My Comprehensive Evaluation

A Comprehensive Evaluation Report
Presented to
The Statistics Faculty
Amherst College

In Partial Fulfillment
of the Requirements for the Degree
Bachelor of Arts
in
Statistics

Gabriel Young

February 2019

Table of Contents

Abstract

This project explores the supervised learning method of SVM(Support Vector Machine) Classification in the context of Voice Recognition. Voice Recognition refers to the ability to correctly identify the voice of an individual speaker from among many speakers' voice recordings and humans are generally adept at this. However, there are situations where voices need to be identified without the inherent biases that come from a human listener, for example in court cases when determining whether a potentially incriminating audio sample does or does not include the accused's voice. Currently there is a critical need for more accurate and objective voice recognition software and SVM Classing is one of the methods being investigated and used.

The project first makes use of the publicly available dataset voice.csv on kaggle.com as a basis for testing learning methods on spectrogram data of human voices to classify those voices by biological Sex. The project seeks to emulate the original experiment and results obtained by experimenter Kory Becker. After verifying the usefulness of those measures in predicting Sex using familiar Random Forest Classification, we test the unfamiliar SVM method on that data and achieve similar, even slightly improved results. Using her dataset as a template, we expand the experiment by bringing in new anonymous audio data from VoxForge and parsing its spectrogram measurements. The dataset is constructed this time with the intention of Voice Recognizing each case. Random Forest Classification and SVM Classification is performed on this new dataset and each attempts Voice Recognition.

The ultimate result of the project is confirmation that, while biological Sex is easily identified in the human voice using these methods, Voice Recognition is a thornier issue, and that although SVM performs better than Random Forest in the former experiment, it winds up being inferior in the latter, although reasons as to why that might be the case are, at this point, speculative.

Chapter 1: Introduction

Introduction to SVM

SVM Classification is a method of supervised learning that separates data into groups much like k-nearest neighbor. The training data is plotted in p-dimensional space where p is the number of variables and the algorithm then seeks a hyperplane (plane with dimension p-1) that divides the data points correctly into their classes while maintaining the largest margin of distance between itself and any of the points. This ensures a "maximum margin classifier", or a classifier that gives us the greatest chance of correctly classifying new test data based on our training data.

In essence, the points are plotted and the algorithm attempts to define a border halfway between the two groups, then new data is classified based on which side of the border it lands on (Figure 1).

The "Support Vector" part of the title refers to those observations that lie closest to the boundaries of the two groups, the points that have the most influence on the shape of the plane. If we expanded the training set by adding another point right on the line, that point would become a strong determining support vector of our line and would shift or even reshape the line. Points that fall well within a grouping do not influence the line as greatly and so are of less interest. The "Machine" part refers to machine learning.

For classification where the response variable is not binary, as in the case for Voice Classification among more than two individuals, several classifiers are trained in succession using a "one vs rest" approach such that the first classifier would be "Person A vs Not Person A", the second being "Person B vs Not Person B", etc. Think of it something along the same logical lines as nested If-Else statements. ()

For more complex data where the classifications are not always linearly seperable in p-dimensional space (Figure 2), the algorithm maps the whole set to a higher dimensional space and seeks a dividing hyperplane in that space (Figure 3). The hyperplane is then projected downward to the original dimensions in order to correctly

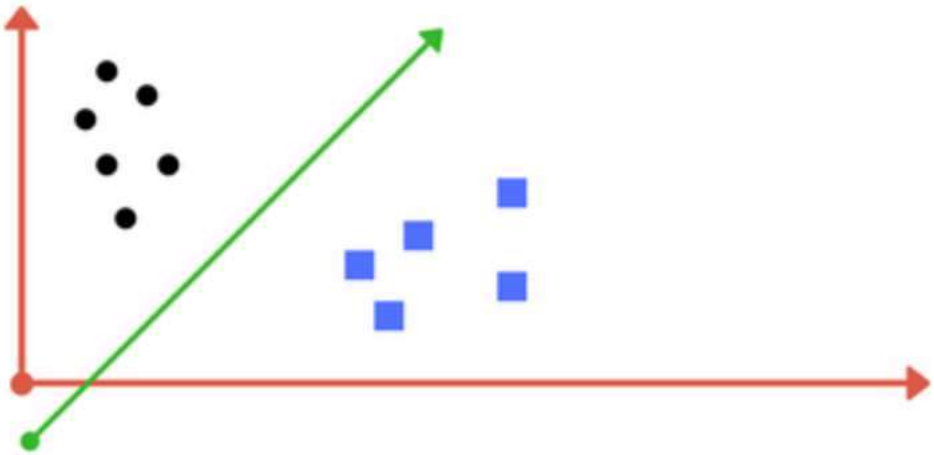


Figure 1: An SVM Split in 2D

class the data (Figure 4). This is known as the kernel trick.

These images are from the 1st source for this section in the Bibliography. The above information was sourced from the 2nd through 4th sources in that section.

