SVM Classification

Jered Hightower, Haniyyah Hamid

https://www.kaggle.com/datasets/vicsuperman/prediction-of-music-genre

```
original <- read.csv("music_genre.csv")
original$key <- factor(original$key)
original$tempo <- as.numeric(original$tempo)
```

```
## Warning: NAs introduced by coercion
```

```
original$mode <- factor(original$mode)
original$music_genre <- factor(original$music_genre)

df <- original[, -c(1,2,3,7,8,16)]

df <- df[complete.cases(df),]

df$key <- droplevels(df$key)
df$mode <- droplevels(df$mode)
df$music_genre <- droplevels(df$music_genre)

str(df)
```

```
## 'data.frame':    45020 obs. of  12 variables:
##  $ popularity      : num  27 31 28 34 32 46 43 39 22 30 ...
##  $ acousticness    : num  0.00468 0.0127 0.00306 0.0254 0.00465 0.0289 0.0297 0.00299 0.00934 0.855
##  $ danceability    : num  0.652 0.622 0.62 0.774 0.638 0.572 0.809 0.509 0.578 0.607 ...
##  $ instrumentalness: num  7.92e-01 9.50e-01 1.18e-02 2.53e-03 9.09e-01 7.74e-06 9.03e-01 2.76e-04 1.
##  $ key             : Factor w/ 12 levels "A","A#","B","C",..: 2 6 12 5 10 3 11 9 1 10 ...
##  $ liveness        : num  0.115 0.124 0.534 0.157 0.157 0.106 0.0635 0.178 0.111 0.106 ...
##  $ loudness        : num  -5.2 -7.04 -4.62 -4.5 -6.27 ...
##  $ mode            : Factor w/ 2 levels "Major","Minor": 2 2 1 1 1 1 2 2 2 2 ...
##  $ speechiness     : num  0.0748 0.03 0.0345 0.239 0.0413 0.351 0.0484 0.268 0.173 0.0345 ...
##  $ tempo           : num  101 115 128 128 145 ...
##  $ valence         : num  0.759 0.531 0.333 0.27 0.323 0.23 0.761 0.273 0.203 0.307 ...
##  $ music_genre     : Factor w/ 10 levels "Alternative",..: 6 6 6 6 6 6 6 6 6 6 ...
```

Train, test, validate

```
set.seed(1234)
spec <- c(train=.6, test=.2, validate=.2)
i <- sample(cut(1:nrow(df), nrow(df) * cumsum(c(0, spec)), labels=names(spec)))

train <- df[i=="train",]
test <- df[i=="test",]
vald <- df[i=="validate",]
```

Data Exploration

```
# How is genre associated with key?

# How often each genre appears
round(table(train$music_genre)/nrow(train), 2)
```

```
##
## Alternative       Anime       Blues   Classical     Country  Electronic
##         0.1         0.1         0.1         0.1         0.1         0.1
##     Hip-Hop        Jazz         Rap        Rock
##         0.1         0.1         0.1         0.1
```

