

Regression

Code ▾

1. In linear regression, we wish to find the relationship between predictor values and a target value in the form of a linear relationship. This linear relationship can be defined by parameters w and b , with w , the slope of the line, quantifying the amount that y changes with changes in x , and b serving as an intercept. Linear regression is a relatively simple and powerful model. However, this algorithm has a bias: it wants to see a line no matter what the data is (even if a line should not exist). Some other disadvantages could be interaction effects, confounding variables, and hidden variables.
2. This notebook explores song data from Kaggle (<https://www.kaggle.com/datasets/yasserh/song-popularity-dataset>)

Load the `song_data.csv` file.

Hide

```
df <- read.csv("song_data.csv")
str(df)
```

```
'data.frame': 18835 obs. of 15 variables:
 $ song_name      : chr  "Boulevard of Broken Dreams" "In The End" "Seven Nation Army" "By The
Way" ...
 $ song_popularity : int   73 66 76 74 56 80 81 76 80 81 ...
 $ song_duration_ms: int  262333 216933 231733 216933 223826 235893 199893 213800 222586 203346
...
 $ acousticness   : num   0.00552 0.0103 0.00817 0.0264 0.000954 0.00895 0.000504 0.00148 0.0010
8 0.00172 ...
 $ danceability    : num   0.496 0.542 0.737 0.451 0.447 0.316 0.581 0.613 0.33 0.542 ...
 $ energy          : num   0.682 0.853 0.463 0.97 0.766 0.945 0.887 0.953 0.936 0.905 ...
 $ instrumentalness: num   2.94e-05 0.00 4.47e-01 3.55e-03 0.00 1.85e-06 1.11e-03 5.82e-04 0.00
1.04e-02 ...
 $ key             : int    8 3 0 0 10 4 4 2 1 9 ...
 $ liveness        : num   0.0589 0.108 0.255 0.102 0.113 0.396 0.268 0.152 0.0926 0.136 ...
 $ loudness        : num   -4.09 -6.41 -7.83 -4.94 -5.07 ...
 $ audio_mode      : int    1 0 1 1 1 0 0 1 1 1 ...
 $ speechiness     : num   0.0294 0.0498 0.0792 0.107 0.0313 0.124 0.0624 0.0855 0.0917 0.054 ...
 $ tempo           : num   167 105 124 122 172 ...
 $ time_signature  : int    4 4 4 4 4 4 4 4 4 4 ...
 $ audio_valence   : num   0.474 0.37 0.324 0.198 0.574 0.32 0.724 0.537 0.234 0.374 ...
```

a. Divide `df` into 80/20 train/test

Hide

```
set.seed(1234)
i <- sample(1:nrow(df), nrow(df)*0.8, replace = FALSE)
train <- df[i,]
test <- df[-i,]
```

b. Data Exploration

c. List the column names

song_popularity is the target variable and the rest are predictors.

Hide

```
names(train)
```

```
[1] "song_name"      "song_popularity" "song_duration_ms" "acousticness"
[5] "danceability"   "energy"          "instrumentalness" "key"
[9] "liveness"       "loudness"        "audio_mode"      "speechiness"
[13] "tempo"          "time_signature"  "audio_valence"
```

ii. See the first 5 rows

Hide

```
head(train, n=5)
```

song_name <chr>	song_popularity <int>	song_duration_ms <int>	acousticness
7452 Never Recover (Lil Baby & Gunna, Drake)	89	194732	
8016 Come Through	58	265160	
7162 Superhero	82	179997	
8086 Macarena	62	249506	
9196 Passing Me By	13	301840	

5 rows | 1-5 of 15 columns

iii. Check the number of NAs in each column.

Hide

```
sapply(train, function(x) sum(is.na(x)))
```

```
song_name    song_popularity song_duration_ms    acousticness    danceability
      0              0              0              0              0
energy instrumentalness    key    liveness    loudness
      0              0              0              0              0
audio_mode    speechiness    tempo    time_signature    audio_valence
      0              0              0              0              0
```

There are no NAs in this dataset. This means we do not have to delete/modify any rows.

iv. Converting columns into factors.

Hide

```
#str(df)
train$key = factor(train$key)
train$audio_mode = factor(train$audio_mode)
train$time_signature = factor(train$time_signature)
str(train)
```

```
'data.frame': 15068 obs. of 15 variables:
 $ song_name      : chr  "Never Recover (Lil Baby & Gunna, Drake)" "Come Through" "Superhero"
"Macarena" ...
 $ song_popularity : int   89 58 82 62 13 67 60 65 70 55 ...
 $ song_duration_ms: int  194732 265160 179997 249506 301840 257440 578040 295480 290706 205786
...
 $ acousticness    : num   0.0716 0.18 0.82 0.422 0.0762 0.42 0.125 0.17 0.167 0.684 ...
 $ danceability    : num   0.757 0.489 0.681 0.927 0.763 0.495 0.729 0.48 0.454 0.939 ...
 $ energy          : num   0.69 0.829 0.34 0.721 0.76 0.609 0.72 0.529 0.229 0.455 ...
 $ instrumentalness: num   0.00 0.00 0.00 4.36e-05 0.00 0.00 3.53e-01 4.27e-06 1.02e-01 6.63e-03
...
 $ key            : Factor w/ 12 levels "0","1","2","3",...: 2 10 8 9 5 8 4 7 3 12 ...
 $ liveness       : num   0.192 0.538 0.285 0.0422 0.26 0.187 0.178 0.257 0.202 0.0903 ...
 $ loudness       : num  -5.33 -8.07 -8.62 -12.56 -7.97 ...
 $ audio_mode     : Factor w/ 2 levels "0","1": 1 1 2 2 1 2 2 2 2 1 ...
 $ speechiness    : num   0.282 0.478 0.0339 0.0908 0.259 0.0339 0.0371 0.0275 0.0297 0.195 ...
 $ tempo          : num   132.1 139.6 130 103 87.1 ...
 $ time_signature : Factor w/ 5 levels "0","1","3","4",...: 4 4 4 4 4 4 4 4 4 5 ...
 $ audio_valence  : num   0.914 0.694 0.417 0.965 0.599 0.33 0.0388 0.469 0.134 0.458 ...
```

Values for the key column range from 0-11 (inclusive) which can be factorized. Similarly, audio mode (2 possible values) and time signature (5 possible values) columns can also be turned into factors from integers.

v. Checking distribution of levels in key, time signature, and audio mode columns.

[Hide](#)

```
summary(train$key)
```

```
 0    1    2    3    4    5    6    7    8    9   10   11
1730 1715 1361 397 1047 1270 1091 1637 1102 1362 1087 1269
```

[Hide](#)

```
summary(train$time_signature)
```

```
 0    1    3    4    5
 3    65   589 14241  170
```

[Hide](#)

```
summary(train$audio_mode)
```

0	1
5600	9468

The key levels are mostly balanced, with “3” being the least key value. The time signature levels are really skewed towards the “4” value. It is more common to have the audio mode of 1 than 0.

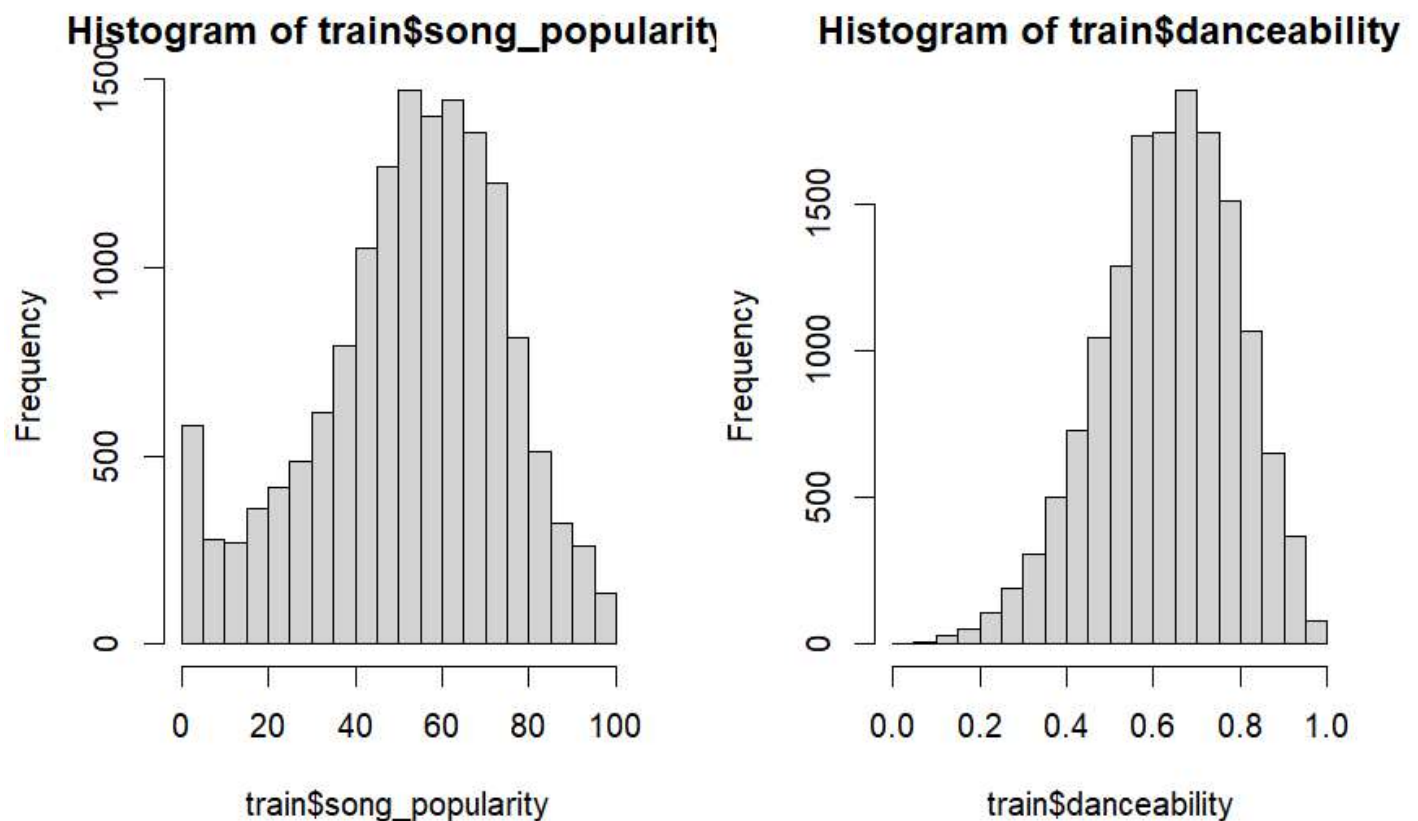
- c. Creating informative graphs using training data.
- d. Explore distribution of song_popularity and danceability.

