Predicting Popularity of a Song

Nutan Sahoo

27 September 2017

Exploratory Analysis

```
#Importing data set
songs<- read.csv("songs.csv", header = TRUE)
View(songs)

#check structure of the data set
str(songs)
```

```
## 'data.frame':    7574 obs. of  39 variables:
##  $ year                    : int  2010 2010 2010 2010 2010 2010 2010 2010 2010 2010 ...
##  $ songtitle               : Factor w/ 7141 levels "'03 Bonnie & Clyde",..: 6204 5522 241 3098 47 60
##  $ artistname              : Factor w/ 1032 levels "50 Cent","98 Degrees",..: 3 3 3 3 3 3 3 3 3 12 .
##  $ songID                  : Factor w/ 7549 levels "SOAACNI1315CD4AC42",..: 595 5439 5252 1716 3431 
##  $ artistID                : Factor w/ 1047 levels "AR00B1I1187FB433EB",..: 671 671 671 671 671 671 
##  $ timesignature           : int  3 4 4 4 4 4 4 4 4 4 ...
##  $ timesignature_confidence: num  0.853 1 1 1 0.788 1 0.968 0.861 0.622 0.938 ...
##  $ loudness                : num  -4.26 -4.05 -3.57 -3.81 -4.71 ...
##  $ tempo                   : num  91.5 140 160.5 97.5 140.1 ...
##  $ tempo_confidence        : num  0.953 0.921 0.489 0.794 0.286 0.347 0.273 0.83 0.018 0.929 ...
##  $ key                     : int  11 10 2 1 6 4 10 5 9 11 ...
##  $ key_confidence          : num  0.453 0.469 0.209 0.632 0.483 0.627 0.715 0.423 0.751 0.602 ...
##  $ energy                  : num  0.967 0.985 0.99 0.939 0.988 ...
##  $ pitch                   : num  0.024 0.025 0.026 0.013 0.063 0.038 0.026 0.033 0.027 0.004 ...
##  $ timbre_0_min            : num  0.002 0 0.003 0 0 ...
##  $ timbre_0_max            : num  57.3 57.4 57.4 57.8 56.9 ...
##  $ timbre_1_min            : num  -6.5 -37.4 -17.2 -32.1 -223.9 ...
##  $ timbre_1_max            : num  171 171 171 221 171 ...
##  $ timbre_2_min            : num  -81.7 -149.6 -72.9 -138.6 -147.2 ...
##  $ timbre_2_max            : num  95.1 180.3 157.9 173.4 166 ...
##  $ timbre_3_min            : num  -285 -380.1 -204 -73.5 -128.1 ...
##  $ timbre_3_max            : num  259 384 251 373 389 ...
##  $ timbre_4_min            : num  -40.4 -48.7 -66 -55.6 -43.9 ...
##  $ timbre_4_max            : num  73.6 100.4 152.1 119.2 99.3 ...
##  $ timbre_5_min            : num  -104.7 -87.3 -98.7 -77.5 -96.1 ...
##  $ timbre_5_max            : num  183.1 42.8 141.4 141.2 38.3 ...
##  $ timbre_6_min            : num  -88.8 -86.9 -88.9 -70.8 -110.8 ...
##  $ timbre_6_max            : num  73.5 75.5 66.5 64.5 72.4 ...
##  $ timbre_7_min            : num  -71.1 -65.8 -67.4 -63.7 -55.9 ...
##  $ timbre_7_max            : num  82.5 106.9 80.6 96.7 110.3 ...
##  $ timbre_8_min            : num  -52 -61.3 -59.8 -78.7 -56.5 ...
##  $ timbre_8_max            : num  39.1 35.4 46 41.1 37.6 ...
##  $ timbre_9_min            : num  -35.4 -81.9 -46.3 -49.2 -48.6 ...
##  $ timbre_9_max            : num  71.6 74.6 59.9 95.4 67.6 ...
##  $ timbre_10_min           : num  -126.4 -103.8 -108.3 -102.7 -52.8 ...
##  $ timbre_10_max           : num  18.7 121.9 33.3 46.4 22.9 ...
```

