Top Song Predictor

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(pROC)
```

```
## Warning: package 'pROC' was built under R version 3.6.2
```

```
## Type 'citation("pROC")' for a citation.
```

```
##
## Attaching package: 'pROC'
```

```
## The following objects are masked from 'package:stats':
##
##     cov, smooth, var
```

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 3.6.2
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.1 --
```

```
## v tibble  3.1.7     v purrr   0.3.4
## v tidyr   1.2.0     v stringr 1.4.0
## v readr   2.1.2     v forcats 0.5.1
```

```
## Warning: package 'tidyr' was built under R version 3.6.2
```

```
## Warning: package 'readr' was built under R version 3.6.2
```

```
## Warning: package 'purrr' was built under R version 3.6.2
```

```
## Warning: package 'forcats' was built under R version 3.6.2
```

```
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
```

```
## x dplyr::lag()    masks stats::lag()
```

Helper Functions

Data Analysis

```
data <-read.csv('MusicData.csv')
data[is.na(data)]
```

```
## character(0)
```

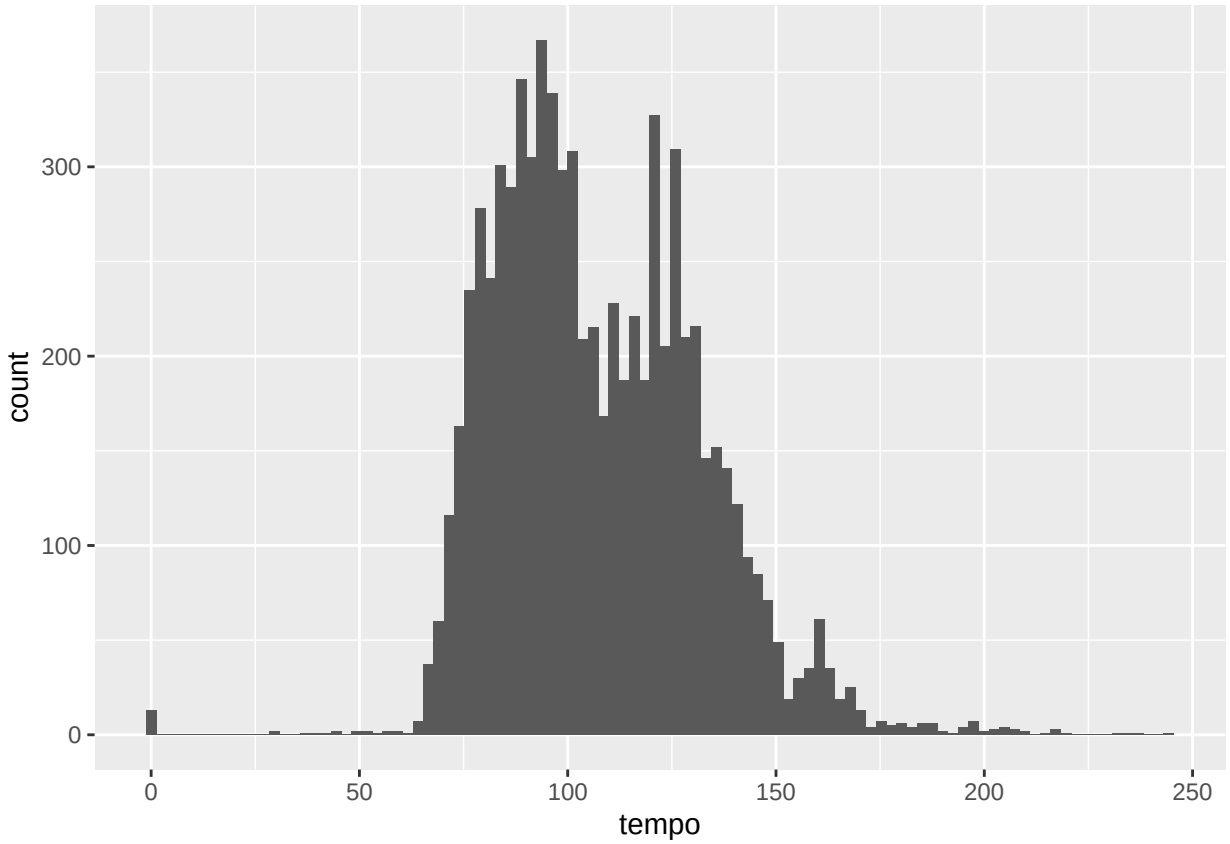
There is no missing data

```
dim(data)
```