Hide

```
opar <- par()
par(mfrow=c(1,2)) # plots show up side by side
hist(train$song_popularity)
hist(train$danceability)
```

Hide

```
par(opar)
```

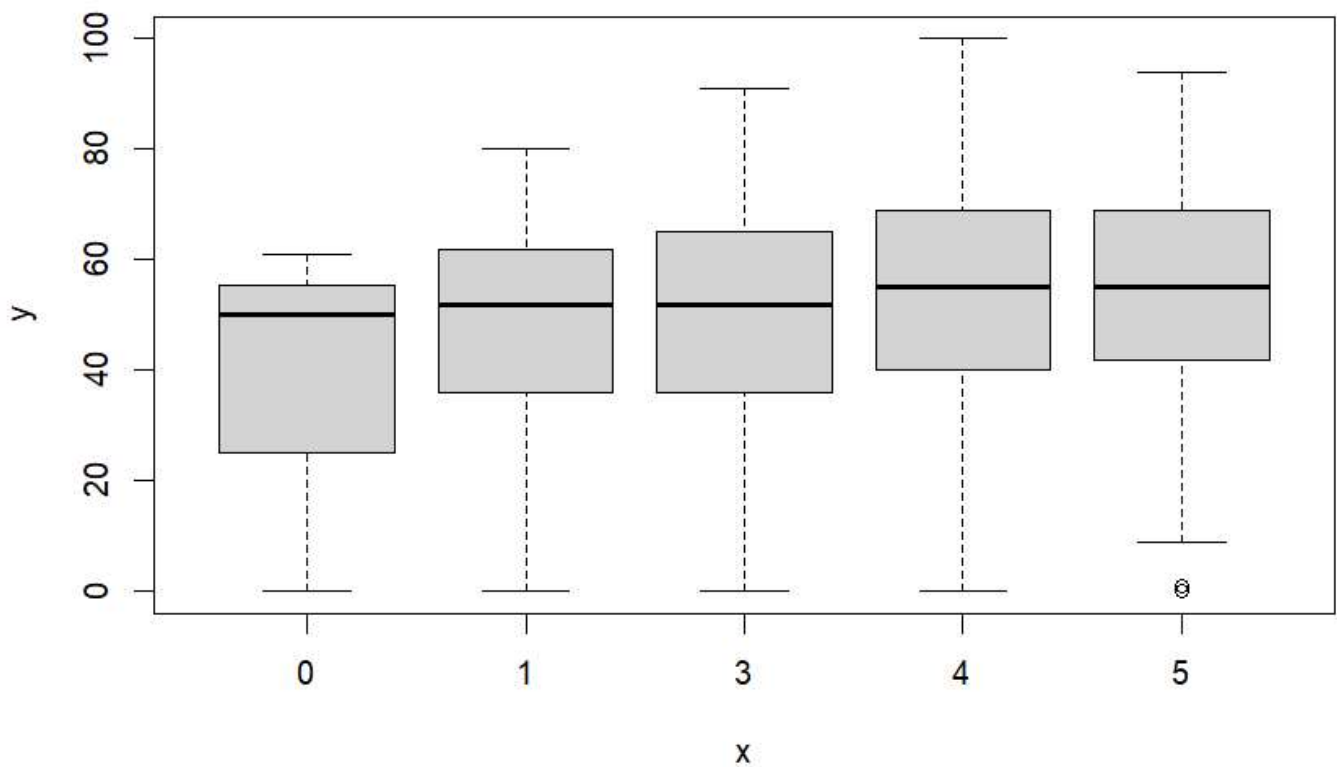


Song popularity and danceability are both slightly skewed to the right. This might mean that danceable songs are more popular.

- ii. Comparing song popularity with time signature.

Hide

```
plot(train$time_signature,train$song_popularity)
```



The average popularity of songs with different time signatures is similar but the distribution varies. Songs with time signature of 1 have lower popularity, and songs with time signature of 5 have higher popularity.

d. Simple linear regression model

[Hide](#)

```
lm1 <- lm(song_popularity~danceability, data=train)
summary(lm1)
```

Call:

```
lm(formula = song_popularity ~ danceability, data = train)
```

Residuals:

Min	1Q	Median	3Q	Max
-57.938	-12.535	2.693	15.817	46.313

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	43.3678	0.7392	58.67	<2e-16 ***
danceability	15.0206	1.1324	13.26	<2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21.77 on 15066 degrees of freedom

Multiple R-squared: 0.01154, Adjusted R-squared: 0.01148

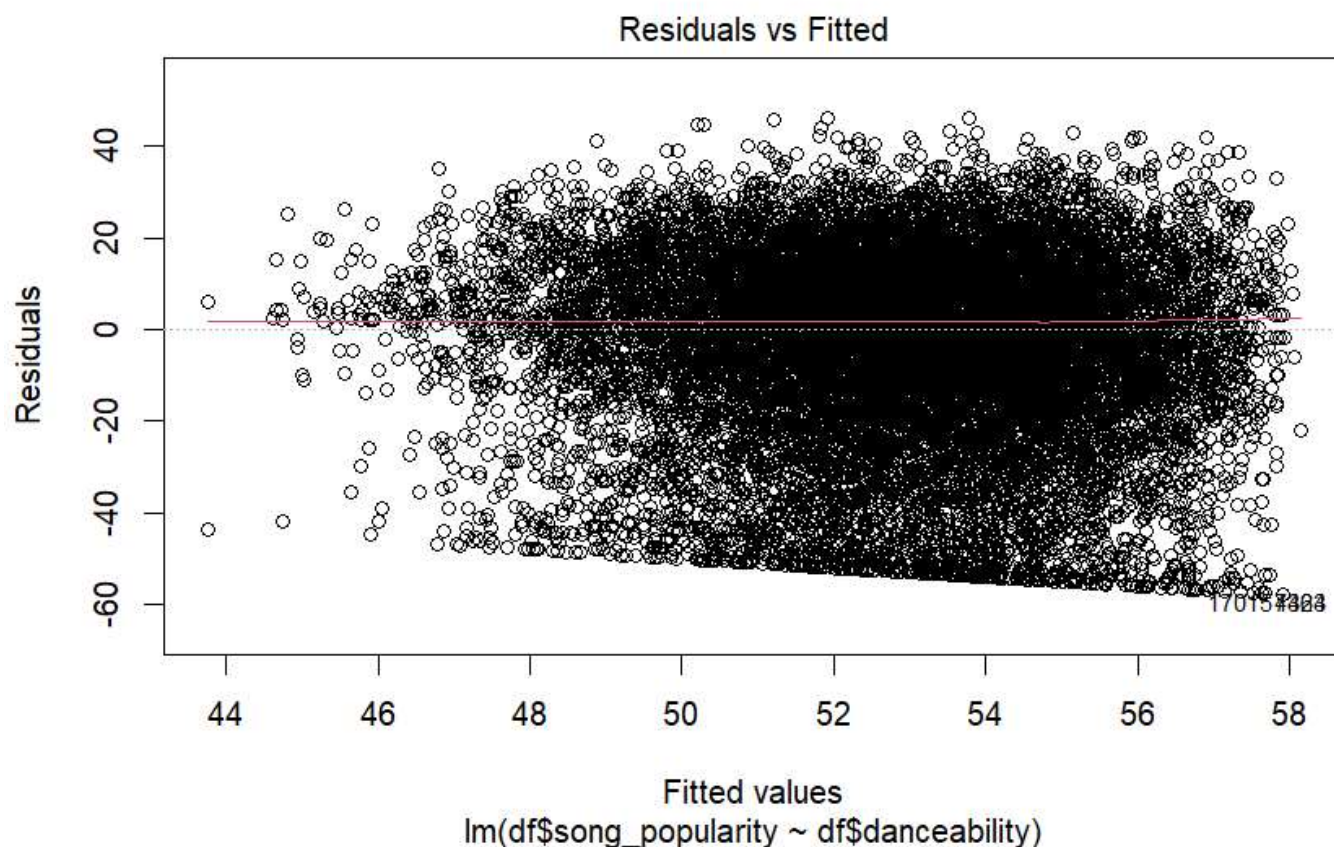
F-statistic: 175.9 on 1 and 15066 DF, p-value: < 2.2e-16