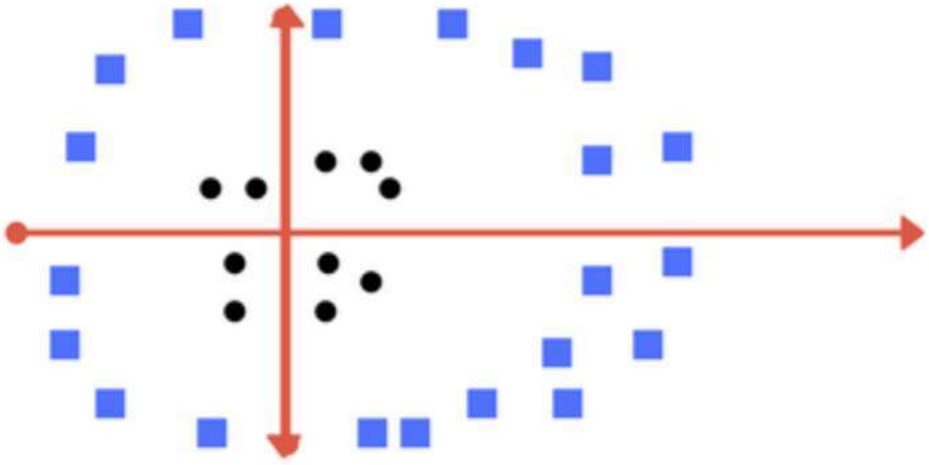


Figure 2: A More Complex Grouping That Cannot Be Linearly Split in 2D

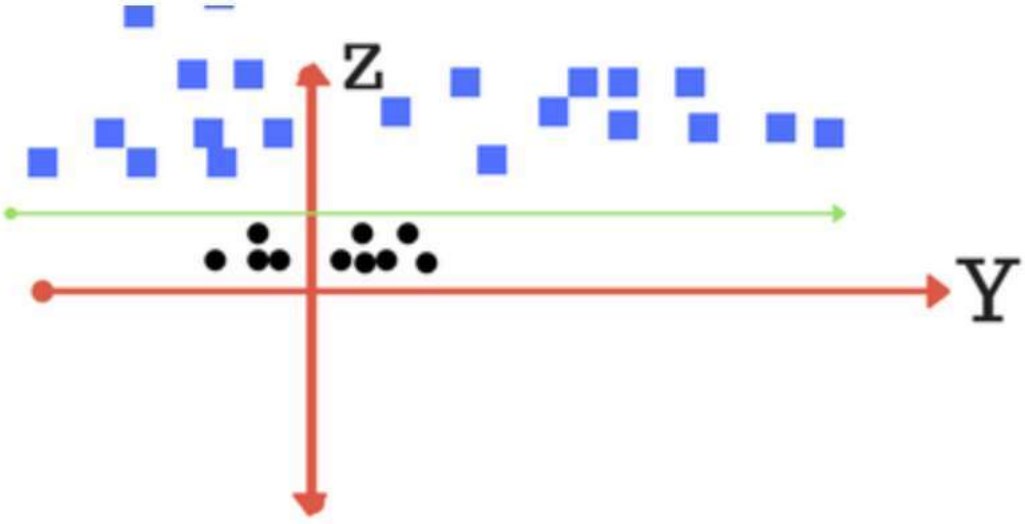


Figure 3: A Projection of the Previous Data Into a Higher Space and Fitted SVM Line

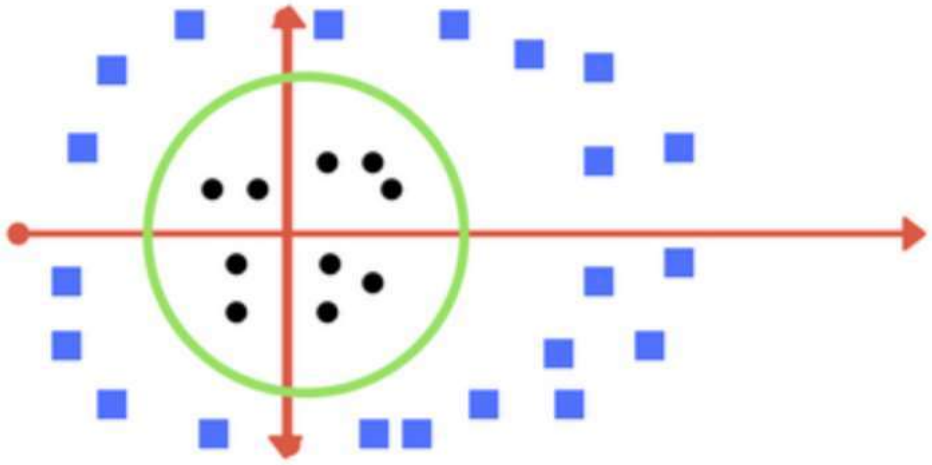


Figure 4: A Projection Back Downward of the Fitted Line

Introduction to Spectrograms

A spectrogram plots the frequency or amplitude of a sound wave over time, often with other information added in as well. Seewave's default spectrograms, shown here, represent the frequency and amplitude information over two graphs, but a full heat-map style of spectrogram might show frequency, time, amplitude, and intensity. If you have never seen a spectrogram before or would like a demonstration to experiment with, there's a free program in Google Chrome's Music Lab at `https://musiclab.chromeexperiments.com/Spectrogram/` . The concept makes a lot more sense if you see it in action.

Essentially, a spectrogram is a visual representation of a sound. Frequency is a measure of the pitch of a sound via the rapidity of its oscillations (in kHz) and amplitude is the volume or loudness of a sound (relative to a mean). These measures alone can be very useful in identifying sounds and even analyzing speech, since different phonemes (most basic unit sounds of a language) have different wave patterns when plotted. It is not a perfect representation of a sound, however. Tambre, the ill-defined notion of the "feel" of a sound or "that which differentiates two sounds of the same pitch and volume" (think how you'd recognize the same note played on a flute versus a violin) is less visible on these sorts of spectrograms, although it comes across moreso on heat-map style spectrograms like those in the Google demonstration, and it's not easy for people to reproduce a sound while looking at that sound's spectrogram. They do provide useful information for computerized analytics.

The above information is sourced from the 5th and 6th sources in that section.