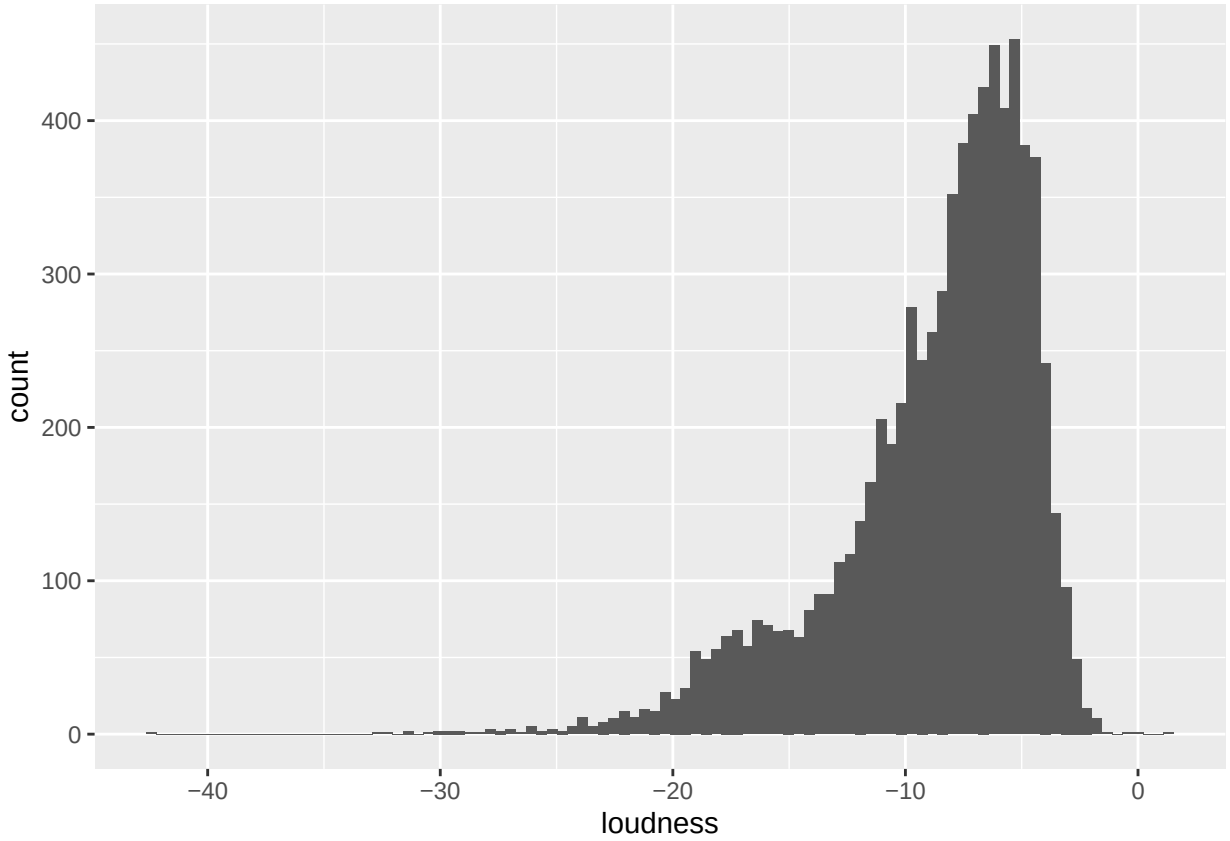
```
## [1] 7574   39
```

There's 7574 rows and 39 columns

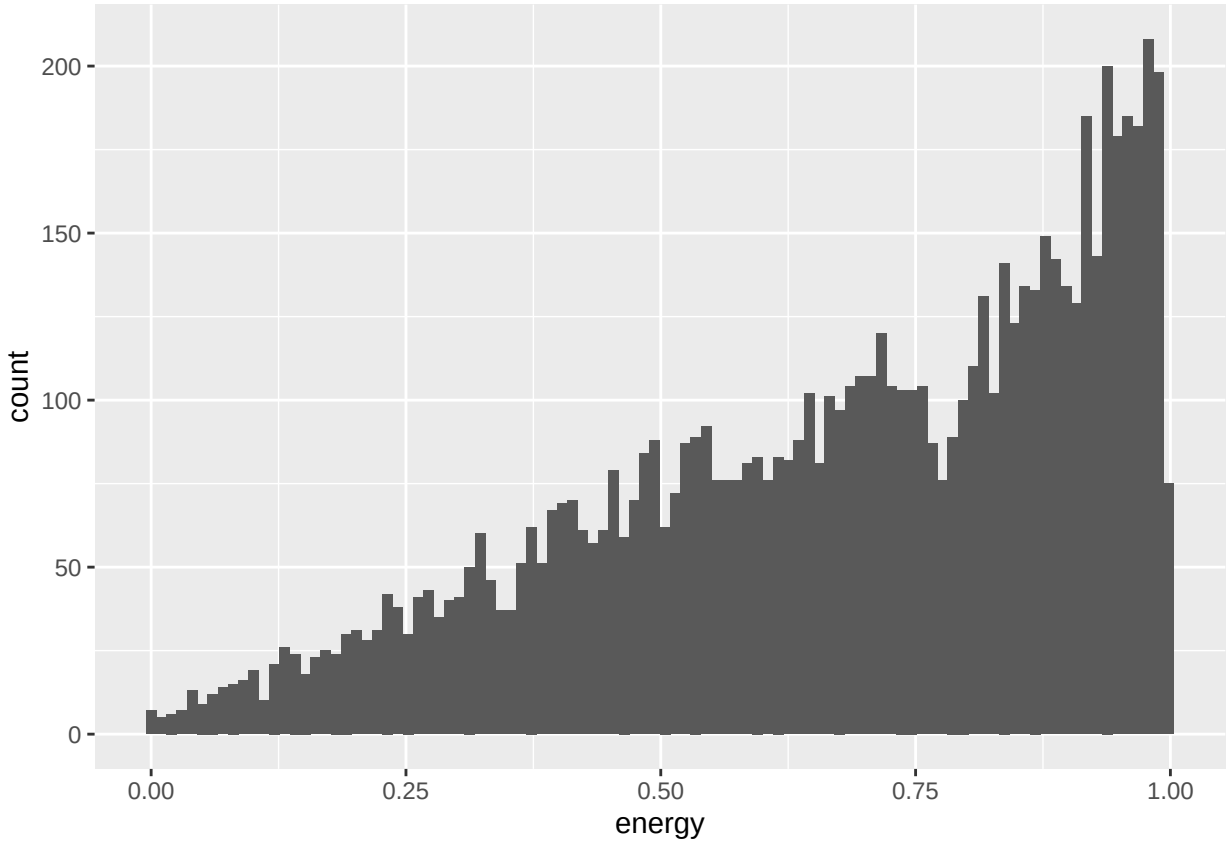
```
p <- ggplot(data, aes(x=tempo)) + geom_histogram(bins = 100)
print(p)
```




```
p <- ggplot(data, aes(x=loudness)) + geom_histogram(bins = 100)
print(p)
```




```
p <- ggplot(data, aes(x=energy)) + geom_histogram(bins = 100)
print(p)
```



Data Processing

```
# standardize data = center data and get equal variance
data$loudness <- standardize(data$loudness)
data$tempo <- standardize(data$tempo)
data$energy <- standardize(data$energy)

# replace the outliers for standardized variables
data$loudness <- replace_outliers(data$loudness)
data$tempo <- replace_outliers(data$tempo)
data$energy <- replace_outliers(data$energy)
```

