.4530612
```

3. Using the `training_data`, calculate the probability that a song chosen at random from your sample is a Jazz song in a major key.

```
training.table/sum(training.table)
```

```
##
##              major      minor
```

```
##   jazz  0.53595784 0.31215127
##   k-pop 0.08307502 0.06881587
```

4. Using the `training_data`, calculate the odds ratio for a song being in a major key given it is K-pop vs Jazz.

```
odds_kpop = 804/666
odds_jazz = 5187/3021
odds_ratio1=odds_kpop/odds_jazz
odds_ratio1
```

```
## [1] 0.7030987
```

5. Using the `training_data`, calculate the odds ratio for a song being Jazz, given that it is likely live vs not.

```
table(training_data$genre, training_data$likely_live)
```

```
##
##            0    1
##   jazz  7986  222
##   k-pop 1404   66
```

```
odds_live=222/66
odds_notlive=7986/1404
odds_ratio2=odds_live/odds_notlive
odds_ratio2
```

```
## [1] 0.591353
```

Part 3: Models

1. Using the `training_data`, fit an appropriate GLM to predict whether or not a song is 'K-pop'. Use `artist_popularity`, `danceability`, `valence` as your predictors.

```
model_1 = glm(is_kpop~artist_popularity+danceability+valence,
              family=binomial(link = "logit"),training_data)
summary(model_1)
```

```
##
## Call:
## glm(formula = is_kpop ~ artist_popularity + danceability + valence,
##     family = binomial(link = "logit"), data = training_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.8892  -0.2604  -0.1306  -0.0541   3.7944
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -32.03033    0.88649 -36.132  < 2e-16 ***
## artist_popularity   0.41246    0.01207  34.162  < 2e-16 ***
## danceability        8.87262    0.51605  17.193  < 2e-16 ***
## valence            -1.96964    0.27031  -7.287 3.18e-13 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
```

```
##
##     Null deviance: 8245.2  on 9677  degrees of freedom
## Residual deviance: 2411.4  on 9674  degrees of freedom
## AIC: 2419.4
##
## Number of Fisher Scoring iterations: 7
```

2. Using the `training_data`, fit an appropriate **mixed effects model** (use the `lme4` package) to predict whether or not a song is K-pop based on `artist_popularity`, `danceability`, `valence` with an appropriate random effect for `artist_name`.

```
model_2=glmer(is_kpop~artist_popularity+danceability+
                valence+(1|artist_name),family=binomial(link = "logit"),training_data,nAGQ=0)

summary(model_2)
```

```
## Generalized linear mixed model fit by maximum likelihood (Adaptive
##   Gauss-Hermite Quadrature, nAGQ = 0) [glmerMod]
##  Family: binomial  ( logit )
## Formula:
## is_kpop ~ artist_popularity + danceability + valence + (1 | artist_name)
##    Data: training_data
##
##      AIC      BIC   logLik deviance df.resid
##     50.4     86.3    -20.2     40.4     9673
##
## Scaled residuals:
##       Min        1Q    Median        3Q       Max
## -0.016754 -0.002763 -0.002112 -0.001554  0.045446
##
## Random effects:
##  Groups      Name        Variance Std.Dev.
##  artist_name (Intercept) 850.7    29.17
## Number of obs: 9678, groups:  artist_name, 21
##
## Fixed effects:
##                   Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -65.2735    86.6984  -0.753    0.452
## artist_popularity   1.0146     1.4135   0.718    0.473
## danceability        4.7981    20.5608   0.233    0.815
## valence            -0.9898    13.0171  -0.076    0.939
##
## Correlation of Fixed Effects:
##             (Intr) artst_ dncblt
## artst_pplrt -0.989
## danceabilty -0.124  0.022
## valence     -0.039  0.035 -0.502
```

3. Using the `training_data`, fit an appropriate **GAM** (use the `mgcv` package) to predict whether or not a song is K-pop based on `artist_popularity`, `danceability`, `valence` with a random effect for `artist_name`.

```
model_3 <- gam(is_kpop ~ s(artist_popularity)+ s(danceability)+
                 s(valence)+ s(artist_name,bs="re"),
               family = binomial(link="logit"), data =  training_data)
summary(model_3)
```

```
##
## Family: binomial
## Link function: logit
##
## Formula:
## is_kpop ~ s(artist_popularity) + s(danceability) + s(valence) +
##     s(artist_name, bs = "re")
##
## Parametric coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.342      6.245  -0.215     0.83
##
## Approximate significance of smooth terms:
##                         edf Ref.df Chi.sq p-value
## s(artist_popularity)  1.00  1.001  0.684   0.408
## s(danceability)       1.00  1.000  0.074   0.786
## s(valence)            1.00  1.000  0.008   0.929
## s(artist_name)       14.64 19.000 20.759   0.114
##
## R-sq.(adj) =      1   Deviance explained =  100%
## UBRE = -0.9961  Scale est. = 1         n = 9678
```