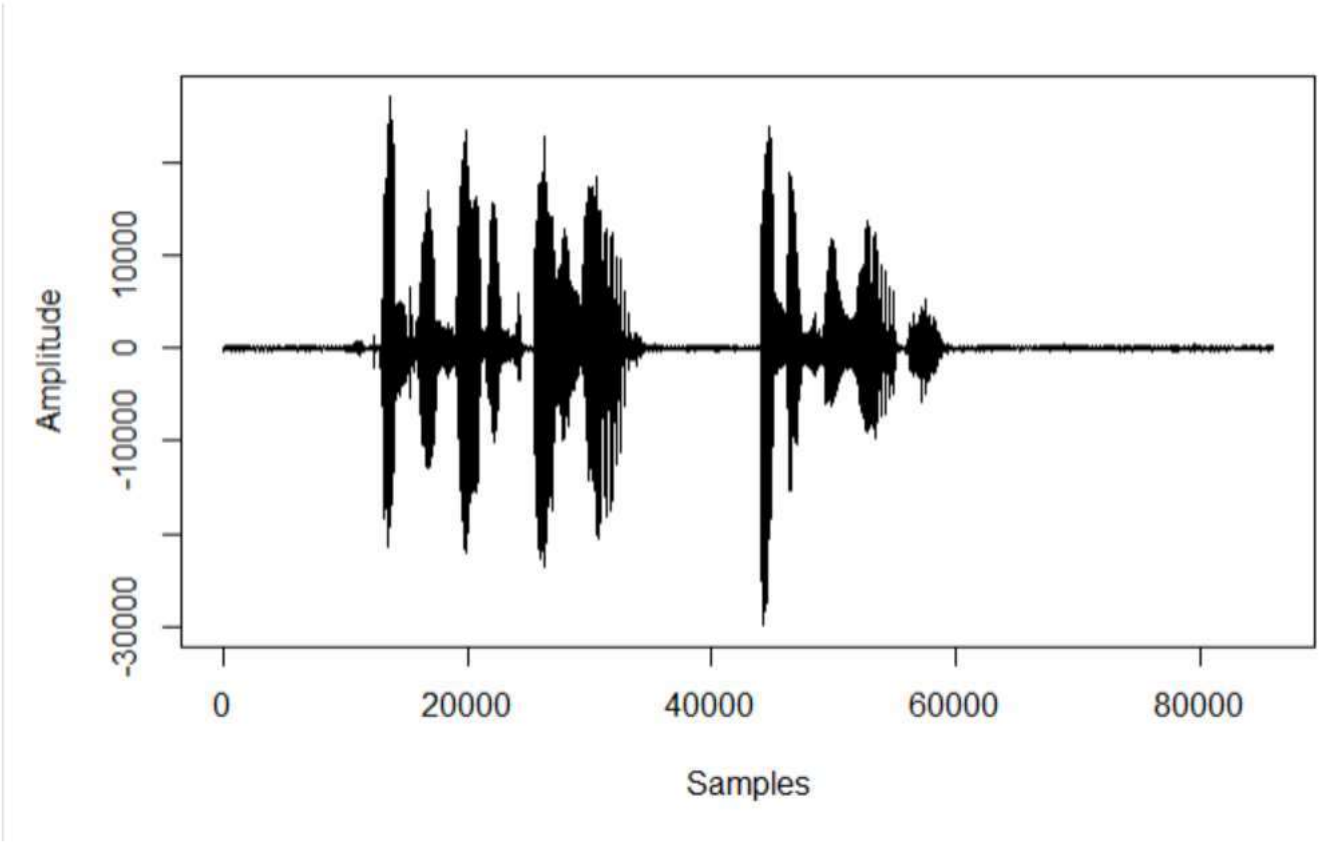


Figure 5: Amplitude Versus Time

The images were created by myself from one of the voice recordings used to produce the Voice Recognition oriented data set.

The measures we mined from the spectrogram data are listed and explained here. Their explanations come from the codebook of the Becker Data, source 7, with some additional clarifying explanations obtained by Google web search for those terms.

meanfreq: mean frequency of the wave(in kHz)

sd: standard deviation of frequency of the wave

median: median frequency (in kHz)

Q25: first quantile for frequency (in kHz)

Q75: third quantile for frequency (in kHz)

IQR: interquantile range for frequency (in kHz)

skew: skewness of the spectrogram's frequency, computed as $S = \sum_{i=1}^{N}(freq_i - meanfreq)^3 \times \frac{1}{sd^3}$. S<0 indicates left skew, S>0 indicates right skew, S=0 indicates perfect symmetry

kurt: kurtosis of the spectrogram's frequency, computed according to $K = \sum_{i=1}^{N}(freq_i - meanfreq)^4 \times \frac{1}{sd^4}$, measures the spectrogram relative to the normal