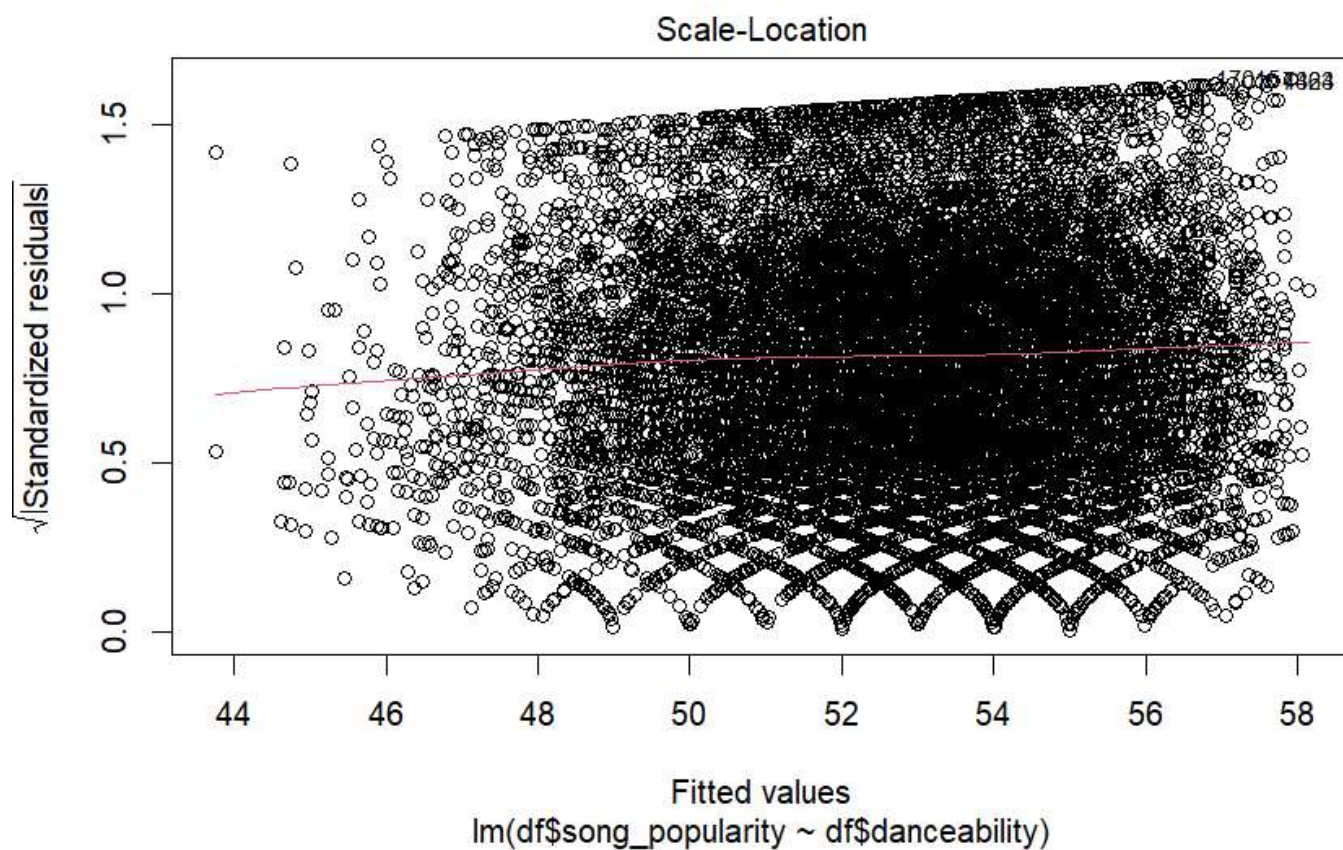
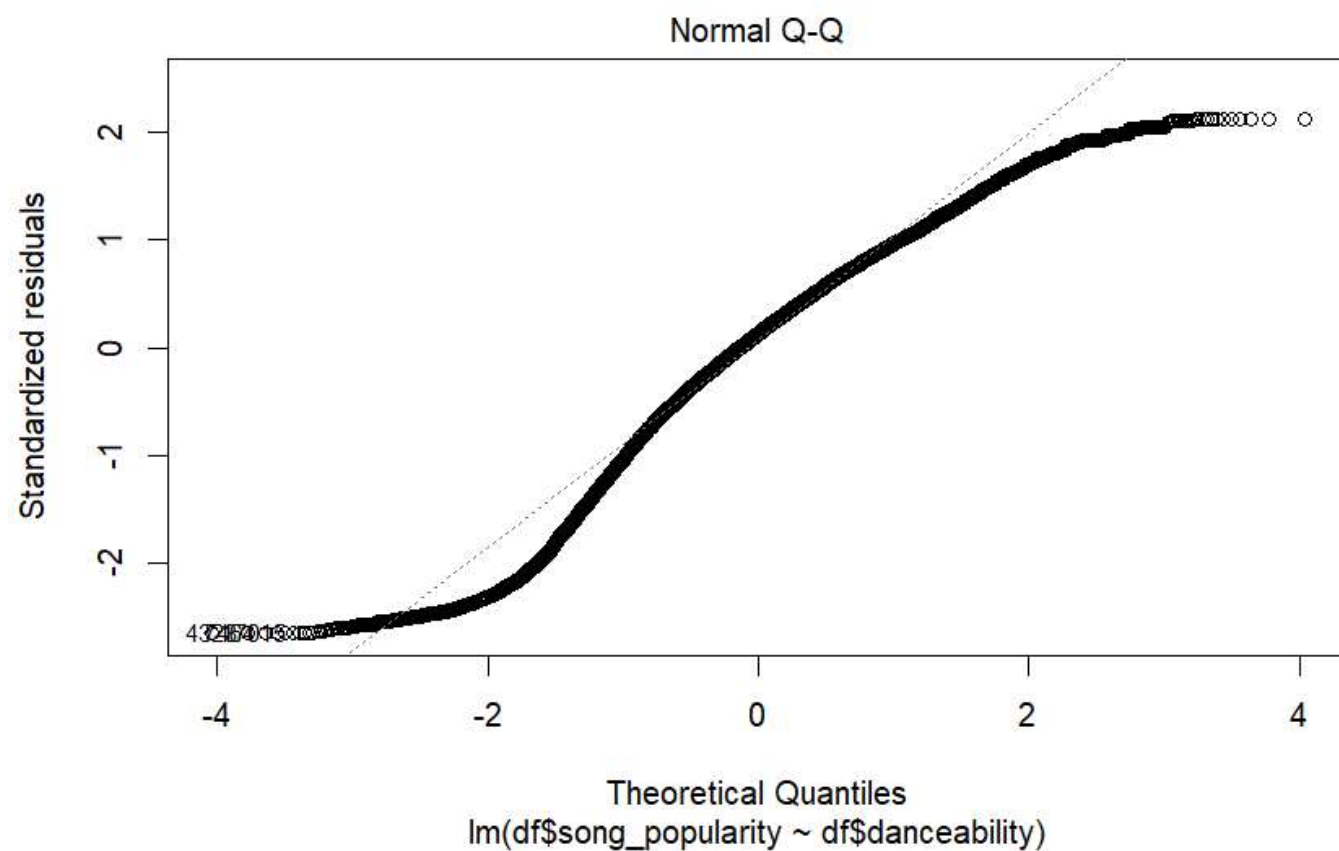
The residuals are not small, which is expected for a simple regression model. At most, there is a difference of about 58 from the predicted song popularity score and the actual score, which is large considering the range of scores (0-100). The p-value is low (which means the high confidence) for danceability, which is good after comparing it to the significant codes section. [p-value is the probability of observing a larger t-statistic if the null hypothesis is true.] This means that it is a great predictor for the popularity score, and other predictors can be used to improve the model. From the coefficients, the linear equation derived from the model can be equated to be song popularity = $15.02 * \text{danceability} + 43.37$. Residual standard error (RSE), which measures the lack of fit of the model, is 21.79 in this case. This is not a small value, which is further proved in the R-squared value of 0.01. [R-squared should be as close to 1 as possible.] At least the overall p-value is still low. The F-statistic provides evidence against the null hypothesis, and is not low in this model.