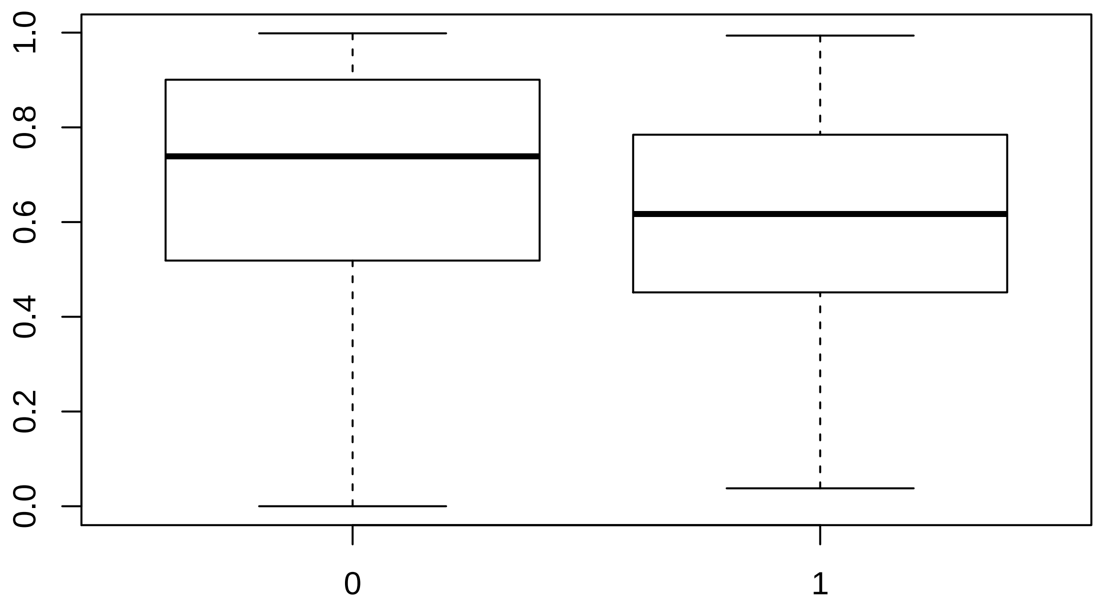
```
##  $ timbre_11_min          : num  -44.8 -38.9 -43.7 -59.4 -50.4 ...
##  $ timbre_11_max          : num  26 22.5 25.7 37.1 32.8 ...
##  $ Top10                  : int  0 0 0 0 0 0 0 0 0 1 ...
```

```r
#look at the dependent varible i.e. Top10
table(songs$Top10)
```

```
##
##    0    1
## 6455 1119
```

There are 6455 unpopular songs and 1119 popular songs.

```r
#comparison between the energy of popular and unpop songs
boxplot(energy~ Top10, data= songs)
```




```r
en_pop<- songs$energy[songs$Top10==1]
en_unpop<- songs$energy[songs$Top10==0]
t.test(en_pop, en_unpop)
```

```
##
##  Welch Two Sample t-test
##
## data:  en_pop and en_unpop
## t = -11.266, df = 1667.9, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.09424164 -0.06629374
## sample estimates:
```

```
## mean of x mean of y
## 0.6070623 0.6873300
```

```r
#pitch of pop and unpop songs
mean(songs$pitch[songs$Top10==0])
```

```
## [1] 0.01160031
```