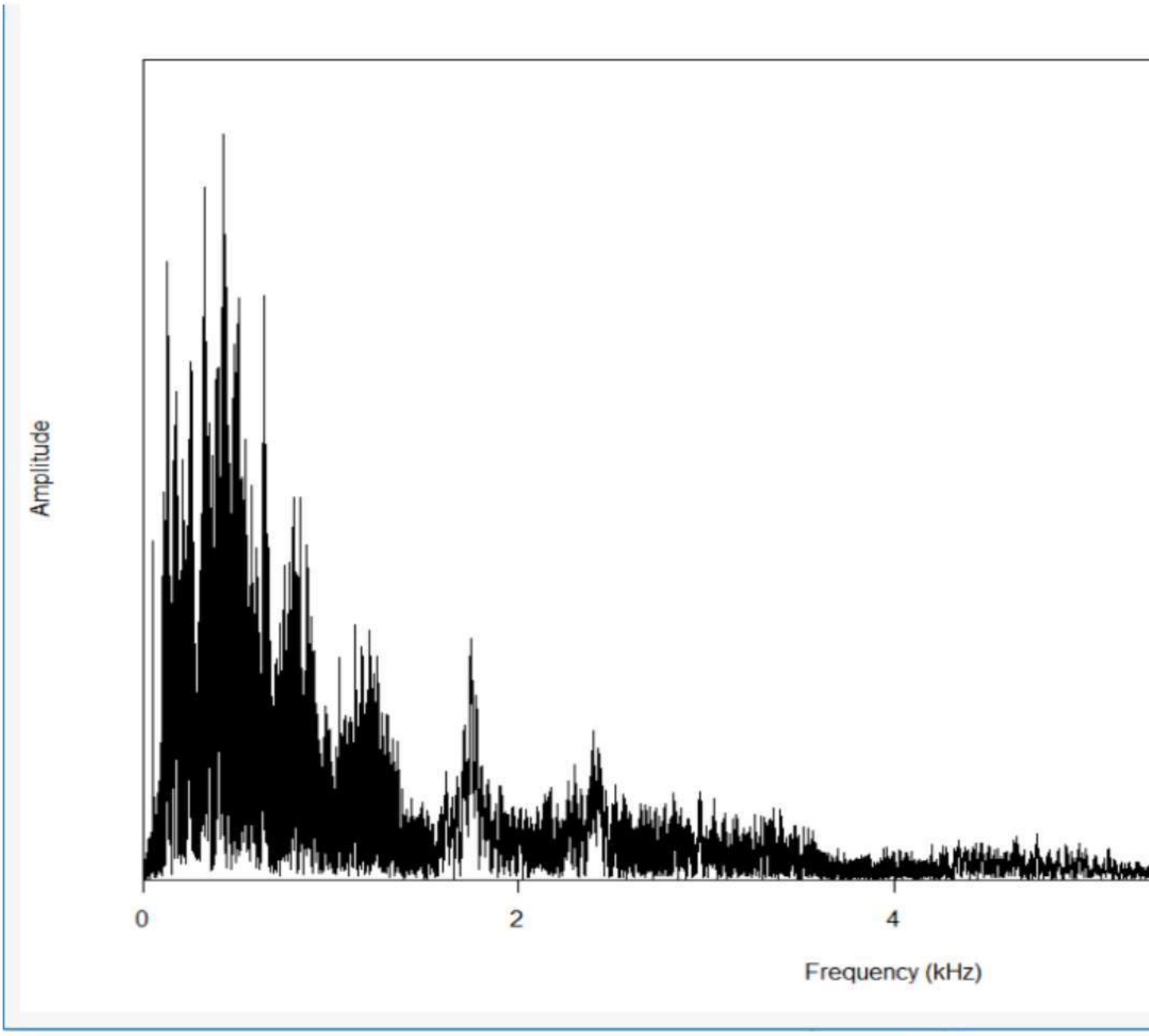


Figure 6: Frequency Versus Amplitude

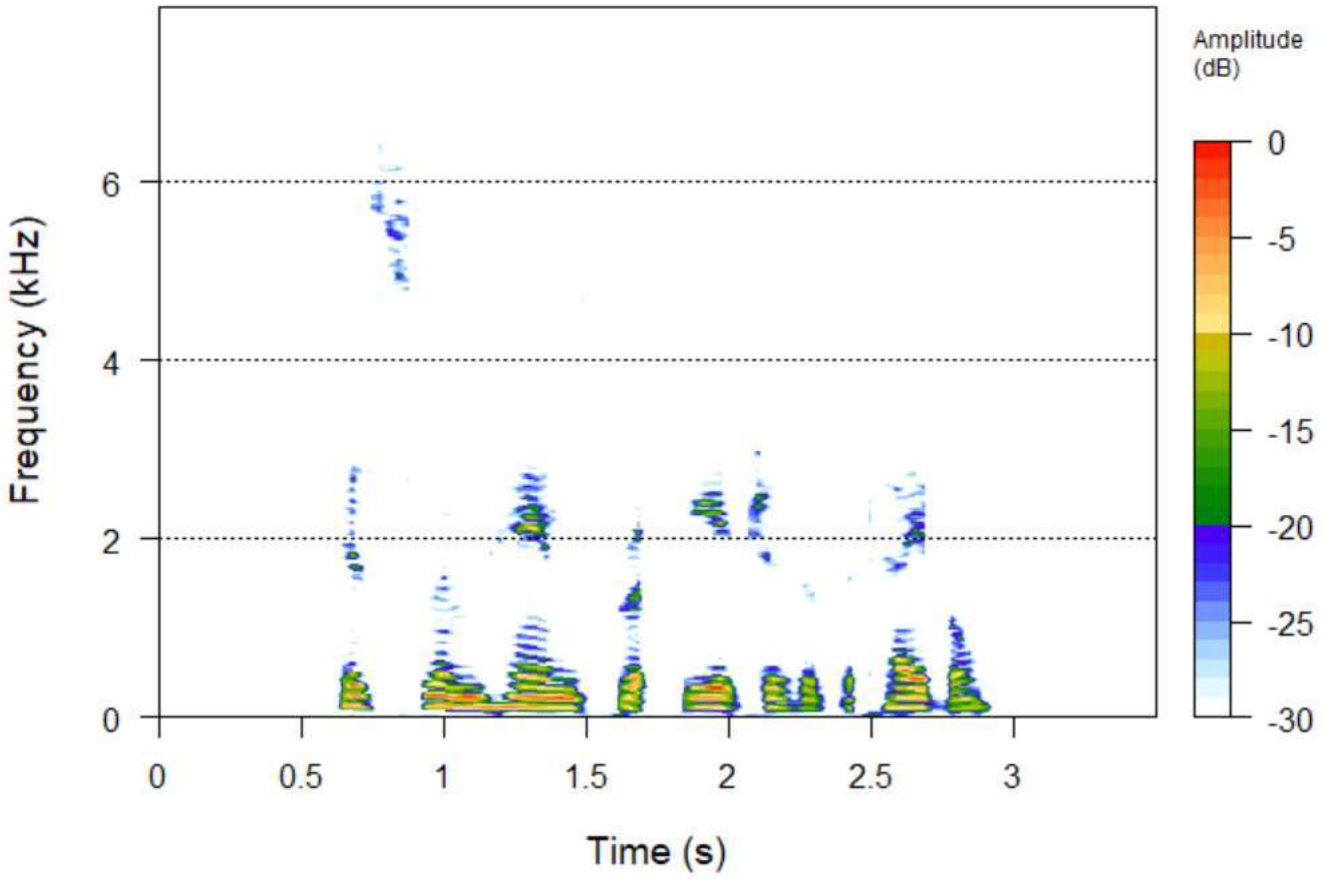


Figure 7: A More Complex Spectrogram: Frequency, Amplitude, and Time

curve. K<3 indicates fewer items at center and tails than expected from normal but more in the shoulders, K>3 indicates more items at the center and tails than expected but fewer at the shoulders, and K=3 indicates a perfect normal curve.