Train - Test Split

```
SongsTrain = data %>% filter(year <= 2009)
SongsTest = data %>% filter(year == 2010)

#non-predictors
nonvars = c("year", "songtitle", "artistname", "songID", "artistID")

# To remove these variables from your training and testing sets:
```

```
SongsTrain = SongsTrain[ , !(names(SongsTrain) %in% nonvars) ]
SongsTest = SongsTest[ , !(names(SongsTest) %in% nonvars) ]
```

Modelling

```
mod1 <- glm(formula = Top10 ~ . , family = binomial, data = SongsTrain)

mod2 <- glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min,
            family = binomial, data = SongsTrain)
mod3 <- glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min
            - key - timbre_6_max - timesignature - timbre_7_max,
            family = binomial, data = SongsTrain)

mod4 <- glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min
            - key - timbre_6_max - timesignature - timbre_7_max
            - key_confidence - timbre_7_min - timbre_10_min,
            family = binomial, data = SongsTrain)

mod5 <- glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min
            - key - timbre_6_max - timesignature - timbre_7_max
            - key_confidence - timbre_7_min - timbre_10_min-energy,
            family = binomial, data = SongsTrain)

mod6 <- glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min
            - key - timbre_6_max - timesignature - timbre_7_max
            - key_confidence - timbre_7_min - timbre_10_min-energy - tempo
            - timbre_1_max - timbre_2_max - timbre_3_min - timbre_5_max
            -timbre_9_min  - timbre_9_max,
            family = binomial, data = SongsTrain)
```

Performance

```
eval1 <- classification_report(mod1)
eval2 <- classification_report(mod2)
eval3 <- classification_report(mod3)
eval4 <- classification_report(mod4)
eval5 <- classification_report(mod5)
eval6 <- classification_report(mod6)

results <- data.frame(first_mod = c(0,0,0,0), second_mod = eval2-eval1, third_mod=eval3-eval1, fourth_m
                      sixth_mod = eval6-eval1)

metrics <-c("accuracy","precision","recall","f1")
rownames(results) <- metrics

# transpose results
results <- as.data.frame(t(results))

results
```

```
##                accuracy    precision     recall           f1
## first_mod   0.000000000  0.000000000 0.00000000  0.000000000
## second_mod  0.005361930  0.018333333 0.03389831  0.027780159
## third_mod  -0.002680965 -0.011054422 0.00000000 -0.004499827
## fourth_mod -0.002680965 -0.012254902 0.01694915  0.004927782
## fifth_mod   0.010723861  0.029761905 0.10169492  0.070540431
## sixth_mod   0.000000000 -0.003205128 0.03389831  0.018523196
```

plot the performance

```
ggplot(results, aes(x=c(1,2,3,4,5,6))) +
  geom_line(aes(y = accuracy), color = "black") +
  geom_line(aes(y =precision), color = "red") +
  geom_line(aes(y = recall), color = "green") +
  geom_line(aes(y = f1), color = "blue") +
  xlab("Model Number") +
  ylab("Differences") +
  ggtitle("Difference in Model Performance")
```

The first model had used all predictors.

Model 2

Based off the p-value for the predictors, we removed an additional 3 variables that was the most insignificant, which were `timbre_8_min`, `timbre_8_max`, and `timbre_2_min`. The model's overall performance had improved.

Model 3

Based off the p-value for the predictors, we removed an additional 4 variables that was the most insignificant, `key`, `timbre_6_max`, `timesignature`, and `timbre_7_max`. The model's performance had signficanetly decreased in all areas of performance.

Model 4

Based off the p-value for the predictors, we removed an additional 3 variables that was the most insignificant,`key_confidence`, `timbre_7_max`, and `timbre_10_min`. There was an improvement from model 3 but not quite as good performing as model 2. Model 4 is perferred over model 2 since its performance metrics are similar and model 4 has less complexity.

Model 5

Based off the p-value for the predictors, we removed an additional variable that was the most insignificant, `energy`. This model would be the most perferred over the previous models since it has the lowest complexity and best performance metrics in all 4 areas, accuracy, precision, recall, and f1 score.