e. Plotting residuals

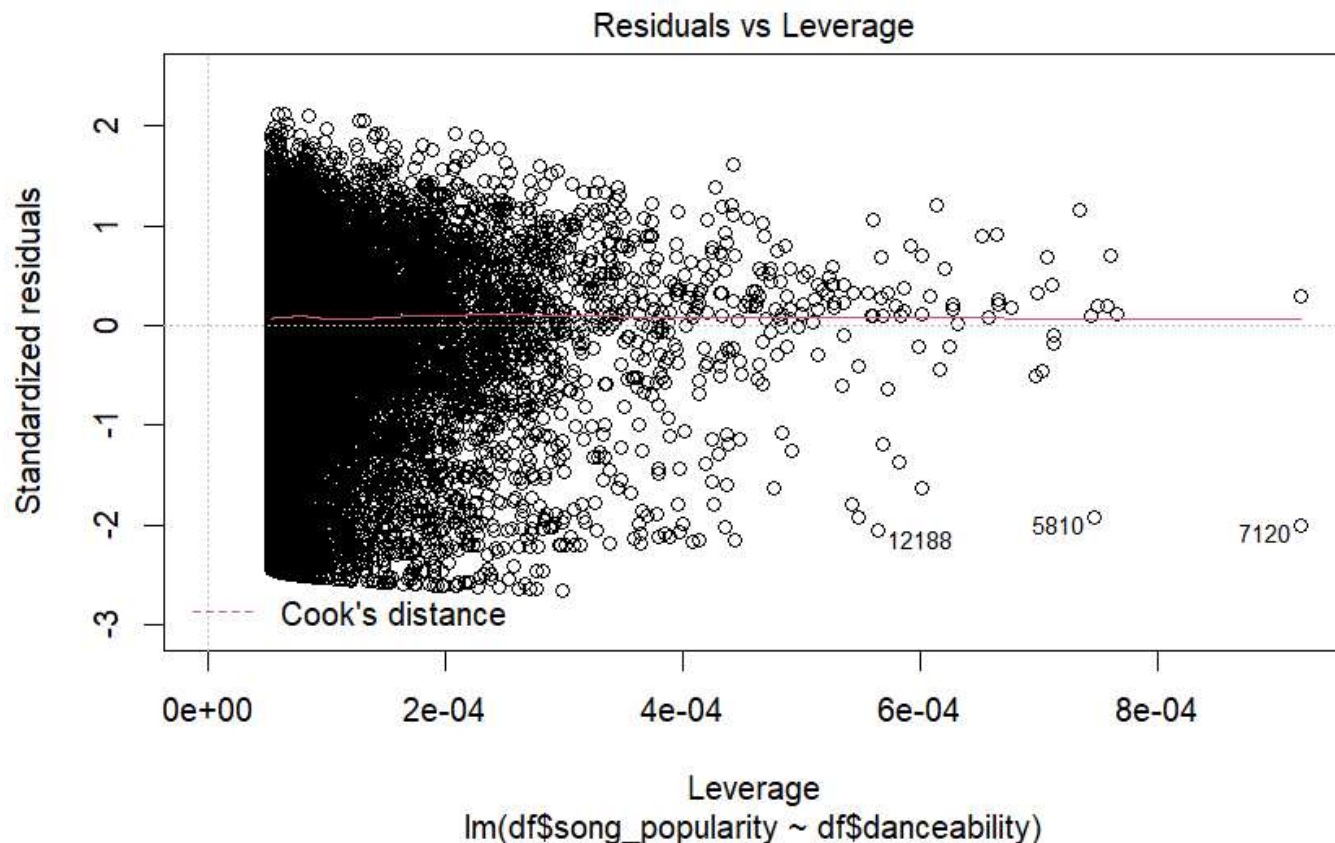
Hide

```
plot(lm1)
```




[Hide](#)

```
par(mfrow=c(1,1))
```

There are 4 different graphs as output:

- i. Residuals vs fitted- shows if residuals have non-linear patterns. (There could be a non-linear relationship between predictor variables and an outcome variable and the pattern could show up in this plot if the model doesn't capture the non-linear relationship. If you find equally spread residuals around a horizontal line without distinct patterns, that is a good indication you don't have non-linear relationships.) There is no distinctive pattern which is good.
- ii. Normal Q-Q- shows if residuals are normally distributed. It's good if residuals are lined well on the straight dashed line, which is the case here. Of course it's not a perfect straight line.
- iii. Scale-location- shows if residuals are spread equally along the ranges of predictors. (This is how you can check the assumption of equal variance (homoscedasticity). It's good if you see a horizontal line with equally (randomly) spread points.) The residuals appear to be scattered everywhere which is good.
- iv. Residuals vs Leverage- helps us to find influential cases (i.e., subjects) if any. (We watch out for outlying values at the upper right corner or at the lower right corner. Those spots are the places where cases can be influential against a regression line.) Cook's line is not visible in this case (because all cases are well inside Cook's line) which means there are no cases that can be influential against a regression line.
- f. Multiple linear regression model (using all predictors except song_name)

Hide

```
#str(df)
lm2 <- lm(song_popularity~danceability+song_duration_ms+acousticness+energy+instrumentalness+key
+liveness+audio_mode+speechiness+tempo+time_signature+audio_valence, data=train)
summary(lm2)
```


Call:

```
lm(formula = song_popularity ~ danceability + song_duration_ms +
    acousticness + energy + instrumentalness + key + liveness +
    audio_mode + speechiness + tempo + time_signature + audio_valence,
    data = train)
```

Residuals:

Min	1Q	Median	3Q	Max
-65.625	-12.360	2.804	15.656	45.947

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	4.681e+01	1.243e+01	3.766	0.000167	***
danceability	1.518e+01	1.316e+00	11.534	< 2e-16	***
song_duration_ms	-8.635e-06	3.083e-06	-2.801	0.005099	**
acousticness	-4.156e+00	8.518e-01	-4.878	1.08e-06	***
energy	-2.743e+00	1.203e+00	-2.281	0.022588	*
instrumentalness	-1.386e+01	8.119e-01	-17.075	< 2e-16	***
key1	1.393e+00	7.365e-01	1.891	0.058641	.
key2	-1.535e+00	7.766e-01	-1.977	0.048111	*
key3	-1.089e+00	1.198e+00	-0.910	0.363004	
key4	-1.298e+00	8.480e-01	-1.531	0.125823	
key5	-5.417e-01	7.974e-01	-0.679	0.496904	
key6	1.471e+00	8.375e-01	1.756	0.079078	.
key7	-2.571e+00	7.389e-01	-3.480	0.000503	***
key8	-1.424e+00	8.283e-01	-1.719	0.085637	.
key9	-1.916e+00	7.803e-01	-2.456	0.014060	*
key10	1.349e-01	8.407e-01	0.160	0.872494	
key11	4.238e-01	8.050e-01	0.526	0.598608	
liveness	-4.708e+00	1.245e+00	-3.783	0.000156	***
audio_mode1	5.584e-01	3.818e-01	1.463	0.143610	
speechiness	-3.885e+00	1.790e+00	-2.170	0.030014	*
tempo	-1.037e-02	6.255e-03	-1.658	0.097309	.
time_signature1	7.874e+00	1.266e+01	0.622	0.534013	
time_signature3	9.051e+00	1.242e+01	0.729	0.466141	
time_signature4	1.016e+01	1.239e+01	0.820	0.412470	
time_signature5	1.197e+01	1.249e+01	0.958	0.337912	
audio_valence	-9.747e+00	8.268e-01	-11.790	< 2e-16	***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21.4 on 15042 degrees of freedom