sp.ent: spectral entropy. Describes the complexity of a sound wave, ie how much information is being conveyed. A pure synthetic tone has low entropy, a recording from a crowded diner has high entropy. Roughly speaking it indicates how "noise-like" a sound is compared to how "tonelike". Ranges between 0 and 1.

sfm: spectral flatness. White noise produces a spectrogram that looks nearly flat, with only minor rising and falling around its central tone. This measure indicates how steady and near to white noise a spectrogram is. Sfm closer to 1.0 indicates a very monotonic sound and human voices typically score much closer to 0.0.

mode: mode frequency

centroid: frequency centroid. Computed as $C = \sum_{i=1}^{N}(freq_i - meanfreq)^2 \times \frac{1}{sd^2}$

peakf: peak frequency (frequency with highest energy)

meanfun: average of fundamental frequency measured across acoustic signal. Fundamental frequency is the lowest frequency produced by oscillation of the object. In terms of hearing, it is the lowest pitch or tone that you hear with harmonics rising above it where the the frequency of the sound waves exactly double that frequency. The mean fundamental frequency thus gives an approximation of a person's most comfortable "natural" pitch, though it can be forced higher or lower by modulation of the voice.

minfun: minimum fundamental frequency measured across acoustic signal

maxfun: maximum fundamental frequency measured across acoustic signal

meandom: average of dominant frequency measured across acoustic signal. Dominant frequencies are local maximums of the frequencies, the apexes in the wave.

mindom: minimum of dominant frequency measured across acoustic signal

maxdom: maximum of dominant frequency measured across acoustic signal

dfrange: range of dominant frequency measured across acoustic signal

modindx: modulation index. Calculated as the accumulated absolute difference between adjacent measurements of fundamental frequencies divided by the frequency range.

Chapter 2: Spec Measures and Biological Sex

The following is an example of a classification forest model built upon measurements from spectrogram data. The work is done on the dataset voice.csv compiled by Kory Becker and available on Kaggle. This dataset was created using functions from the libraries seewave and tuneR in R's Cran repository to create and analyze spectrograms from a set of voice recordings. Each entry (row) in the data is a voice recording and each variable (column) is a measurement taken off of the spectrogram (excepting the variable "label" which is the biological sex of the entry, reported directly from Becker's data). This dataset was compiled with the express intention of predicting biological sex. My intent is to construct a model that groups entries by the individual who speaks them, however this seemed a fair way to test my procedures and this sort of measurement data. It is a simpler problem than Voice Recognition, previous work has shown biological sex to be one of the foremost predictable features of a voice. Emulating this experiment lets us test our understanding of the new SVM method against known data and a known method of classification, Random Forest.