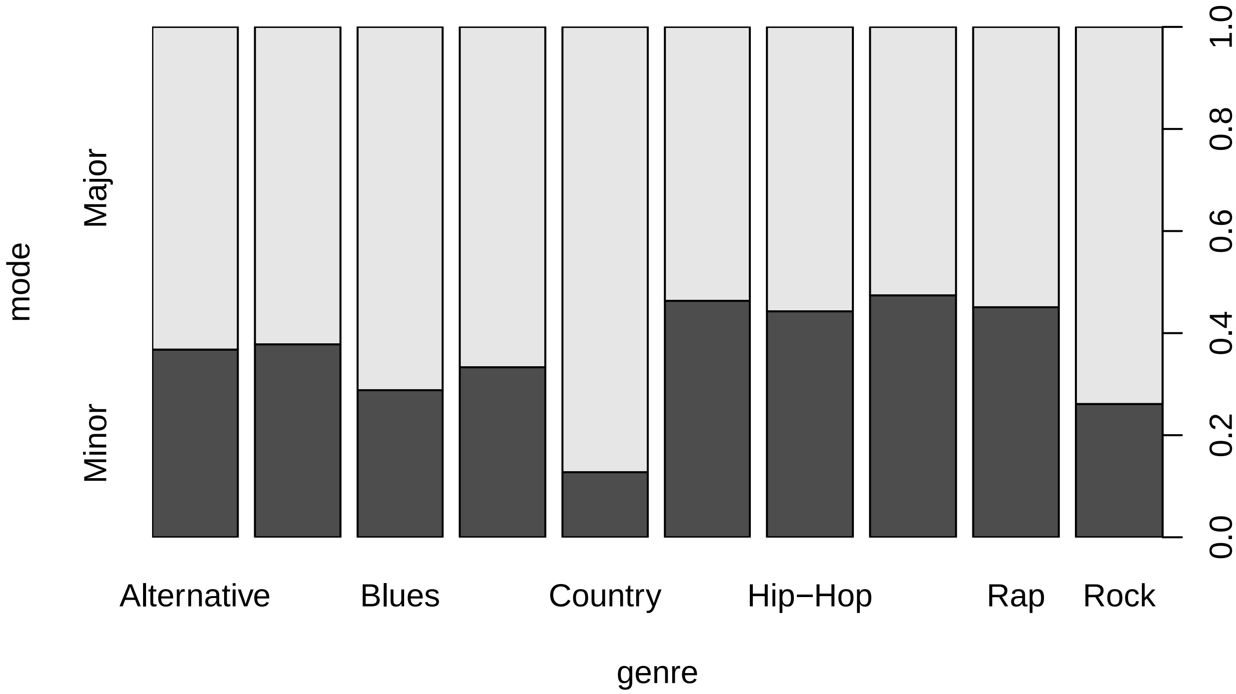
```
# Proportion of Genre that is in a specific key
tr <- table(train$music_genre, train$key)
prop <- prop.table(tr, margin = 1)
round(prop, 2)
```

```
##
##                  A   A#    B    C   C#    D   D#    E    F   F#    G   G#
##   Alternative 0.10 0.05 0.09 0.11 0.10 0.10 0.03 0.08 0.09 0.07 0.11 0.06
##   Anime       0.09 0.05 0.07 0.13 0.10 0.11 0.04 0.08 0.09 0.07 0.11 0.07
##   Blues       0.13 0.05 0.07 0.13 0.06 0.13 0.02 0.09 0.09 0.04 0.14 0.05
##   Classical   0.09 0.07 0.05 0.12 0.08 0.12 0.06 0.09 0.09 0.05 0.12 0.06
##   Country     0.10 0.05 0.07 0.11 0.07 0.12 0.04 0.10 0.07 0.06 0.14 0.06
##   Electronic  0.09 0.08 0.09 0.10 0.14 0.09 0.02 0.07 0.08 0.08 0.11 0.07
##   Hip-Hop     0.08 0.08 0.09 0.09 0.18 0.08 0.02 0.05 0.09 0.07 0.08 0.09
##   Jazz        0.09 0.10 0.06 0.11 0.09 0.09 0.04 0.07 0.12 0.05 0.11 0.06
##   Rap         0.07 0.08 0.09 0.09 0.18 0.08 0.02 0.05 0.08 0.07 0.09 0.09
##   Rock        0.12 0.05 0.07 0.13 0.08 0.13 0.03 0.10 0.07 0.06 0.12 0.05
```

Plotting

```
# Are different modes more common depending on genre?

# Alternative, Anime, Blues, Classical, Country, Electronic, Hip-Hop, Jazz, Rap, Rock
plot(df$music_genre, df$mode, xlab = "genre", ylab = "mode")
```