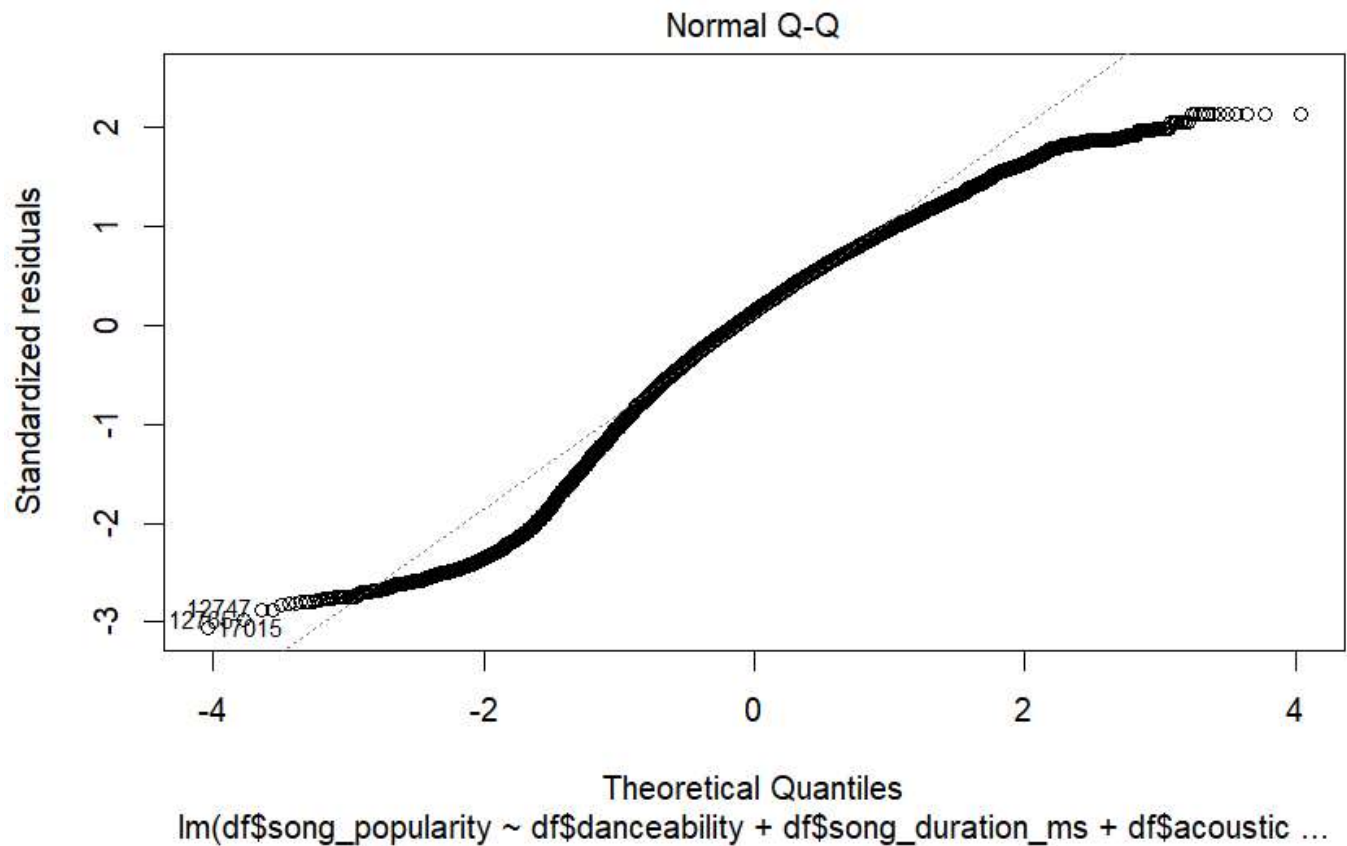
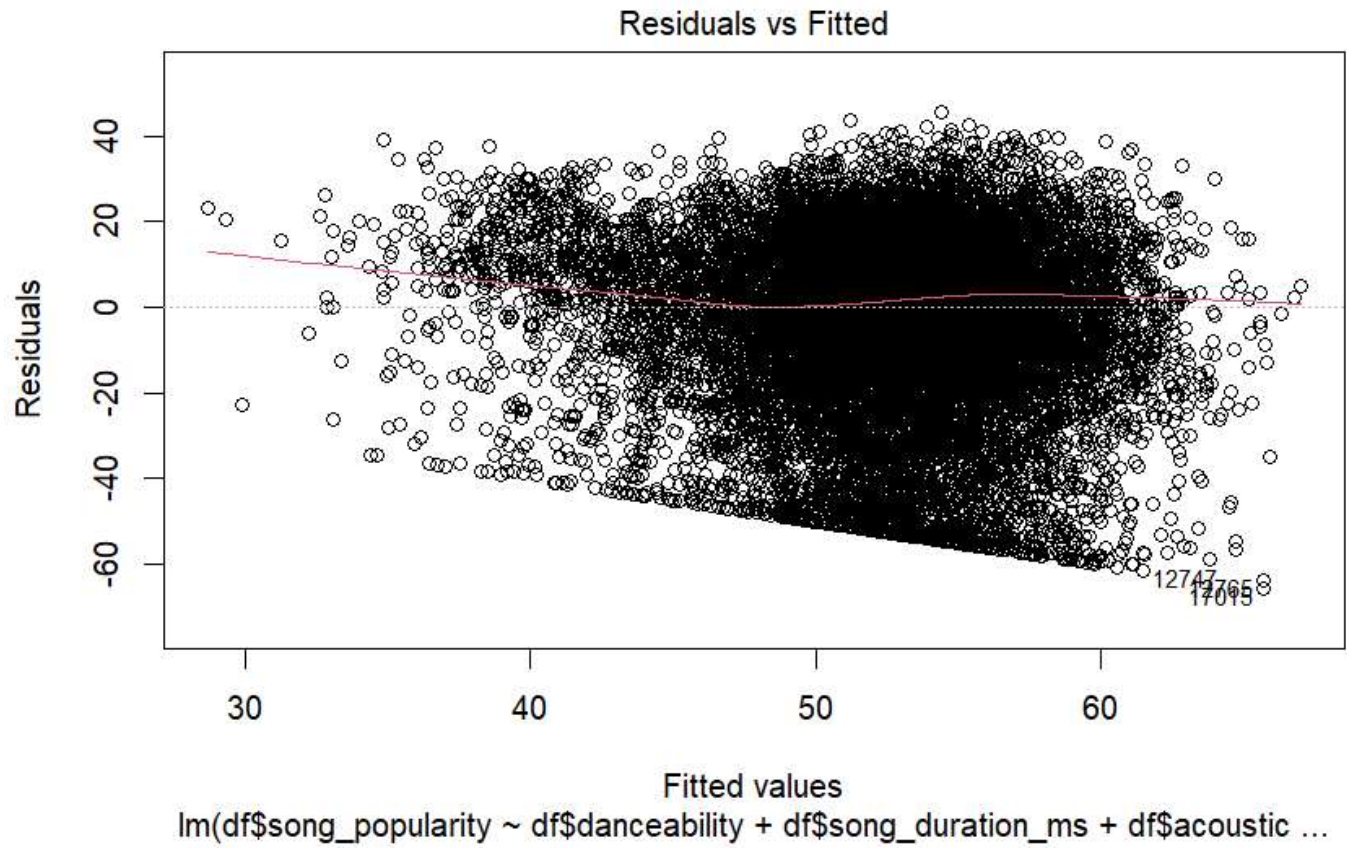
Multiple R-squared: 0.0462, Adjusted R-squared: 0.04461

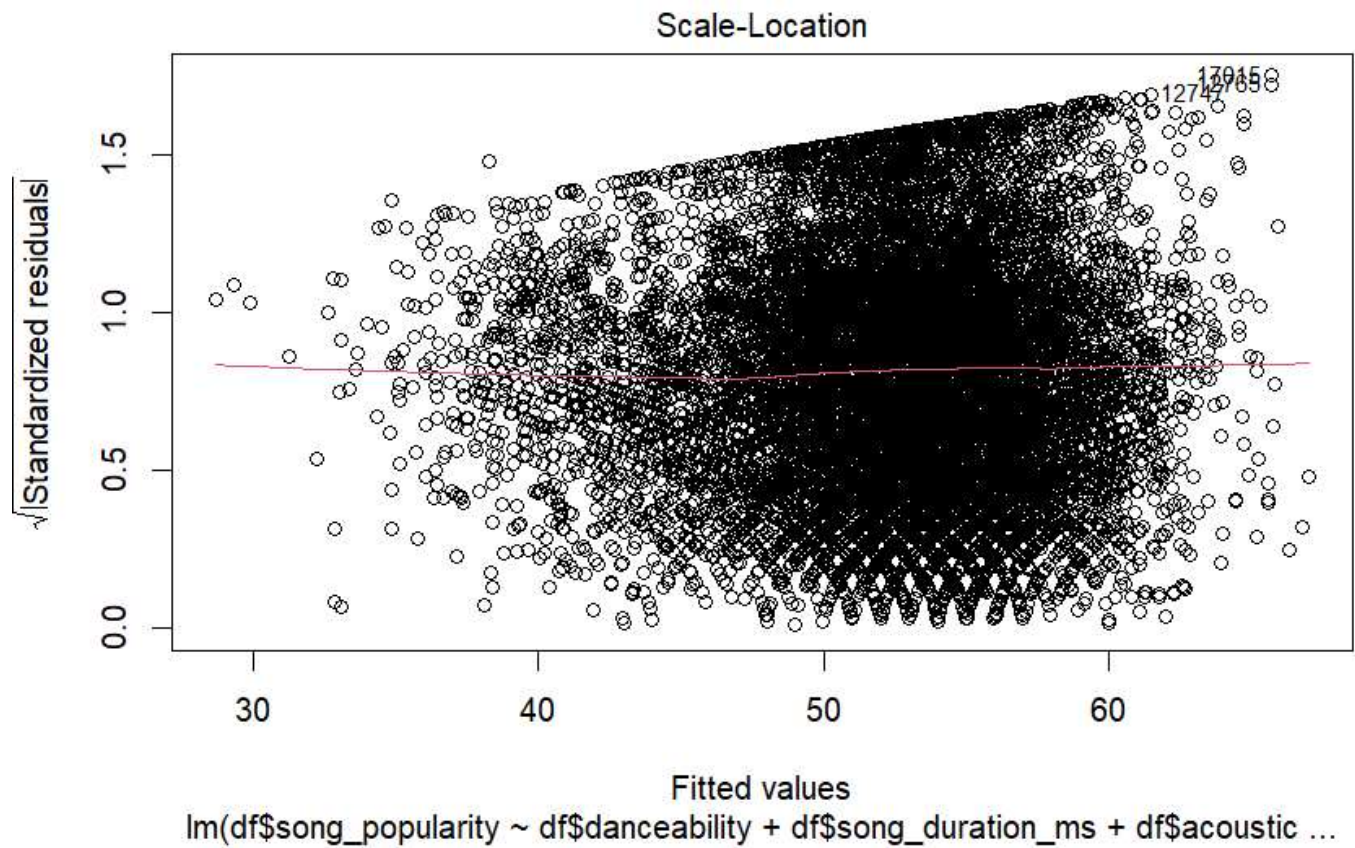
F-statistic: 29.14 on 25 and 15042 DF, p-value: < 2.2e-16