First, let's take a look at some of our data. For definitions of the variables, see the last page of the Introduction.

```
voice = read.csv("/home/class19/gyoung19/StatComps/voice.csv")
```

```
msummary(voice)
```

```
    meanfreq            sd              median            Q25
 Min.   :0.03936   Min.   :0.01836   Min.   :0.01097   Min.   :0.0002288
 1st Qu.:0.16366   1st Qu.:0.04195   1st Qu.:0.16959   1st Qu.:0.1110865
 Median :0.18484   Median :0.05916   Median :0.19003   Median :0.1402864
 Mean   :0.18091   Mean   :0.05713   Mean   :0.18562   Mean   :0.1404556
```

```
3rd Qu.:0.19915   3rd Qu.:0.06702   3rd Qu.:0.21062   3rd Qu.:0.1759388
Max.   :0.25112   Max.   :0.11527   Max.   :0.26122   Max.   :0.2473469
      Q75               IQR               skew              kurt
Min.   :0.04295   Min.   :0.01456   Min.   : 0.1417   Min.   :   2.068
1st Qu.:0.20875   1st Qu.:0.04256   1st Qu.: 1.6496   1st Qu.:   5.670
Median :0.22568   Median :0.09428   Median : 2.1971   Median :   8.319
Mean   :0.22476   Mean   :0.08431   Mean   : 3.1402   Mean   :  36.569
3rd Qu.:0.24366   3rd Qu.:0.11418   3rd Qu.: 2.9317   3rd Qu.:  13.649
Max.   :0.27347   Max.   :0.25223   Max.   :34.7255   Max.   :1309.613
    sp.ent            sfm               mode           centroid
Min.   :0.7387   Min.   :0.03688   Min.   :0.0000   Min.   :0.03936
1st Qu.:0.8618   1st Qu.:0.25804   1st Qu.:0.1180   1st Qu.:0.16366
Median :0.9018   Median :0.39634   Median :0.1866   Median :0.18484
Mean   :0.8951   Mean   :0.40822   Mean   :0.1653   Mean   :0.18091
3rd Qu.:0.9287   3rd Qu.:0.53368   3rd Qu.:0.2211   3rd Qu.:0.19915
Max.   :0.9820   Max.   :0.84294   Max.   :0.2800   Max.   :0.25112
   meanfun            minfun            maxfun           meandom
Min.   :0.05557   Min.   :0.009775   Min.   :0.1031   Min.   :0.007812
1st Qu.:0.11700   1st Qu.:0.018223   1st Qu.:0.2540   1st Qu.:0.419828
Median :0.14052   Median :0.046110   Median :0.2712   Median :0.765795
Mean   :0.14281   Mean   :0.036802   Mean   :0.2588   Mean   :0.829211
3rd Qu.:0.16958   3rd Qu.:0.047904   3rd Qu.:0.2775   3rd Qu.:1.177166
Max.   :0.23764   Max.   :0.204082   Max.   :0.2791   Max.   :2.957682
    mindom            maxdom            dfrange          modindx
Min.   :0.004883   Min.   : 0.007812   Min.   : 0.000   Min.   :0.00000
1st Qu.:0.007812   1st Qu.: 2.070312   1st Qu.: 2.045   1st Qu.:0.09977
Median :0.023438   Median : 4.992188   Median : 4.945   Median :0.13936
Mean   :0.052647   Mean   : 5.047277   Mean   : 4.995   Mean   :0.17375
3rd Qu.:0.070312   3rd Qu.: 7.007812   3rd Qu.: 6.992   3rd Qu.:0.20918
Max.   :0.458984   Max.   :21.867188   Max.   :21.844   Max.   :0.93237
   label
female:1584
male  :1584
```

```r
voice2 <- voice %>% dplyr::select(-label)
cor(voice2)
```

```
             meanfreq         sd     median        Q25          Q75
meanfreq   1.0000000 -0.7390388  0.9254454  0.9114163  0.740996718
sd        -0.7390388  1.0000000 -0.5626026 -0.8469309 -0.161075841
median     0.9254454 -0.5626026  1.0000000  0.7749216  0.731849232
Q25        0.9114163 -0.8469309  0.7749216  1.0000000  0.477139811
Q75        0.7409967 -0.1610758  0.7318492  0.4771398  1.000000000
IQR       -0.6276051  0.8746603 -0.4773520 -0.8741890  0.009635774
skew      -0.3223269  0.3145970 -0.2574071 -0.3194753 -0.206338932
kurt      -0.3160356  0.3462409 -0.2433816 -0.3501824 -0.148880617
sp.ent    -0.6012025  0.7166200 -0.5020049 -0.6481258 -0.174905239
sfm       -0.7843323  0.8380865 -0.6616899 -0.7668745 -0.378198373
mode       0.6877152 -0.5291500  0.6774327  0.5912770  0.486857375
centroid   1.0000000 -0.7390388  0.9254454  0.9114163  0.740996718
meanfun    0.4608444 -0.4662815  0.4149093  0.5450351  0.155090956
minfun     0.3839368 -0.3456089  0.3376019  0.3209943  0.258002476
maxfun     0.2740041 -0.1296619  0.2513280  0.1998407  0.285583560
meandom    0.5366661 -0.4827262  0.4559427  0.4674028  0.359180617
mindom     0.2292610 -0.3576670  0.1911687  0.3022549 -0.023750103
maxdom     0.5195277 -0.4822778  0.4389190  0.4596832  0.335114045
dfrange    0.5155699 -0.4759991  0.4356207  0.4543938  0.335647521
modindx   -0.2169787  0.1226597 -0.2132975 -0.1413774 -0.216474678
                 IQR        skew        kurt     sp.ent         sfm
meanfreq -0.627605054 -0.32232693 -0.31603555 -0.6012025 -0.78433231
sd        0.874660319  0.31459695  0.34624087  0.7166200  0.83808650
median   -0.477352003 -0.25740709 -0.24338163 -0.5020049 -0.66168990
Q25      -0.874188990 -0.31947531 -0.35018239 -0.6481258 -0.76687452
Q75       0.009635774 -0.20633893 -0.14888062 -0.1749052 -0.37819837
IQR       1.000000000  0.24949748  0.31618474  0.6408132  0.66360146
skew      0.249497476  1.00000000  0.97702046 -0.1954592  0.07969407
kurt      0.316184735  0.97702046  1.00000000 -0.1276436  0.10988403
sp.ent    0.640813242 -0.19545924 -0.12764358  1.0000000  0.86641084
sfm       0.663601458  0.07969407  0.10988403  0.8664108  1.00000000
```

```
mode     -0.403763599 -0.43485906 -0.40672189 -0.3252985 -0.48591287
centroid -0.627605054 -0.32232693 -0.31603555 -0.6012025 -0.78433231
meanfun  -0.534461948 -0.16766801 -0.19455985 -0.5131937 -0.42106568
minfun   -0.222679719 -0.21695429 -0.20320141 -0.3058260 -0.36210032
maxfun   -0.069588302 -0.08086107 -0.04566725 -0.1207380 -0.19236944
meandom  -0.333362476 -0.33684839 -0.30323357 -0.2935624 -0.42844249
mindom   -0.357036676 -0.06160765 -0.10331264 -0.2948689 -0.28959288
maxdom   -0.337876663 -0.30565086 -0.27450011 -0.3242531 -0.43664879
dfrange  -0.331563477 -0.30464003 -0.27272943 -0.3190536 -0.43157977
modindx   0.041252438 -0.16932471 -0.20553932  0.1980743  0.21147723
                mode   centroid     meanfun       minfun       maxfun
meanfreq  0.6877152  1.0000000  0.46084440  0.383936793  0.27400407
sd       -0.5291500 -0.7390388 -0.46628148 -0.345608905 -0.12966188
median    0.6774327  0.9254454  0.41490926  0.337601923  0.25132802
Q25       0.5912770  0.9114163  0.54503508  0.320994291  0.19984072
Q75       0.4868574  0.7409967  0.15509096  0.258002476  0.28558356
IQR      -0.4037636 -0.6276051 -0.53446195 -0.222679719 -0.06958830
skew     -0.4348591 -0.3223269 -0.16766801 -0.216954285 -0.08086107
kurt     -0.4067219 -0.3160356 -0.19455985 -0.203201414 -0.04566725
sp.ent   -0.3252985 -0.6012025 -0.51319368 -0.305826013 -0.12073798
sfm      -0.4859129 -0.7843323 -0.42106568 -0.362100316 -0.19236944
mode      1.0000000  0.6877152  0.32477126  0.385467306  0.17232879
centroid  0.6877152  1.0000000  0.46084440  0.383936793  0.27400407
meanfun   0.3247713  0.4608444  1.00000000  0.339386726  0.31195050
minfun    0.3854673  0.3839368  0.33938673  1.000000000  0.21398718
maxfun    0.1723288  0.2740041  0.31195050  0.213987182  1.00000000
meandom   0.4914794  0.5366661  0.27083961  0.375979020  0.33755275
mindom    0.1981496  0.2292610  0.16216251  0.082015330 -0.24342566
maxdom    0.4771867  0.5195277  0.27798214  0.317860109  0.35539024
dfrange   0.4737750  0.5155699  0.27515429  0.316486170  0.35988049
modindx  -0.1823435 -0.2169787 -0.05485794  0.002041973 -0.36302924
             meandom      mindom      maxdom     dfrange      modindx
meanfreq  0.53666606  0.22926100  0.51952765  0.51556987 -0.216978748
sd       -0.48272620 -0.35766702 -0.48227782 -0.47599914  0.122659705
median    0.45594268  0.19116867  0.43891903  0.43562066 -0.213297510
Q25       0.46740280  0.30225493  0.45968325  0.45439385 -0.141377375
```