Model 6

Based off the p-value for the predictors, we removed an additional variable that was the most insignificant, `tempo`,`timbre_1_max`,`timbre_2_max` , `timbre_3_min`,`timbre_5_max`,`timbre_9_min` and `timbre_9_max`. The performance had decreased in all areas. All predictors are significant. If we were to only consider the accuracy of a model, then we would prefer model 6 over model 5 since the complexity of model 6 is 7 variables less than model 5 and a relatively similar performing model with a lower complexity is preferred.

```
performances <- data.frame(first_mod = eval1, second_mod = eval2, third_mod=eval3, fourth_mod=eval4, fi

rownames(performances) <- metrics

# transpose performances
performances <- as.data.frame(t(performances))
performances
```

```
##             accuracy precision    recall        f1
## first_mod  0.8525469 0.5416667 0.4406780 0.4859813
## second_mod 0.8579088 0.5600000 0.4745763 0.5137615
## third_mod  0.8498660 0.5306122 0.4406780 0.4814815
## fourth_mod 0.8498660 0.5294118 0.4576271 0.4909091
## fifth_mod  0.8632708 0.5714286 0.5423729 0.5565217
## sixth_mod  0.8525469 0.5384615 0.4745763 0.5045045
```

The final model with its predictors is shown below.

```
summary(mod6)
```

```
##
## Call:
## glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min -
##     key - timbre_6_max - timesignature - timbre_7_max - key_confidence -
##     timbre_7_min - timbre_10_min - energy - tempo - timbre_1_max -
##     timbre_2_max - timbre_3_min - timbre_5_max - timbre_9_min -
##     timbre_9_max, family = binomial, data = SongsTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7761  -0.5534  -0.3598  -0.1932   3.3869
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               6.505e+00  1.181e+00   5.507 3.66e-08 ***
## timesignature_confidence  7.824e-01  1.885e-01   4.151 3.31e-05 ***
## loudness                  3.193e+00  4.370e-01   7.307 2.74e-13 ***
## tempo_confidence          4.829e-01  1.369e-01   3.528 0.000419 ***
## pitch                    -5.794e+01  6.340e+00  -9.138  < 2e-16 ***
## timbre_0_min              2.426e-02  4.096e-03   5.923 3.16e-09 ***
## timbre_0_max             -2.416e-01  2.088e-02 -11.573  < 2e-16 ***
## timbre_1_min              5.354e-03  6.620e-04   8.088 6.05e-16 ***
## timbre_3_max             -3.458e-03  4.883e-04  -7.081 1.43e-12 ***
## timbre_4_min              9.311e-03  1.854e-03   5.022 5.11e-07 ***
## timbre_4_max              6.761e-03  1.363e-03   4.959 7.10e-07 ***
## timbre_5_min             -6.454e-03  1.178e-03  -5.479 4.28e-08 ***
## timbre_6_min             -1.838e-02  2.077e-03  -8.847  < 2e-16 ***
## timbre_10_max             7.206e-03  1.635e-03   4.407 1.05e-05 ***
## timbre_11_min            -2.972e-02  3.525e-03  -8.431  < 2e-16 ***
## timbre_11_max             2.150e-02  3.144e-03   6.837 8.11e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6017.5  on 7200  degrees of freedom
## Residual deviance: 4861.9  on 7185  degrees of freedom
## AIC: 4893.9
##
## Number of Fisher Scoring iterations: 6
```

```
base_last <- data.frame(first_mod = eval1, sixth_mod=eval6)

rownames(base_last) <- metrics

# transpose results
base_last <- as.data.frame(t(base_last))
base_last
```

```
##            accuracy precision    recall        f1
## first_mod 0.8525469 0.5416667 0.4406780 0.4859813
## sixth_mod 0.8525469 0.5384615 0.4745763 0.5045045
```

```
thresholds = seq.int(5,84,5)/100
```