Plotting residuals:

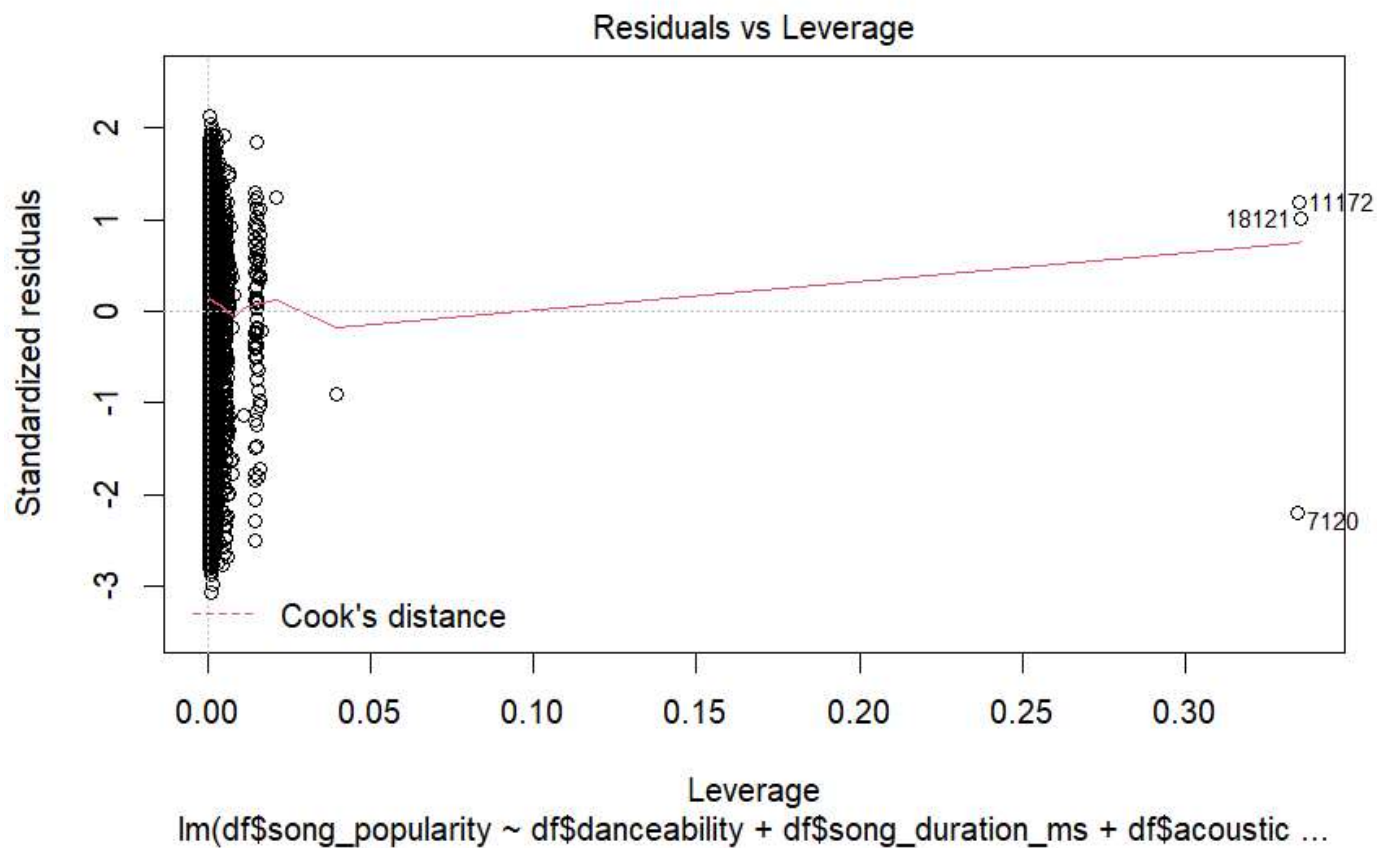
Hide

```
plot(lm2)
```




[Hide](#)

```
par(mfrow=c(1,1))
```



g. Third linear model to try to improve results

Hide

```
str(df)
```

```
'data.frame':  18835 obs. of  15 variables:
 $ song_name      : chr  "Boulevard of Broken Dreams" "In The End" "Seven Nation Army" "By The
Way" ...
 $ song_popularity : int   73 66 76 74 56 80 81 76 80 81 ...
 $ song_duration_ms: int  262333 216933 231733 216933 223826 235893 199893 213800 222586 203346
...
 $ acousticness   : num   0.00552 0.0103 0.00817 0.0264 0.000954 0.00895 0.000504 0.00148 0.0010
8 0.00172 ...
 $ danceability   : num   0.496 0.542 0.737 0.451 0.447 0.316 0.581 0.613 0.33 0.542 ...
 $ energy         : num   0.682 0.853 0.463 0.97 0.766 0.945 0.887 0.953 0.936 0.905 ...
 $ instrumentalness: num   2.94e-05 0.00 4.47e-01 3.55e-03 0.00 1.85e-06 1.11e-03 5.82e-04 0.00
1.04e-02 ...
 $ key           : Factor w/ 12 levels "0","1","2","3",...: 9 4 1 1 11 5 5 3 2 10 ...
 $ liveness      : num   0.0589 0.108 0.255 0.102 0.113 0.396 0.268 0.152 0.0926 0.136 ...
 $ loudness      : num   -4.09 -6.41 -7.83 -4.94 -5.07 ...
 $ audio_mode    : Factor w/ 2 levels "0","1": 2 1 2 2 2 1 1 2 2 2 ...
 $ speechiness   : num   0.0294 0.0498 0.0792 0.107 0.0313 0.124 0.0624 0.0855 0.0917 0.054 ...
 $ tempo         : num   167 105 124 122 172 ...
 $ time_signature : Factor w/ 5 levels "0","1","3","4",...: 4 4 4 4 4 4 4 4 4 4 ...
 $ audio_valence  : num   0.474 0.37 0.324 0.198 0.574 0.32 0.724 0.537 0.234 0.374 ...
```

Hide

```
lm3 <- lm(song_popularity~danceability+song_duration_ms+acousticness+energy+instrumentalness+key
+liveness+speechiness+time_signature+audio_valence, data=train)
summary(lm3)
```

Call:

```
lm(formula = song_popularity ~ danceability + song_duration_ms +
    acousticness + energy + instrumentalness + key + liveness +
    speechiness + time_signature + audio_valence, data = train)
```

Residuals:

Min	1Q	Median	3Q	Max
-65.475	-12.429	2.865	15.730	46.421

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	4.702e+01	1.242e+01	3.786	0.000154	***
danceability	1.537e+01	1.295e+00	11.868	< 2e-16	***
song_duration_ms	-8.763e-06	3.080e-06	-2.845	0.004442	**
acousticness	-4.019e+00	8.492e-01	-4.733	2.24e-06	***
energy	-2.912e+00	1.200e+00	-2.426	0.015263	*
instrumentalness	-1.390e+01	8.111e-01	-17.133	< 2e-16	***
key1	1.344e+00	7.359e-01	1.827	0.067792	.
key2	-1.511e+00	7.765e-01	-1.946	0.051699	.
key3	-1.194e+00	1.194e+00	-1.000	0.317485	
key4	-1.477e+00	8.387e-01	-1.761	0.078340	.
key5	-6.768e-01	7.913e-01	-0.855	0.392388	
key6	1.326e+00	8.293e-01	1.599	0.109773	
key7	-2.597e+00	7.386e-01	-3.516	0.000439	***
key8	-1.439e+00	8.282e-01	-1.738	0.082248	.
key9	-2.041e+00	7.763e-01	-2.629	0.008580	**
key10	-4.153e-02	8.313e-01	-0.050	0.960160	
key11	2.386e-01	7.940e-01	0.300	0.763841	
liveness	-4.658e+00	1.244e+00	-3.743	0.000182	***
speechiness	-4.400e+00	1.775e+00	-2.479	0.013194	*
time_signature1	7.074e+00	1.265e+01	0.559	0.576173	
time_signature3	8.015e+00	1.241e+01	0.646	0.518286	
time_signature4	9.172e+00	1.238e+01	0.741	0.458827	
time_signature5	1.103e+01	1.248e+01	0.884	0.376792	
audio_valence	-9.783e+00	8.255e-01	-11.851	< 2e-16	***

Signif. codes:

0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21.4 on 15044 degrees of freedom