Q75	0.35918062	-0.02375010	0.33511405	0.33564752	-0.216474678
IQR	-0.33336248	-0.35703668	-0.33787666	-0.33156348	0.041252438
skew	-0.33684839	-0.06160765	-0.30565086	-0.30464003	-0.169324710
kurt	-0.30323357	-0.10331264	-0.27450011	-0.27272943	-0.205539321
sp.ent	-0.29356241	-0.29486887	-0.32425314	-0.31905357	0.198074268
sfm	-0.42844249	-0.28959288	-0.43664879	-0.43157977	0.211477226
mode	0.49147940	0.19814956	0.47718671	0.47377496	-0.182343536
centroid	0.53666606	0.22926100	0.51952765	0.51556987	-0.216978748
meanfun	0.27083961	0.16216251	0.27798214	0.27515429	-0.054857943
minfun	0.37597902	0.08201533	0.31786011	0.31648617	0.002041973
maxfun	0.33755275	-0.24342566	0.35539024	0.35988049	-0.363029240
meandom	1.00000000	0.09965605	0.81283770	0.81130367	-0.180954102
mindom	0.09965605	1.00000000	0.02663969	0.00866554	0.200212223
maxdom	0.81283770	0.02663969	1.00000000	0.99983841	-0.425531023
dfrange	0.81130367	0.00866554	0.99983841	1.00000000	-0.429266452
modindx	-0.18095410	0.20021222	-0.42553102	-0.42926645	1.000000000

It seems the data is split evenly between samples from biological males and females. Most of the variables appear to occupy narrow ranges centered about their means. This may be due to the large number of observations in the dataset or due to the general similarity of human voice ranges. We do, however see a high degree of correlation among these variables and this may make interpretation of our models in terms of a single variable more difficult. We could do a Principal Components Reduction(PCR) here to make dealing with the many variables more straightforward, but let's see if it is necessary first. If we can obtain a model that depends on one of these direct measurements from the spectrograms, that would be much preferable.

We began by splitting the data into training and testing sets.

```
set.seed(36) #for reproducability
train <- voice %>%
  sample_frac(0.80) %>%
  mutate(label = as.factor(label))

test <- voice %>%
  setdiff(train) %>%
  mutate(label = as.factor(label))
```

We create a random forest on the training set, using default params of 500 trees considering 4 variables at each step with no maximum tree length for the forest and the default options for the svm.

```
set.seed(2250) #reproducability
rf <- randomForest(label ~ . , data = train, importance = TRUE)
svm <- svm(label ~ ., data = train)
```

```
estimate1 <- predict(rf, newdata = test)
misclass1 <- ifelse(estimate1 != test$label,1,0)
mean(misclass1)
```

```
[1] 0.009463722
```

Our random forest model seems to be able to differentiate between the biological sex of the speakers with a high degree of accuracy, misclassifying only 0.94% of the time. We would also like to check which of these many variables were most informative to our model.

```
importance(rf) #list variables and importance
```

```
            female      male MeanDecreaseAccuracy MeanDecreaseGini
meanfreq 10.223607 11.481962             14.99099        26.770622
sd       16.851184 15.945382             21.81419        97.635865
median    9.696393 14.214570             16.44871        18.682786
Q25      18.925172 23.620311             28.38364       159.398463
Q75      10.617579 12.139675             15.66817        13.699193
IQR      20.394572 35.714711             38.58051       244.137512
skew     10.372807  8.376263             12.99566        13.602288
kurt     10.526120  8.171386             12.32354        10.229845
sp.ent   14.865760  9.826768             17.64559        57.016189
sfm      16.982912 12.861330             20.18108        41.084571
mode     10.639303 11.233951             13.37967        19.905143
centroid  9.520364 12.591789             16.01013        22.352289
meanfun  42.814560 65.873813             71.07732       473.254813
minfun   11.732828 12.073388             16.17091        12.338545
maxfun    9.386261  7.286574             11.16917         6.341982
```

```
meandom  12.123758  8.913113          13.98629         9.560521
mindom    5.866225 10.972492          11.43918         8.878321
maxdom   13.384097  9.478714          15.51142        12.034714
dfrange  15.439257  9.604017          17.23617        11.119503
modindx  14.207147  7.524171          14.92663         8.412018
```

It seems that the mean fundamental frequency leads by quite a large margin and probably merits further consideration in my continuing work. This is not at all unexpected, fundamental frequency has been useful in differentiating biological sex of a speaker before. Reports on the actual numbers vary, but generally state that the average range of male fundamental frequency is somewhere between 85 to 180 Hz and the average female range is between 165 and 255 Hz.