Logistic Regression Baseline

```
library(nnet)
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(mltools)
```

```
##
## Attaching package: 'mltools'
##
## The following object is masked from 'package:tidyr':
##
##     replace_na
```

```
model <- multinom(music_genre~., data = train)
```

```
## # weights:  230 (198 variable)
## initial  value 62197.428532
## iter  10 value 51717.181935
## iter  20 value 50477.724126
## iter  30 value 43379.658162
## iter  40 value 38351.479896
## iter  50 value 35686.916192
## iter  60 value 35045.609505
```

```
## iter  70 value 34736.352353
## iter  80 value 34576.333057
## iter  90 value 34458.453953
## iter 100 value 34375.597625
## final  value 34375.597625
## stopped after 100 iterations
```

```
summary(model)
```

```
## Call:
## multinom(formula = music_genre ~ ., data = train)
##
## Coefficients:
##            (Intercept)  popularity acousticness danceability instrumentalness
## Anime       10.9991015 -0.26968907     1.440888    -3.595187        1.8377885
## Blues        6.2913048 -0.17751213     1.129768    -3.288870       -0.6730619
## Classical    6.0100433 -0.19350771     3.380680    -8.220032        1.5230155
## Country      0.9245176 -0.05464995     1.312060     1.476015       -6.6427225
## Electronic   2.0677059 -0.13620692    -1.400947     5.672780        2.9519777
## Hip-Hop    -12.7776238  0.11739779    -0.269281     8.787975       -2.2685837
## Jazz         1.3561301 -0.10696628     2.652410     1.243188        2.3139803
## Rap        -13.8580281  0.14787477    -0.367548     8.086549       -2.2237135
## Rock        -7.8205152  0.13974316    -0.458678    -1.524760       -0.5477955
##                  keyA#        keyB       keyC      keyC#        keyD
## Anime      -0.02888737  0.05803174  0.3201543  0.2779544  0.20930891
## Blues      -0.34183188 -0.11017521 -0.1550807 -0.5386692  0.13055649
## Classical   0.24060059  0.20373823  0.2212570  0.4067553  0.32801399
## Country     0.14687307  0.09358805 -0.2839081 -0.1179857  0.15013357
## Electronic  0.18445480 -0.26871000 -0.0119110  0.2292913 -0.19046323
## Hip-Hop     0.32161631 -0.12414624  0.1211809  0.3283413 -0.16384972
## Jazz        0.69941814  0.05517951  0.2717533  0.3675372  0.28415644
## Rap         0.27416198 -0.08721147  0.1111293  0.3185423 -0.13489991
## Rock       -0.13906596 -0.22448920 -0.1320520 -0.3342903 -0.04488417
##                  keyD#         keyE        keyF       keyF#        keyG
## Anime       0.20366744  0.066295885 -0.05405412  0.12604678  0.08735173
## Blues      -0.57670112  0.004411721 -0.43028569 -0.52627899 -0.08258864
## Classical   0.41786330  0.528906901 -0.06714290 -0.11912635  0.25524475
## Country     0.10673016  0.469606956 -0.22950571  0.09590677  0.07916784
## Electronic -0.22464529 -0.322497799 -0.10266996  0.04493928 -0.05774333
## Hip-Hop    -0.36567917 -0.244336214  0.09606005  0.19414256 -0.12584664
## Jazz        0.35857598  0.085160217  0.36032752  0.03435282  0.23013795
## Rap        -0.04053238 -0.189371523 -0.04192283  0.13214857 -0.04876882
## Rock       -0.16418000  0.171440729 -0.37298752 -0.37380916 -0.26695069
##                  keyG#   liveness     loudness    modeMinor speechiness
## Anime       0.34017492 -0.8871541  0.020165738  0.246312772  -5.4562816
## Blues      -0.27800390  0.9865994 -0.168113471 -0.151266790  -4.6469496
## Classical   0.35046223  0.3681981 -0.250136352  0.001758408  -1.8843959
## Country    -0.08636558  0.2293075 -0.027530620 -1.225748411 -13.3433031
## Electronic  0.24336724  0.4140598  0.008179687  0.451787903   1.2585839
## Hip-Hop     0.46832370  0.6808609 -0.012153883  0.218552888   8.0624111
## Jazz        0.56663411 -0.1255640 -0.112597729  0.611110975  -0.2439643
## Rap         0.41841153  0.6626609 -0.019021995  0.289662875   6.9974791
## Rock       -0.21077703  0.2523448 -0.112398618 -0.423355601 -10.7970638
##                   tempo   valence
## Anime       0.003943744  1.350452
```