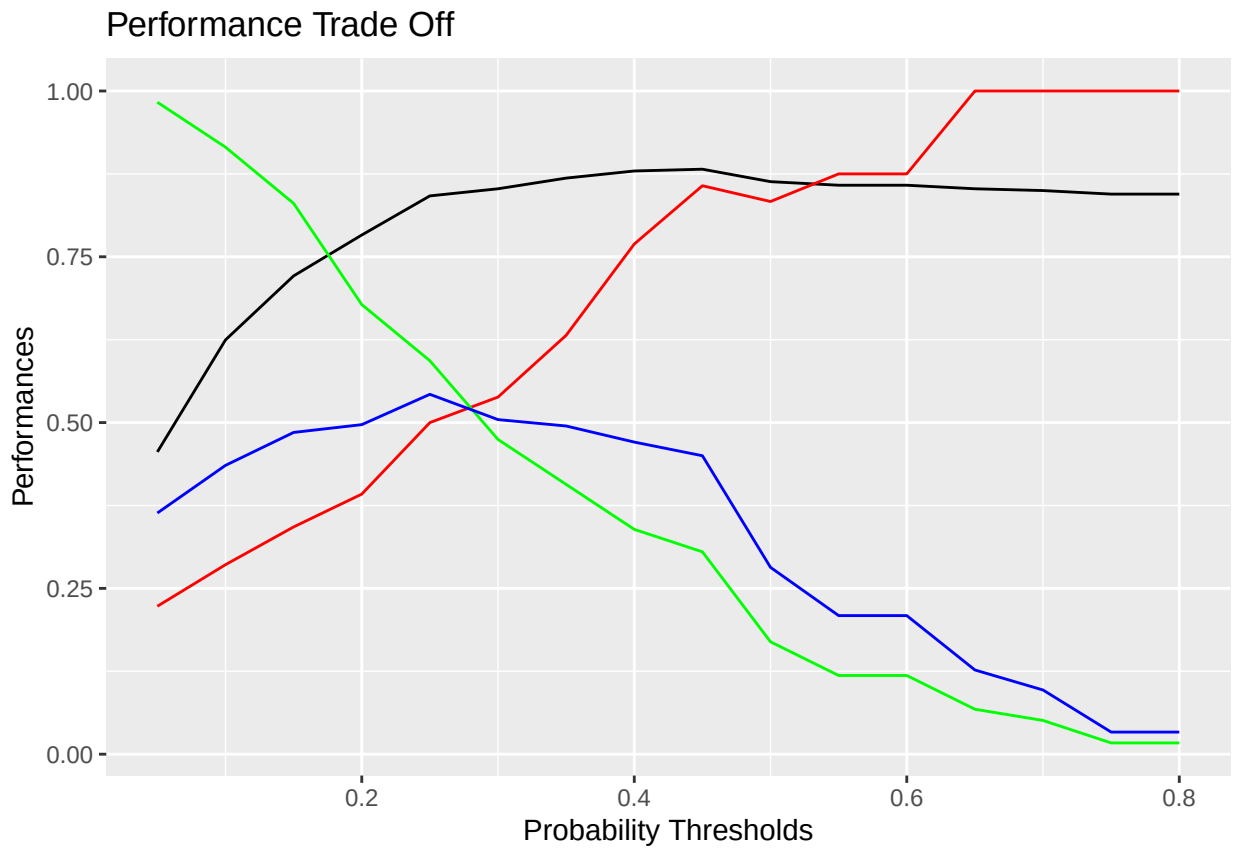
```
acc <- c()
pre <- c()
rec <- c()
f1 <- c()

for (i in thresholds){
  values <- classification_report(mod6,i)
  acc <- append(acc, values[1])
  pre <- append(pre, values[2])
  rec <- append(rec, values[3])
  f1 <- append(f1, values[4])
}

best_thresh <- data.frame(accuracy = acc, precision = pre, recall = rec, f1 = f1)

plot_it <- ggplot(best_thresh, aes(x=thresholds[1:16])) +
  geom_line(aes(y = accuracy), color = "black") +
  geom_line(aes(y =precision), color = "red") +
  geom_line(aes(y = recall), color = "green") +
  geom_line(aes(y = f1), color = "blue") +
  xlab("Probability Thresholds") +
  ylab("Performances") +
  ggtitle("Performance Trade Off")

plot_it
```