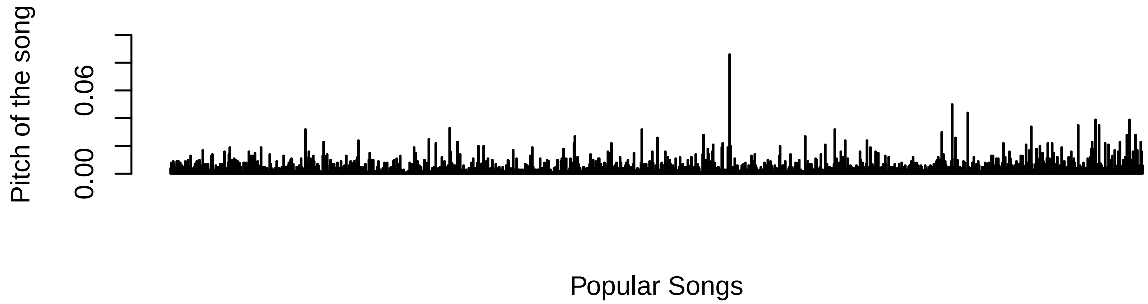
```r
mean(songs$pitch[songs$Top10==1])
```

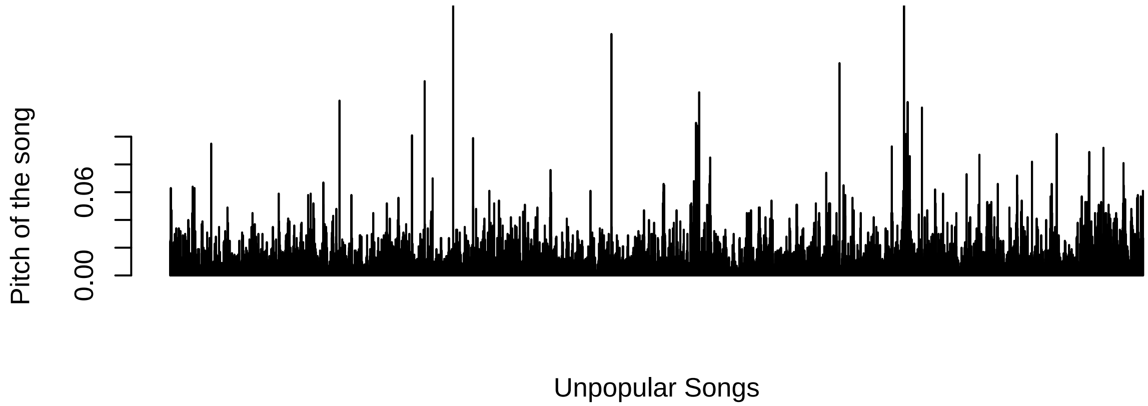
```
## [1] 0.006298481
```

```r
t.test(songs$pitch[songs$Top10==0], songs$pitch[songs$Top10==1]) #significant
```

```
##
##  Welch Two Sample t-test
##
## data:  songs$pitch[songs$Top10 == 0] and songs$pitch[songs$Top10 == 1]
## t = 20.25, df = 3506.1, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.004788495 0.005815163
## sample estimates:
##   mean of x   mean of y
## 0.011600310 0.006298481
```

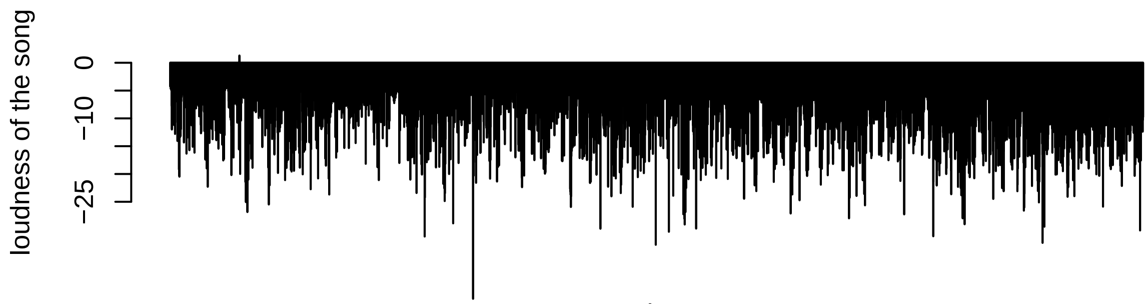
```r
#comparing the pop and umpop songs' pitch with a barplot
layout(matrix(c(1,1,2,2),2,2,byrow=T))
barplot(songs$pitch[songs$Top10==1], ylim = c(0,0.1), ylab = "Pitch of the song",
     xlab="Popular Songs")
barplot(songs$pitch[songs$Top10==0], ylim = c(0,0.1), ylab = "Pitch of the song",
     xlab="Unpopular Songs")
```






```
#comparing loudness of pop and unpop songs
layout(matrix(c(1,1,2,2),2,2,byrow=T))
barplot(songs$loudness[songs$Top10==1], ylab = "loudness of the song",
         xlab="Popular Songs")
barplot(songs$loudness[songs$Top10==0], ylab = "loudness of the song",
    xlab="Unpopular Songs", ylim = c(-25, 0))#since limit of the popular songs is -25 to 0
```