```
## Blues      -0.002484066  4.157257
## Classical  -0.003340875  2.341676
## Country     0.006373046  1.566040
## Electronic  0.007945948 -2.373167
## Hip-Hop     0.002526266 -1.688547
## Jazz       -0.008711052  2.254845
## Rap         0.002814098 -1.942823
## Rock        0.002236397  1.944815
##
## Std. Errors:
##            (Intercept)  popularity acousticness danceability instrumentalness
## Anime        0.3195106 0.004227117    0.1595647    0.2847931        0.1474590
## Blues        0.2966277 0.003823592    0.1450703    0.2654298        0.1599405
## Classical    0.3669916 0.004503583    0.1894797    0.3603076        0.1598355
## Country      0.2849525 0.003349633    0.1355647    0.2459999        0.5547777
## Electronic   0.3012793 0.003767660    0.1746014    0.2547906        0.1356059
## Hip-Hop      0.3622873 0.004122232    0.1637813    0.2597556        0.3124610
## Jazz         0.3015935 0.003718225    0.1383302    0.2544564        0.1352582
## Rap          0.3653668 0.004170301    0.1634371    0.2576536        0.3089280
## Rock         0.3193617 0.003916902    0.1486716    0.2454163        0.1808198
##                keyA#      keyB      keyC     keyC#      keyD     keyD#
## Anime      0.1868633 0.1694329 0.1517026 0.1579774 0.1550540 0.2214891
## Blues      0.1695030 0.1508397 0.1351127 0.1500912 0.1356001 0.2169580
## Classical  0.2194857 0.2207853 0.1864241 0.2037305 0.1898733 0.2552628
## Country    0.1551082 0.1397450 0.1263241 0.1331151 0.1254794 0.1842437
## Electronic 0.1646023 0.1532276 0.1432624 0.1385274 0.1478143 0.2211417
## Hip-Hop    0.1656426 0.1532233 0.1461804 0.1389659 0.1527386 0.2380372
## Jazz       0.1600982 0.1556356 0.1419500 0.1468011 0.1465033 0.2000834
## Rap        0.1660183 0.1524649 0.1458115 0.1391595 0.1518260 0.2232048
## Rock       0.1610090 0.1406919 0.1267563 0.1359090 0.1283827 0.2006584
##                 keyE      keyF     keyF#      keyG     keyG#  liveness
## Anime      0.1687462 0.1614940 0.1756320 0.1507513 0.1792023 0.2233634
## Blues      0.1480307 0.1446830 0.1659779 0.1317330 0.1653506 0.1888084
## Classical  0.2023354 0.1956059 0.2362546 0.1839793 0.2256122 0.2654290
## Country    0.1354184 0.1372552 0.1451962 0.1217705 0.1485863 0.1926717
## Electronic 0.1583530 0.1501476 0.1559709 0.1404059 0.1615966 0.2002910
## Hip-Hop    0.1674630 0.1518766 0.1597921 0.1475036 0.1584146 0.2090910
## Jazz       0.1570387 0.1436985 0.1644785 0.1406478 0.1630207 0.2110471
## Rap        0.1649594 0.1535842 0.1606055 0.1462062 0.1589497 0.2105308
## Rock       0.1374589 0.1409918 0.1535209 0.1275028 0.1541986 0.2016929
##              loudness  modeMinor speechiness       tempo   valence
## Anime      0.01256110 0.07702190   0.5514557 0.001186027 0.1771606
## Blues      0.01120353 0.07234073   0.4762368 0.001095912 0.1653935
## Classical  0.01214570 0.09463149   0.6540115 0.001407846 0.2298579
## Country    0.01146831 0.07525639   0.7066560 0.001035723 0.1452305
## Electronic 0.01254939 0.06955645   0.3722552 0.001153281 0.1594569
## Hip-Hop    0.01320407 0.06960320   0.3067643 0.001117982 0.1562093
## Jazz       0.01094571 0.06824142   0.3856795 0.001123449 0.1584026
## Rap        0.01326581 0.06934091   0.3113952 0.001118989 0.1566212
## Rock       0.01159209 0.06851944   0.6403135 0.001047165 0.1504290
##
## Residual Deviance: 68751.2
## AIC: 69147.2
```

```r
pr <- model %>% predict(test)


acc_rf <- mean(pr==test$music_genre)
print(paste("accuracy=", acc_rf))
```

```
## [1] "accuracy= 0.528431808085295"
```