```
coef(model_3)
```

```
##            (Intercept) s(artist_popularity).1 s(artist_popularity).2
##          -1.341524e+00          -3.638410e-04           5.331591e-04
## s(artist_popularity).3 s(artist_popularity).4 s(artist_popularity).5
##           2.200836e-04          -1.524084e-04           1.087808e-04
## s(artist_popularity).6 s(artist_popularity).7 s(artist_popularity).8
##           1.233740e-04          -1.245754e-04          -4.858806e-04
## s(artist_popularity).9      s(danceability).1      s(danceability).2
##           9.286379e+00          -3.392828e-06          -8.281135e-07
##      s(danceability).3      s(danceability).4      s(danceability).5
##          -4.757465e-07           1.875190e-06           1.665413e-08
##      s(danceability).6      s(danceability).7      s(danceability).8
##          -1.996300e-06           1.937691e-07          -9.490236e-06
##      s(danceability).9           s(valence).1           s(valence).2
##           7.209487e-01           7.527222e-07          -1.305529e-06
##           s(valence).3           s(valence).4           s(valence).5
##           1.017943e-06           1.415891e-06          -1.091554e-06
##           s(valence).6           s(valence).7           s(valence).8
##           1.490519e-06          -1.127813e-06           8.008519e-06
##           s(valence).9       s(artist_name).1       s(artist_name).2
##          -2.466678e-01          -2.145473e+00          -6.862415e+00
##       s(artist_name).3       s(artist_name).4       s(artist_name).5
##           6.268192e-11          -5.812281e+00           9.498087e+00
##       s(artist_name).6       s(artist_name).7       s(artist_name).8
##          -1.000383e+01           4.356855e-01           7.015672e+00
##       s(artist_name).9      s(artist_name).10      s(artist_name).11
##          -6.422267e+00          -9.195516e+00           1.262858e+01
##      s(artist_name).12      s(artist_name).13      s(artist_name).14
##          -4.665022e+00           4.011419e+00           8.475304e-01
##      s(artist_name).15      s(artist_name).16      s(artist_name).17
##          -2.972799e+00           4.458862e+00          -6.499608e+00
##      s(artist_name).18      s(artist_name).19      s(artist_name).20
```

```
##          -1.368191e+01           1.514761e+01           6.488375e+00
##      s(artist_name).21
##           7.729307e+00
```

Part 4: Checking your predictions

```
add_my_predictions(testing_data, model_1, model_2, model_3)
```

```
## Warning in predict.gam(model_3, newdata = testing_data, type = "response"):
## factor levels 2NE1, 2PM, 4Minute, AKMU, Al Jarreau, B.A.P, BIGBANG, Bill
## Evans, Billie Holiday, Brenda Lee, Cafe Jazz Deluxe, Cap Kendricks, Chet Baker,
## Chick Corea, Coffee Shop Jazz Relax, Coleman Hawkins, Count Basie, DAY6, Dizzy
## Gillespie, Doris Day, Duke Ellington, Ella Fitzgerald, Epik High, Etta James,
## EXO, GFRIEND, Groove Armada, Harry James, INFINITE, IU, Jay Park, Kid Koala,
## Kim Bum Soo, Lester Young, Louis Armstrong, Luiz Bonfá, Mel Tormé, Miles Davis,
## N.Flying, Nancy Wilson, NCT 127, Nightmares On Wax, Norah Jones, NU'EST, OH
## MY GIRL, Oscar Peterson, Pat Metheny, Peggy Lee, Sarah Vaughan, Sérgio Mendes,
## SEVENTEEN, SF9, Sik-K, Smooth Dinner Jazz, Stan Getz, Standing Egg, Stray Kids,
## suggi, Sung Si Kyung, SUPER JUNIOR, T-ARA, TAEMIN, TAEYEON, The Andrews Sisters,
## TVXQ!, VICTON, WINNER, Wynton Marsalis, Yoon Mirae not in original fit
```

2. Find out how many correct predictions each of these models make when applied to the `testing_data`. The testing data is for 100 tracks by artists not in your `training data`.

```
tab_1 <- xtabs(~is_kpop+model_1, data = testing_data)
sum(diag(tab_1))/100
```

```
## [1] 0.56
```

```
tab_2 <- xtabs(~is_kpop+model_2, data = testing_data)
sum(diag(tab_2))/100
```

```
## [1] 0.55
```

```
tab_3 <- xtabs(~is_kpop+model_3, data = testing_data)
sum(diag(tab_3))/100
```

```
## [1] 0.55
```