Multiple R-squared: 0.0459, Adjusted R-squared: 0.04444

F-statistic: 31.46 on 23 and 15044 DF, p-value: < 2.2e-16

Hide

```
#?predict
pred3 <- predict(lm3, newdata=test)
summary(pred3)
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
30.32	50.61	53.36	52.93	55.95	66.81

Hide

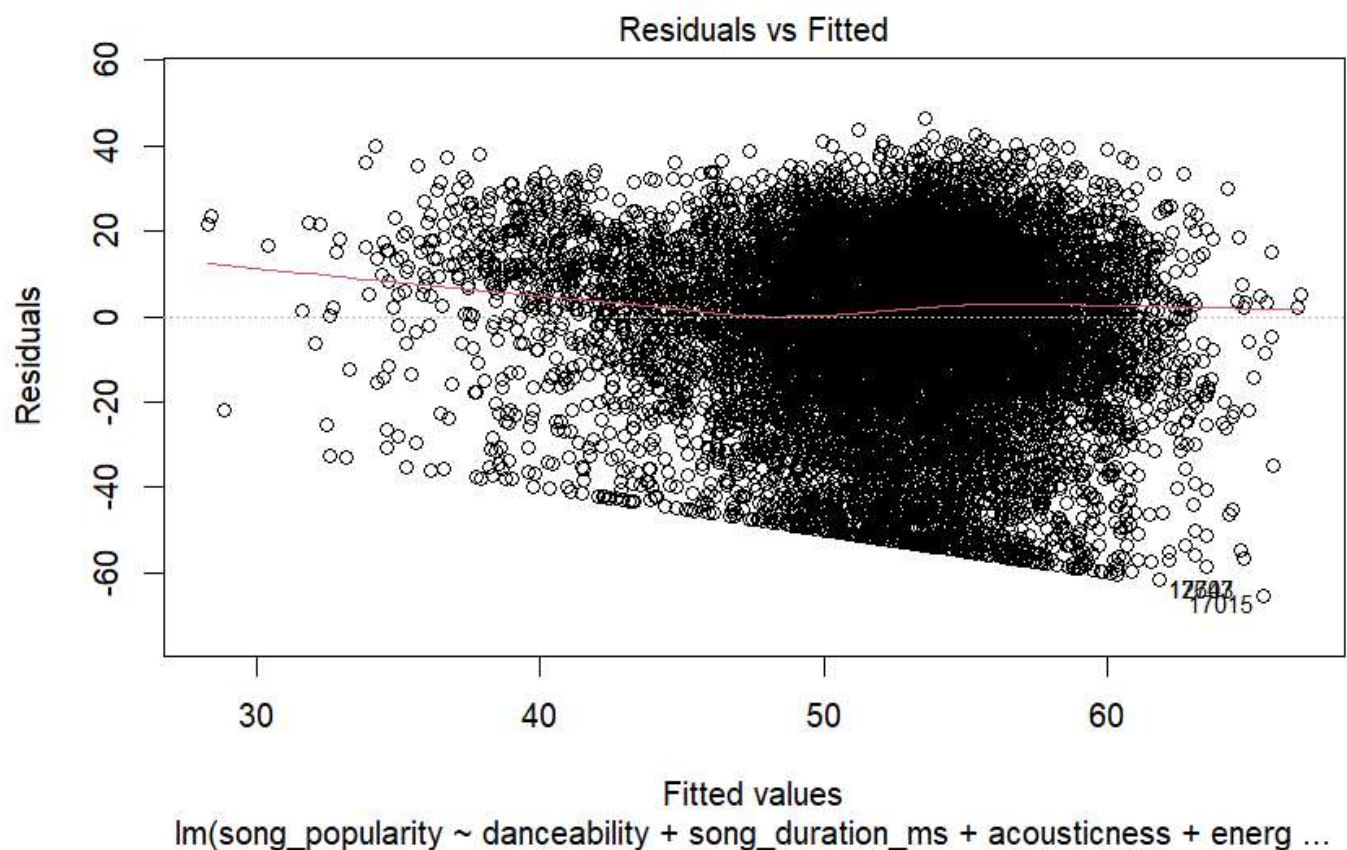
```
#cor(pred1, test$song_popularity)
#sqrt(mean((pred1-test$song_popularity)^2))
#range(test$song_popularity)
```

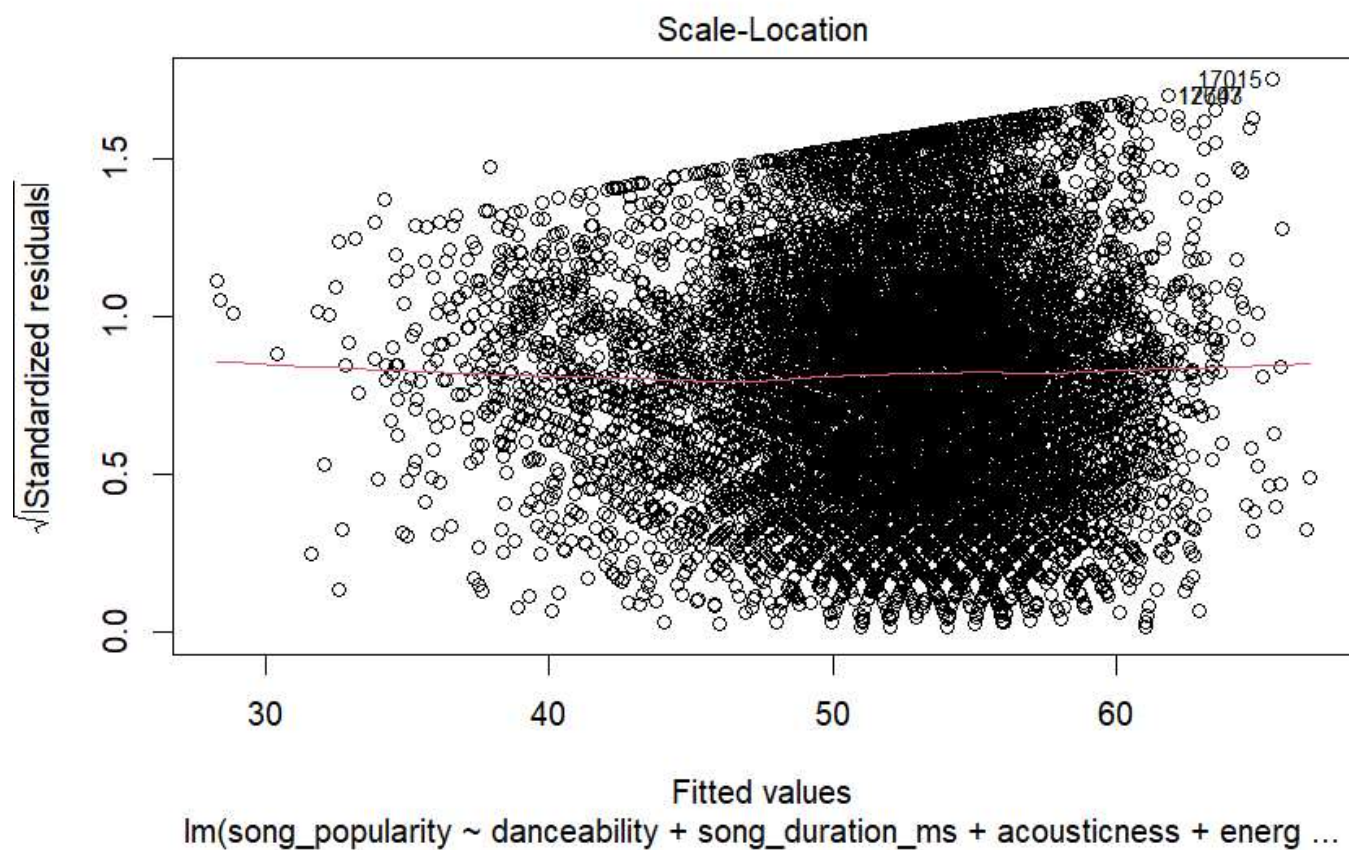
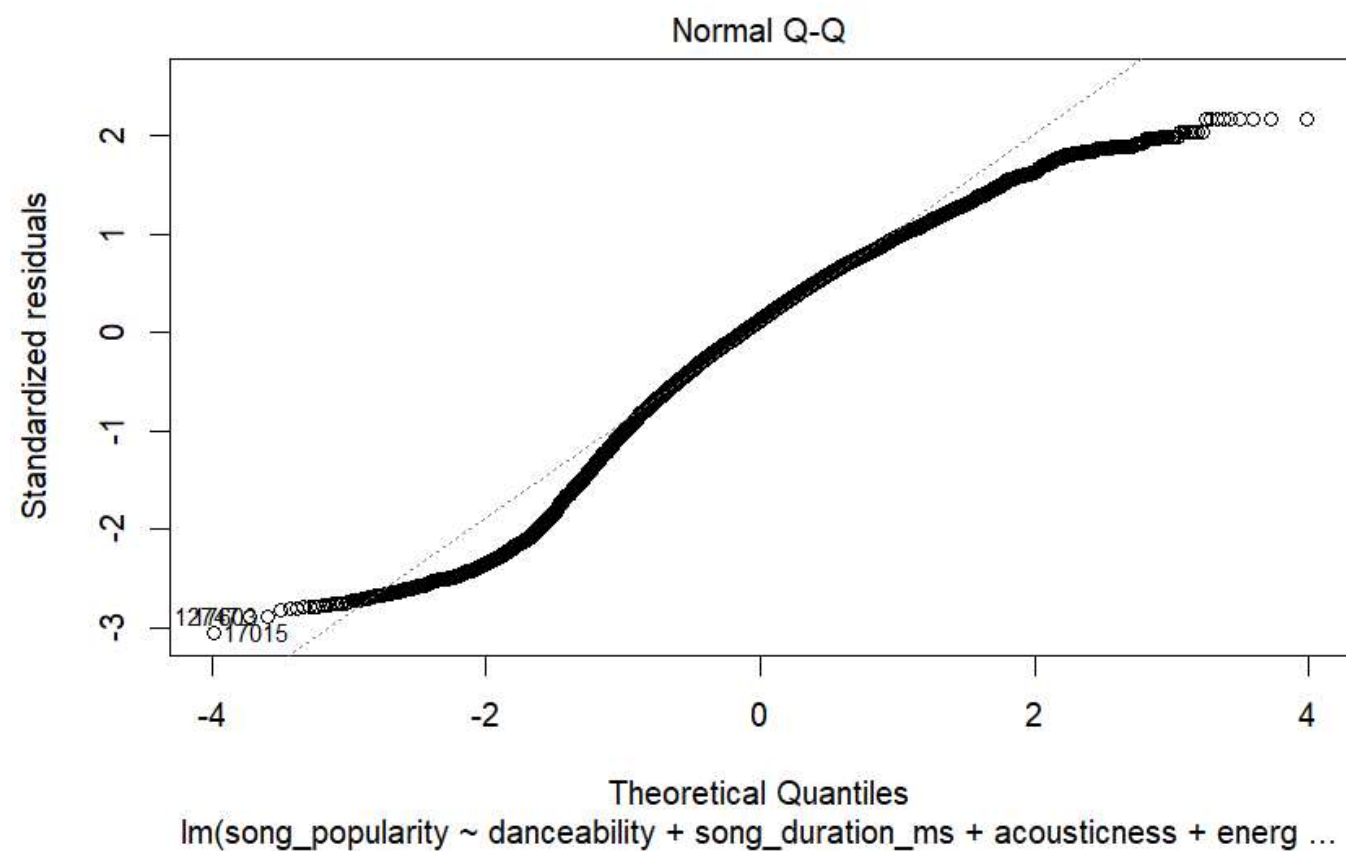
Used all predictors except song name, loudness, audio mode, and tempo after observing their significance in linear regression models.

Plotting residuals:

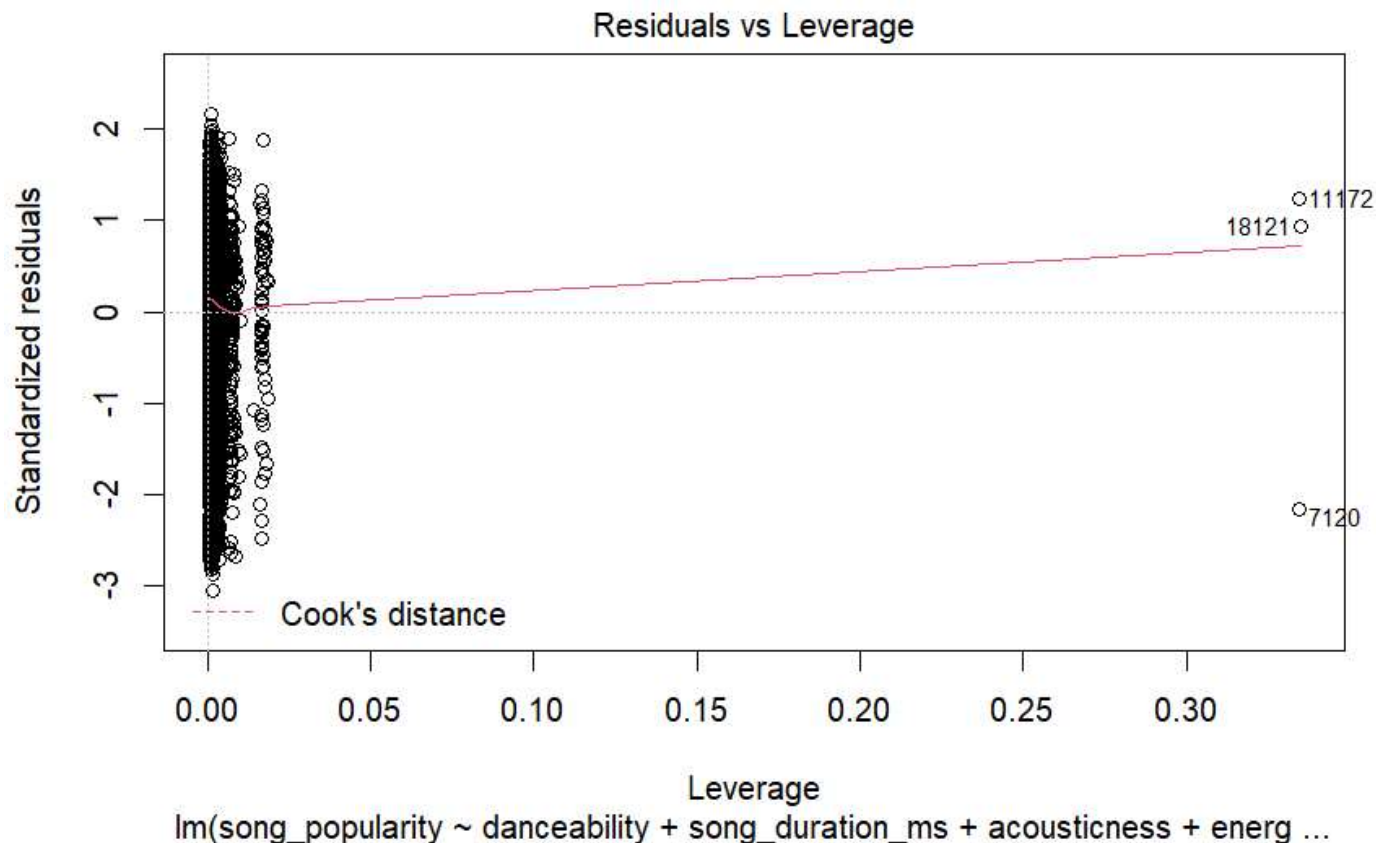
Hide