Our best middle ground is about when the threshold is 0.30

```
classification_report(mod6,0.30)
```

```
## [1] 0.8525469 0.5384615 0.4745763 0.5045045
```

```
# accuracy, precision, recall, f1
```

Interpret Coefficients

```
mod_summary <- summary(mod6)
mod_summary
```

```
##
## Call:
## glm(formula = Top10 ~ . - timbre_8_min - timbre_8_max - timbre_2_min -
##     key - timbre_6_max - timesignature - timbre_7_max - key_confidence -
##     timbre_7_min - timbre_10_min - energy - tempo - timbre_1_max -
##     timbre_2_max - timbre_3_min - timbre_5_max - timbre_9_min -
##     timbre_9_max, family = binomial, data = SongsTrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7761  -0.5534  -0.3598  -0.1932   3.3869
##
## Coefficients:
##                            Estimate Std. Error z value Pr(>|z|)
## (Intercept)               6.505e+00  1.181e+00   5.507 3.66e-08 ***
## timesignature_confidence  7.824e-01  1.885e-01   4.151 3.31e-05 ***
## loudness                  3.193e+00  4.370e-01   7.307 2.74e-13 ***
## tempo_confidence          4.829e-01  1.369e-01   3.528 0.000419 ***
## pitch                    -5.794e+01  6.340e+00  -9.138  < 2e-16 ***
## timbre_0_min              2.426e-02  4.096e-03   5.923 3.16e-09 ***
## timbre_0_max             -2.416e-01  2.088e-02 -11.573  < 2e-16 ***
## timbre_1_min              5.354e-03  6.620e-04   8.088 6.05e-16 ***
## timbre_3_max             -3.458e-03  4.883e-04  -7.081 1.43e-12 ***
## timbre_4_min              9.311e-03  1.854e-03   5.022 5.11e-07 ***
## timbre_4_max              6.761e-03  1.363e-03   4.959 7.10e-07 ***
## timbre_5_min             -6.454e-03  1.178e-03  -5.479 4.28e-08 ***
## timbre_6_min             -1.838e-02  2.077e-03  -8.847  < 2e-16 ***
## timbre_10_max             7.206e-03  1.635e-03   4.407 1.05e-05 ***
## timbre_11_min            -2.972e-02  3.525e-03  -8.431  < 2e-16 ***
## timbre_11_max             2.150e-02  3.144e-03   6.837 8.11e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 6017.5  on 7200  degrees of freedom
## Residual deviance: 4861.9  on 7185  degrees of freedom
## AIC: 4893.9
##
## Number of Fisher Scoring iterations: 6
```

The coefficients above are logged odd ratios.

```
print(mod_summary$coefficients[2:6])
```

```
## [1]   0.7824316   3.1930916   0.4828912 -57.9414514   0.0242596
```

timesignature_confidence 0.78 loudness 3.19 tempo_confidence 0.48 pitch h -57.94
timbre_0_min 0.02

To understand the coeffients we need to have an exponential transformation on the coefficients

```
print(exp(mod_summary$coefficients[2:6]))
```

```
## [1] 2.186783e+00 2.436363e+01 1.620754e+00 6.860367e-26 1.024556e+00
```

For each unit of increase in timesignature_confidence, there is an increase odds of 2.19 the song will be a top billboard song on average.

For each unit of loudness, there is an increase in odds of 3.19 the song will be a top billboard song on average.

For each unit of tempo_confidence, there is an increase in odds of 0.48 the song will be a top billboard song on average.

For each unit of pitch, there is an increase in odds of -57.94 the song will be a top billboard song on average.