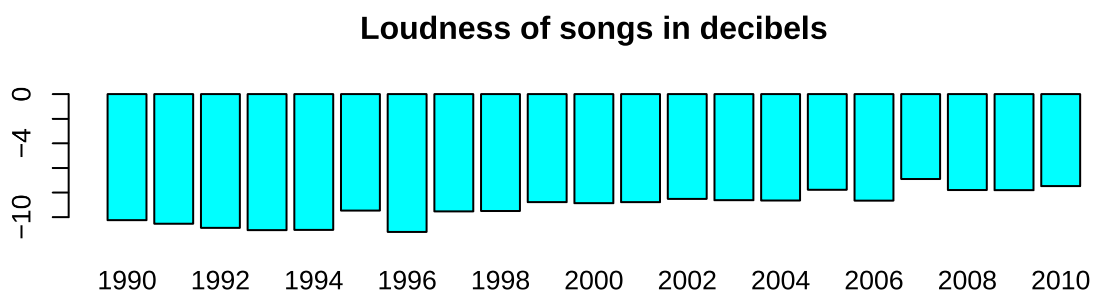
Popular Songs



Unpopular Songs

```
#loudness of songs overtime
x<- 1990:2010
x<- as.character(x)
m<- aggregate(loudness~year, data=songs, mean)
m<- as.data.frame(m)
barplot(m[,2], names.arg = x, col="cyan", main="Loudness of songs in decibels")

#loudness increases overtime
```




```r
#how are all variables correlated to one another?
songs1<- songs[-(1:7)]
names(songs1)
```

```
##  [1] "loudness"         "tempo"            "tempo_confidence"
##  [4] "key"              "key_confidence"   "energy"
##  [7] "pitch"            "timbre_0_min"     "timbre_0_max"
## [10] "timbre_1_min"     "timbre_1_max"     "timbre_2_min"
## [13] "timbre_2_max"     "timbre_3_min"     "timbre_3_max"
## [16] "timbre_4_min"     "timbre_4_max"     "timbre_5_min"
## [19] "timbre_5_max"     "timbre_6_min"     "timbre_6_max"
## [22] "timbre_7_min"     "timbre_7_max"     "timbre_8_min"
## [25] "timbre_8_max"     "timbre_9_min"     "timbre_9_max"
## [28] "timbre_10_min"    "timbre_10_max"    "timbre_11_min"
## [31] "timbre_11_max"    "Top10"
```

```r
#we should remove timbre_0_min to timbre_11_max as it doesn't seem important
songs1<- songs1[-(8:31)]
head(songs1,3)
```

```
##   loudness   tempo tempo_confidence key key_confidence    energy pitch
## 1   -4.262  91.525            0.953  11          0.453 0.9666556 0.024
## 2   -4.051 140.048            0.921  10          0.469 0.9847095 0.025
## 3   -3.571 160.512            0.489   2          0.209 0.9899004 0.026
##   Top10
## 1     0
## 2     0
```

```
## 3     0
```

```r
cor(songs1[, 1:8])
```

```
##                      loudness        tempo tempo_confidence          key
## loudness          1.000000000  0.052429357       0.13963892 -0.007502873
## tempo             0.052429357  1.000000000      -0.02298595  0.013343567
## tempo_confidence  0.139638920 -0.022985953       1.00000000  0.016604819
## key              -0.007502873  0.013343567       0.01660482  1.000000000
## key_confidence    0.018723570  0.071195471      -0.01282969 -0.043337193
## energy            0.741991823  0.155757810       0.14847670  0.009073184
## pitch             0.060418284  0.040799063      -0.05475646  0.010346962
## Top10            -0.087648652 -0.002544598       0.08485215  0.029124759
##                  key_confidence       energy       pitch        Top10
## loudness             0.01872357  0.741991823  0.06041828 -0.087648652
## tempo                0.07119547  0.155757810  0.04079906 -0.002544598
## tempo_confidence    -0.01282969  0.148476702 -0.05475646  0.084852154
## key                 -0.04333719  0.009073184  0.01034696  0.029124759
## key_confidence       1.00000000 -0.052696572 -0.08244120  0.010182457
## energy              -0.05269657  1.000000000  0.31351604 -0.116992015
## pitch               -0.08244120  0.313516037  1.00000000 -0.137622469
## Top10                0.01018246 -0.116992015 -0.13762247  1.000000000
```