Linear SVM

```r
library(e1071)
```

```
##
## Attaching package: 'e1071'

## The following object is masked from 'package:mltools':
##
##     skewness
```

```r
svm1 <- svm(music_genre~., data=train, kernel="linear", cost=10, scale=TRUE)

summary(svm1)
```

```
##
## Call:
## svm(formula = music_genre ~ ., data = train, kernel = "linear", cost = 10,
##     scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  linear
##        cost:  10
##
## Number of Support Vectors:  22391
##
##  ( 1995 1956 2433 2679 2500 2639 2432 1945 1209 2603 )
##
##
## Number of Classes:  10
##
## Levels:
##  Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz Rap Rock
```

Evaluate

```r
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
pred <- predict(svm1, newdata=test)
caret:: confusionMatrix(as.factor(pred), reference=test$music_genre)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz
##   Alternative         346    28    35        25      81         65      62   28
##   Anime                 3   584   156        34      15         77       0   35
##   Blues                17    83   388        21      84         53       2  103
##   Classical             4   118    24       756       1         13       0   66
##   Country             144    40   103        11     443         38      20   63
##   Electronic           54    62    58        19      28        512      10  125
##   Hip-Hop              97     2     2         0      20         38     475   23
##   Jazz                 49    18    99        32      55        101      12  409
##   Rap                  22     0     1         0       5          9     238    1
##   Rock                148     5    44         5     180         28      60   31
##              Reference
## Prediction    Rap Rock
##   Alternative  57   95
##   Anime         0    2
##   Blues         1    2
##   Classical     0    3
##   Country      11   65
##   Electronic    3   10
##   Hip-Hop     362   18
##   Jazz          8   35
##   Rap         290   40
##   Rock        103  653
##
## Overall Statistics
##
##                Accuracy : 0.5393
##                  95% CI : (0.529, 0.5497)
##     No Information Rate : 0.1044
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4879
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Alternative Class: Anime Class: Blues
## Sensitivity                     0.39140      0.62128      0.42637
## Specificity                     0.94138      0.96007      0.95478
## Pos Pred Value                  0.42092      0.64459      0.51459
## Neg Pred Value                  0.93425      0.95604      0.93673
## Prevalence                      0.09818      0.10440      0.10107
## Detection Rate                  0.03843      0.06486      0.04309
## Detection Prevalence            0.09129      0.10062      0.08374
## Balanced Accuracy               0.66639      0.79067      0.69058
##                      Class: Classical Class: Country Class: Electronic
## Sensitivity                   0.83721         0.4857           0.54818
```

```
## Specificity                 0.97173        0.9388          0.95428
## Pos Pred Value              0.76751        0.4723          0.58116
## Neg Pred Value              0.98167        0.9419          0.94805
## Prevalence                  0.10029        0.1013          0.10373
## Detection Rate              0.08396        0.0492          0.05686
## Detection Prevalence        0.10940        0.1042          0.09785
## Balanced Accuracy           0.90447        0.7123          0.75123
##                      Class: Hip-Hop Class: Jazz Class: Rap Class: Rock
## Sensitivity                 0.54039     0.46267    0.34731     0.70748
## Specificity                 0.93083     0.94963    0.96132     0.92526
## Pos Pred Value              0.45805     0.50000    0.47855     0.51949
## Neg Pred Value              0.94929     0.94197    0.93510     0.96515
## Prevalence                  0.09762     0.09818    0.09274     0.10251
## Detection Rate              0.05275     0.04542    0.03221     0.07252
## Detection Prevalence        0.11517     0.09085    0.06730     0.13960
## Balanced Accuracy           0.73561     0.70615    0.65431     0.81637
```

Tune

```
tune_svm1 <- tune(svm, music_genre~., data=vald, kernel="linear", ranges = list(cost=c(.001, .01, .1, 1
summary(tune_svm1)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost
##     5
##
## - best performance: 0.4749032
##
## - Detailed performance results:
##    cost     error dispersion
## 1 1e-03 0.5442041 0.01441899
## 2 1e-02 0.4827868 0.01342398
## 3 1e-01 0.4754578 0.01273078
## 4 1e+00 0.4759026 0.01192062
## 5 5e+00 0.4749032 0.01360166
## 6 1e+01 0.4751258 0.01424545
## 7 1e+02 0.4760139 0.01405247
```

Evaluate on best linear svm

The best linear svm happens to be the one we first used. We already found the optimal cost of 10. No need to rerun model.

Try Polynomial Kernel

```
svm2 <- svm(music_genre~., data = train, kernel="polynomial", cost = 10, scale = TRUE)
summary(svm2)
```

```
##
```

```
## Call:
## svm(formula = music_genre ~ ., data = train, kernel = "polynomial",
##     cost = 10, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  polynomial
##        cost:  10
##      degree:  3
##      coef.0:  0
##
## Number of Support Vectors:  22014
##
##  ( 1966 1713 2354 2661 2552 2643 2323 2147 1015 2640 )
##
##
## Number of Classes:  10
##
## Levels:
##  Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz Rap Rock
```