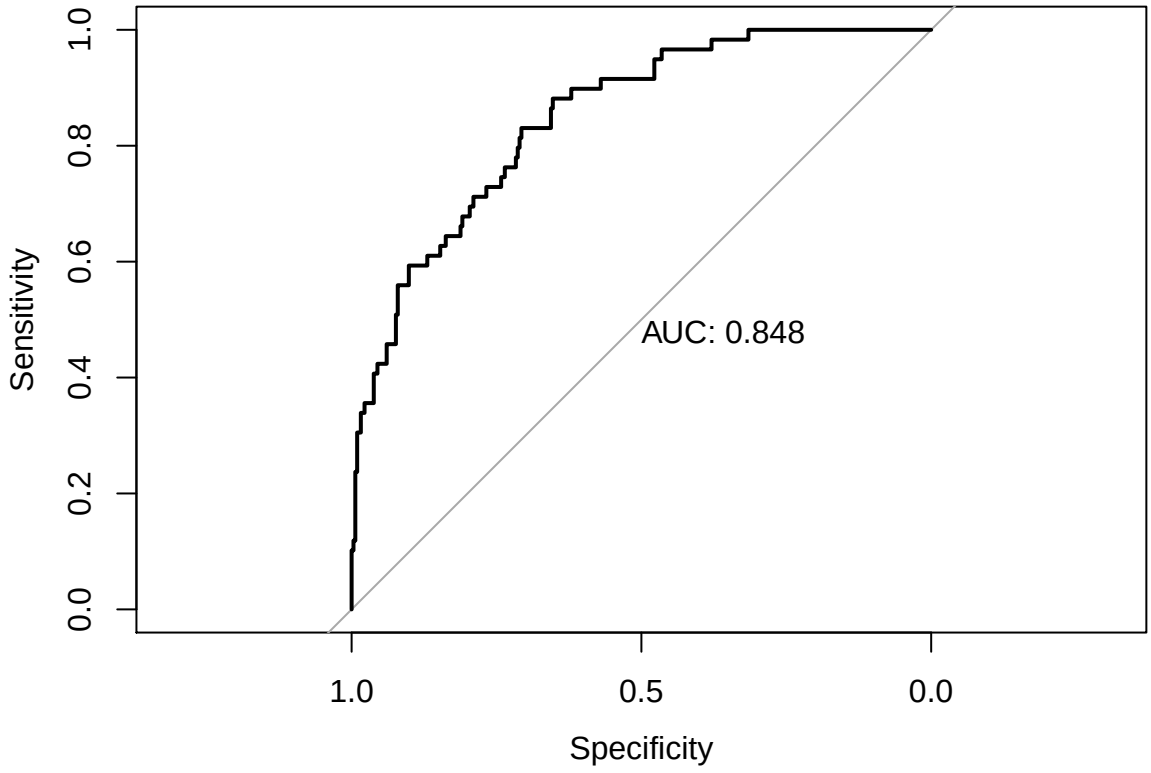
For each unit of timbre_0_min, there is an increase in odds of 0.02 the song will be a top billboard song on average.

ROC

```
test_prob = predict(mod6, newdata = SongsTest, type = "response")
test_roc = roc(SongsTest$Top10 ~ test_prob, plot = TRUE, print.auc = TRUE)
```

```
## Setting levels: control = 0, case = 1
```

```
## Setting direction: controls < cases
```




```
test_roc
```

```
##
## Call:
## roc.formula(formula = SongsTest$Top10 ~ test_prob, plot = TRUE,     print.auc = TRUE)
##
## Data: test_prob in 314 controls (SongsTest$Top10 0) < 59 cases (SongsTest$Top10 1).
## Area under the curve: 0.8481
```

The AUC value of 0.848 means the model predicted 84.8% predictions correctly from the testing data.

Some predictions

```
testPredict = predict(mod6, newdata=SongsTest, type="response")
testPredict[1:10]
```

```
##           1           2           3           4           5           6
## 0.034411865 0.031689264 0.040368455 0.063975455 0.001219940 0.013628954
##           7           8           9          10
## 0.033937560 0.006666321 0.018631647 0.351557245
```

Above shows the prediction probabilities

```
testPredict[1:10] > 0.30
```

```
##     1     2     3     4     5     6     7     8     9    10
## FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE  TRUE
```

In the case above, only song number 10 will be a top billboards song.

```
table(SongsTest$Top10, testPredict >= 0.30)
```

```
##
##      FALSE TRUE
##   0   290   24
##   1    31   28
```