```
plot(lm3)
```




[Hide](#)

```
par(mfrow=c(1,1))
```



h. There is a minor difference between the residual plot of the first model and the others. On the other hand, there are differences in the logistic regression models. The minimum residual for the first model was -57.938, and was -65.625 and -65.475 for the second and third models. The maximum residuals are about the same, with a difference of about 1 (in the popularity score). In all the three models, danceability was a significant predictor (with similar p-values) but its coefficient in the regression equation was different (~15 for the first model and ~1.5e+01 for the others). Obviously, there were other significant predictors in the second and third model, with the third model removing the predictors that the model felt were not helpful (loudness, audio mode, and tempo). In the second and third models, the most significant predictors are danceability, instrumentality, and audio valence. Acousticness and liveness are also important predictors. Other predictors like song duration, energy, key, and speechiness are not as important, but removing them from the model would decrease the accuracy. The R^2 values for the models are 0.01148, 0.04461, and 0.04444 respectively. There is a significant increase in the R^2 value from the first model, which is great. Unfortunately, I was unable to find a combination of predictors that improved the model to have a greater value of R^2 that is closer to 1. The residual standard error is similar in all models, but the F-statistic is different (it changes from 175.9 to 29.14). The overall p-value is small in all the models, which is good. I think the second model is better because it has all the predictors (as it seems that most predictors are greatly important in predicting the song popularity) and has the greatest R-squared value. The third model is not much different, but with less predictors, so it might predict the test values better.

i. Predict and evaluate on test data

First model:

Hide

```
pred1 <- predict(lm1, newdata=test)
cor(pred1, test$song_popularity)
```

```
[1] 0.09199206
```

[Hide](#)

```
mean((pred1-test$song_popularity)^2)
```

```
[1] 477.7823
```

[Hide](#)

```
sqrt(mean((pred1-test$song_popularity)^2))
```

```
[1] 21.85823
```

Second model:

[Hide](#)

```
pred2 <- predict(lm2, newdata=test)
cor(pred2, test$song_popularity)
```

```
[1] 0.1871685
```

[Hide](#)

```
mean((pred2-test$song_popularity)^2)
```

```
[1] 465.2417
```

[Hide](#)

```
sqrt(mean((pred2-test$song_popularity)^2))
```

```
[1] 21.56946
```

Third model:

[Hide](#)

```
pred3 <- predict(lm3, newdata=test)
cor(pred3, test$song_popularity)
```

```
[1] 0.1875484
```

[Hide](#)

```
mean((pred3-test$song_popularity)^2)
```

```
[1] 465.1427
```

[Hide](#)

```
sqrt(mean((pred3-test$song_popularity)^2))
```

```
[1] 21.56717
```

The correlation is highest for the third model because the insignificant predictors have been removed. The second model has a similar correlation (since it had all of the same predictors plus a few more insignificant ones) but the first one is significantly less (due to only having one significant predictor). The mse is ~12 more in the first model compared to the second and third one, while the rmse amounts to about the same in each. It is interesting to note that the difference in predictions is about the same in all the models but the correlation varies. This might be because the second and third models have a greater difference in predicted and actual popularity for some test cases, while having higher accuracy overall.