splitting into a test and training set and fitting logistic reg models

```r
#install.packages("caTools")
library(caTools)
```

```
## Warning: package 'caTools' was built under R version 3.3.3
```

```r
set.seed(888)
split<- sample.split(songs$Top10, SplitRatio = 0.75)
#makes sure that the outcome variable is well balanced in both the sets

train<- subset(songs, split== TRUE)
head(train)
dim(train)
test<- subset(songs, split==FALSE)
```

```r
#fitting model
mod1<- glm(Top10 ~pitch, data= train, family = "binomial" )
summary(mod1)
```

```
##
## Call:
## glm(formula = Top10 ~ pitch, family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7396  -0.6503  -0.5329  -0.3284   3.8418
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.15664    0.05597  -20.67   <2e-16 ***
## pitch       -72.35230    6.18077  -11.71   <2e-16 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4756.6  on 5679  degrees of freedom
## Residual deviance: 4553.8  on 5678  degrees of freedom
## AIC: 4557.8
##
## Number of Fisher Scoring iterations: 6
```

```r
pred1<- predict( mod1, type="response")
table(train$Top10, pred1 >=0.6)
```

```
##
##     FALSE
##   0  4841
##   1   839
```

```r
mod2<- glm(Top10~ tempo+key+energy+pitch, data=train, family = "binomial")
summary(mod2)
```

```
##
## Call:
## glm(formula = Top10 ~ tempo + key + energy + pitch, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8626  -0.6412  -0.5309  -0.3325   3.6358
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -1.224350   0.194629  -6.291 3.16e-10 ***
## tempo         0.001830   0.001594   1.148  0.25113
## key           0.028448   0.010723   2.653  0.00798 **
## energy       -0.518756   0.168831  -3.073  0.00212 **
## pitch       -65.957343   6.563372 -10.049  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 4756.6  on 5679  degrees of freedom
## Residual deviance: 4537.0  on 5675  degrees of freedom
## AIC: 4547
##
## Number of Fisher Scoring iterations: 6
```

```r
pred2<- predict( mod1, type="response")
table(train$Top10, pred2 >=0.3)
```

```
##
##     FALSE
##   0  4841
##   1   839
```

```r
table(train$Top10, pred2 >=0.4)
```

```
##
##     FALSE
##   0  4841
##   1   839
```

```r
table(train$Top10, pred2 >=0.5)
```

```
##
##     FALSE
##   0  4841
##   1   839
```

```r
table(train$Top10, pred2 >=0.6)
```

```
##
##     FALSE
##   0  4841
##   1   839
```

For a cutoff prob of 0.3 or greater the model gives the same output as the baseline model. It seems that logistic regression is not a good method of predicting popularity. The variables are also not correlated with top10 variable. It would be safe to conclude that logistics regression is not an apt method for this. We should try some other models like decision trees or maybe classifiers or maybe Text analysis. It would be very interesting to see what machine learning alogorithm would give us the perfect model.