Evaluate

```
pred2 <- predict(svm2, newdata=test)
caret:: confusionMatrix(as.factor(pred2), reference=test$music_genre)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz
##   Alternative         375    33    52        29      79         93      88   53
##   Anime                 3   613    69        36      17         44       0   16
##   Blues                15    83   466        26      66         59       0  120
##   Classical             2    72    13       732       0          3       0   42
##   Country             177    60   135        12     512         70      26   92
##   Electronic           39    48    58        16      17        509       9  114
##   Hip-Hop              86     2     1         0      15         32     493   16
##   Jazz                 36    25    77        48      48         93       7  405
##   Rap                  21     0     2         0       4          9     209    3
##   Rock                130     4    37         4     154         22      47   23
##              Reference
## Prediction    Rap Rock
##   Alternative  87  150
##   Anime         0    1
##   Blues         1    4
##   Classical     0    1
##   Country      23  115
##   Electronic    1    6
##   Hip-Hop     400   21
##   Jazz          4   28
##   Rap         215   21
##   Rock        104  576
##
```

```
## Overall Statistics
##
##                Accuracy : 0.5438
##                  95% CI : (0.5334, 0.5541)
##     No Information Rate : 0.1044
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4929
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Alternative Class: Anime Class: Blues
## Sensitivity                     0.42421      0.65213      0.51209
## Specificity                     0.91823      0.97693      0.95379
## Pos Pred Value                  0.36092      0.76721      0.55476
## Neg Pred Value                  0.93610      0.96015      0.94561
## Prevalence                      0.09818      0.10440      0.10107
## Detection Rate                  0.04165      0.06808      0.05175
## Detection Prevalence            0.11539      0.08874      0.09329
## Balanced Accuracy               0.67122      0.81453      0.73294
##                      Class: Classical Class: Country Class: Electronic
## Sensitivity                   0.81063        0.56140           0.54497
## Specificity                   0.98358        0.91226           0.96183
## Pos Pred Value                0.84624        0.41899           0.62301
## Neg Pred Value                0.97899        0.94860           0.94809
## Prevalence                    0.10029        0.10129           0.10373
## Detection Rate                0.08130        0.05686           0.05653
## Detection Prevalence          0.09607        0.13572           0.09074
## Balanced Accuracy             0.89711        0.73683           0.75340
##                      Class: Hip-Hop Class: Jazz Class: Rap Class: Rock
## Sensitivity                 0.56086     0.45814    0.25749     0.62405
## Specificity                 0.92948     0.95493    0.96707     0.93503
## Pos Pred Value              0.46248     0.52529    0.44421     0.52316
## Neg Pred Value              0.95137     0.94182    0.92723     0.95609
## Prevalence                  0.09762     0.09818    0.09274     0.10251
## Detection Rate              0.05475     0.04498    0.02388     0.06397
## Detection Prevalence        0.11839     0.08563    0.05375     0.12228
## Balanced Accuracy           0.74517     0.70654    0.61228     0.77954
```

Tune hyperparameters

```
tune.poly <- tune(svm, music_genre~., data=vald, kernel="polynomial", ranges = list(cost=c(.1, 1, 5, 10
summary(tune.poly)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost degree
##    10      3
```

```
##
## - best performance: 0.4894477
##
## - Detailed performance results:
##     cost degree     error  dispersion
## 1    0.1      3 0.7544405 0.012601614
## 2    1.0      3 0.5665263 0.019669464
## 3    5.0      3 0.4998872 0.021821734
## 4   10.0      3 0.4894477 0.020303319
## 5  100.0      3 0.5007769 0.015043959
## 6    0.1      4 0.8004192 0.009563913
## 7    1.0      4 0.6946866 0.020729655
## 8    5.0      4 0.6152809 0.020192899
## 9   10.0      4 0.5810745 0.021321885
## 10 100.0      4 0.5239889 0.013427405
## 11   0.1      5 0.8113036 0.010019974
## 12   1.0      5 0.7578832 0.010862062
## 13   5.0      5 0.6758113 0.026879680
## 14  10.0      5 0.6550422 0.018505138
## 15 100.0      5 0.5697468 0.020562180
```

Evaluate on best polynomial svm

The best polynomial svm also happens to be the one we first used. Cost = 10, Degree = 3, Coef.0 = 0. We already found the optimal values. No need to rerun model.

Try a radial kernel

```
svm3 <- svm(music_genre~., data = train, kernel = "radial", cost=10, gamma=1, scale=TRUE)
summary(svm3)
```

```
##
## Call:
## svm(formula = music_genre ~ ., data = train, kernel = "radial", cost = 10,
##     gamma = 1, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  10
##
## Number of Support Vectors:  26211
##
##  ( 2592 2525 2707 2717 2561 2704 2638 2614 2385 2768 )
##
##
## Number of Classes:  10
##
## Levels:
##  Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz Rap Rock
```

Evaluate

```
pred4 <- predict(svm3, newdata=test)
caret:: confusionMatrix(as.factor(pred4), reference=test$music_genre)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz
##   Alternative         276    27    38        26     124         61      58   51
##   Anime                18   601    89        52      43         59       1   19
##   Blues                27    86   431        24     100         63       2  108
##   Classical            10    69    19       681       7          4       0   42
##   Country              85    31    54         4     325         31      15   34
##   Electronic           81    65    86        16      39        502      14  116
##   Hip-Hop              79     4     1         0      26         26     282   20
##   Jazz                 76    53   149        94      86        151      48  465
##   Rap                  69     0     3         0      27         17     421    6
##   Rock                163     4    40         6     135         20      38   23
##              Reference
## Prediction    Rap Rock
##   Alternative  61  192
##   Anime         1    6
##   Blues         4   46
##   Classical     0    6
##   Country      16  134
##   Electronic   18   18
##   Hip-Hop     401   37
##   Jazz         31   59
##   Rap         232   59
##   Rock         71  366
##
## Overall Statistics
##
##                Accuracy : 0.4621
##                  95% CI : (0.4518, 0.4725)
##     No Information Rate : 0.1044
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4024
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Alternative Class: Anime Class: Blues
## Sensitivity                     0.31222      0.63936      0.47363
## Specificity                     0.92143      0.96429      0.94317
## Pos Pred Value                  0.30197      0.67604      0.48373
## Neg Pred Value                  0.92485      0.95823      0.94096
## Prevalence                      0.09818      0.10440      0.10107
## Detection Rate                  0.03065      0.06675      0.04787
## Detection Prevalence            0.10151      0.09873      0.09896
## Balanced Accuracy               0.61682      0.80182      0.70840
```

```
##                      Class: Classical Class: Country Class: Electronic
## Sensitivity                   0.75415        0.35636           0.53747
## Specificity                   0.98062        0.95007           0.94387
## Pos Pred Value                0.81265        0.44582           0.52565
## Neg Pred Value                0.97281        0.92906           0.94633
## Prevalence                    0.10029        0.10129           0.10373
## Detection Rate                0.07563        0.03610           0.05575
## Detection Prevalence          0.09307        0.08096           0.10606
## Balanced Accuracy             0.86739        0.65322           0.74067
##                      Class: Hip-Hop Class: Jazz Class: Rap Class: Rock
## Sensitivity                 0.32082     0.52602    0.27784     0.39653
## Specificity                 0.92689     0.90800    0.92631     0.93813
## Pos Pred Value              0.32192     0.38366    0.27818     0.42263
## Neg Pred Value              0.92655     0.94623    0.92619     0.93156
## Prevalence                  0.09762     0.09818    0.09274     0.10251
## Detection Rate              0.03132     0.05164    0.02577     0.04065
## Detection Prevalence        0.09729     0.13461    0.09263     0.09618
## Balanced Accuracy           0.62386     0.71701    0.60208     0.66733
```

Tune hyperparameters

```r
tune.out <- tune(svm, music_genre~., data=vald, kernel="radial", ranges = list(cost=c(.1, 1, 10, 100, 10
summary(tune.out)
```

```
##
## Parameter tuning of 'svm':
##
## - sampling method: 10-fold cross validation
##
## - best parameters:
##  cost gamma
##     1   0.5
##
## - best performance: 0.4831204
##
## - Detailed performance results:
##     cost gamma     error  dispersion
## 1  1e-01   0.5 0.5462016 0.017023647
## 2  1e+00   0.5 0.4831204 0.017248388
## 3  1e+01   0.5 0.5232145 0.018308855
## 4  1e+02   0.5 0.5249900 0.017001825
## 5  1e+03   0.5 0.5245453 0.017681962
## 6  1e-01   1.0 0.8752859 0.026193153
## 7  1e+00   1.0 0.5350963 0.015421819
## 8  1e+01   1.0 0.5513127 0.015232419
## 9  1e+02   1.0 0.5517569 0.015036867
## 10 1e+03   1.0 0.5516458 0.014936143
## 11 1e-01   2.0 0.9021562 0.009041863
## 12 1e+00   2.0 0.6722574 0.028543695
## 13 1e+01   2.0 0.6612624 0.025161226
## 14 1e+02   2.0 0.6614845 0.025138648
## 15 1e+03   2.0 0.6615956 0.024964049
## 16 1e-01   3.0 0.9021562 0.009041863
## 17 1e+00   3.0 0.7773233 0.023529212
```

```
## 18 1e+01   3.0 0.7585516 0.021457614
## 19 1e+02   3.0 0.7588847 0.021549598
## 20 1e+03   3.0 0.7588847 0.021549598
## 21 1e-01   4.0 0.9021562 0.009041863
## 22 1e+00   4.0 0.8264176 0.025136507
## 23 1e+01   4.0 0.8045353 0.023661072
## 24 1e+02   4.0 0.8046464 0.023697963
## 25 1e+03   4.0 0.8046464 0.023697963
```

Evaluate on best radial svm

```
svm4 <- svm(music_genre~., data = train, kernel = "radial", cost=1, gamma=.5, scale=TRUE)
summary(svm4)
```

```
##
## Call:
## svm(formula = music_genre ~ ., data = train, kernel = "radial", cost = 1,
##     gamma = 0.5, scale = TRUE)
##
##
## Parameters:
##    SVM-Type:  C-classification
##  SVM-Kernel:  radial
##        cost:  1
##
## Number of Support Vectors:  24074
##
##  ( 2353 2119 2576 2673 2370 2682 2443 2321 1794 2743 )
##
##
## Number of Classes:  10
##
## Levels:
##  Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz Rap Rock
```

```
pred5 <- predict(svm4, newdata=test)
caret:: confusionMatrix(as.factor(pred5), reference=test$music_genre)
```

```
## Confusion Matrix and Statistics
##
##              Reference
## Prediction    Alternative Anime Blues Classical Country Electronic Hip-Hop Jazz
##   Alternative         337    28    39        28      86         74      43   49
##   Anime                12   654    94        39      20         48       0   22
##   Blues                20    74   483        19      78         63       0  106
##   Classical             1    69    14       749       2          8       0   51
##   Country             109    33    76         6     435         38      12   46
##   Electronic           50    48    52        12      26        518       9   99
##   Hip-Hop              92     2     1         0      28         35     393   18
##   Jazz                 55    26   106        47      67        111      14  461
##   Rap                  45     0     3         0       8         15     358    3
##   Rock                163     6    42         3     162         24      50   29
##              Reference
## Prediction    Rap Rock
##   Alternative  42  101
```

```
##   Anime         0    2
##   Blues         4   14
##   Classical     0    4
##   Country       5   84
##   Electronic    5    9
##   Hip-Hop     400   29
##   Jazz         10   47
##   Rap         274   46
##   Rock         95  587
##
## Overall Statistics
##
##                Accuracy : 0.5432
##                  95% CI : (0.5328, 0.5535)
##     No Information Rate : 0.1044
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4924
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: Alternative Class: Anime Class: Blues
## Sensitivity                     0.38122      0.69574      0.53077
## Specificity                     0.93966      0.97061      0.95330
## Pos Pred Value                  0.40750      0.73401      0.56098
## Neg Pred Value                  0.93311      0.96475      0.94756
## Prevalence                      0.09818      0.10440      0.10107
## Detection Rate                  0.03743      0.07263      0.05364
## Detection Prevalence            0.09185      0.09896      0.09562
## Balanced Accuracy               0.66044      0.83318      0.74203
##                      Class: Classical Class: Country Class: Electronic
## Sensitivity                   0.82946        0.47697           0.55460
## Specificity                   0.98161        0.94946           0.96159
## Pos Pred Value                0.83408        0.51540           0.62560
## Neg Pred Value                0.98100        0.94154           0.94912
## Prevalence                    0.10029        0.10129           0.10373
## Detection Rate                0.08319        0.04831           0.05753
## Detection Prevalence          0.09973        0.09374           0.09196
## Balanced Accuracy             0.90553        0.71321           0.75809
##                      Class: Hip-Hop Class: Jazz Class: Rap Class: Rock
## Sensitivity                 0.44710     0.52149    0.32814     0.63597
## Specificity                 0.92554     0.94052    0.94149     0.92897
## Pos Pred Value              0.39379     0.48835    0.36436     0.50560
## Neg Pred Value              0.93930     0.94752    0.93202     0.95716
## Prevalence                  0.09762     0.09818    0.09274     0.10251
## Detection Rate              0.04365     0.05120    0.03043     0.06519
## Detection Prevalence        0.11084     0.10484    0.08352     0.12894
## Balanced Accuracy           0.68632     0.73101    0.63481     0.78247
```

Analysis of Results

Of the 3 kernels used on this dataset, the polynomial kernel just barely outperformed radial and linear. The accuracy of all those models sat at just about .54 and their kappa values sat around .49. I suspect that these models all output similar values because there exists a very general linear relationship in the data. Because of how general the relationship is, when each kernel creates its decision boundaries for the data, we end up with similar results.