Now let's see how the svm performed.

```
estimate2 <- predict(svm,newdata = test)
misclass2 <- ifelse(estimate2 != test$label,1,0)
mean(misclass2)
```

```
[1] 0.006309148
```

It performed even better than the Random Forests, misclassifying a scant 0.63% of the time. Let's examine the weights to get an idea of which variables were most important.

```
#code from Source2
W <- t(svm$coefs) %*% svm$SV
W
```

```
     meanfreq        sd   median      Q25       Q75       IQR      skew
[1,]  5.15502 -2.862442 7.208953 13.2442 -1.074973 -15.65304 1.295321
          kurt    sp.ent      sfm     mode centroid  meanfun   minfun
[1,] 0.7507584 -5.611745 10.72697 10.22149  5.15502 74.00405 20.95893
       maxfun  meandom    mindom    maxdom   dfrange   modindx
[1,] 14.30399 4.155208 -3.703357 -7.158458 -7.095421 -3.072451
```

With reference to the high correlation among variables that we noted earlier, let's run both models solely on those highest performing variables.

```
train2 <- train %>% dplyr::select(meanfun, IQR, label)
test2 <- test %>% dplyr::select(meanfun, IQR, label)
```

```
set.seed(2250) #reproducability
rf <- randomForest(label ~ . , data = train2, importance = TRUE)
svm <- svm(label ~ ., data = train2)
```

```
estimate1 <- predict(rf, newdata = test2)
misclass1 <- ifelse(estimate1 != test2$label,1,0)
mean(misclass1)
```

```
[1] 0.01419558
```

```
estimate2 <- predict(svm,newdata = test2)
misclass2 <- ifelse(estimate2 != test2$label,1,0)
mean(misclass2)
```

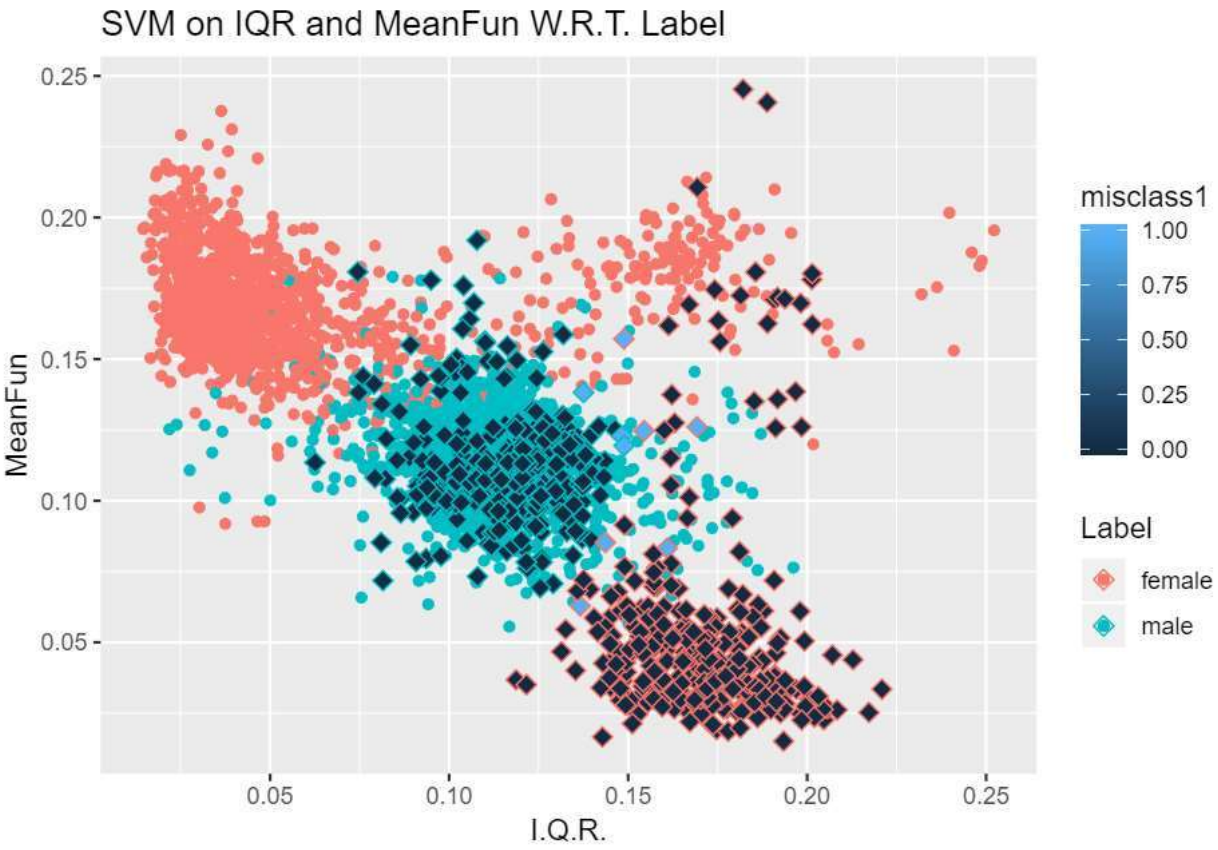
```
[1] 0.01735016
```

Both models are still doing incredibly well, better than 98% accuracy, reinforcing our impression that these two variables are the most indicative. This accuracy is comparable to the accuracy reported in the original experiment by Becker, so this example has served to confirm the legitimacy of these measures and our chosen methods.

Let's visualize things.

```
estimatedf2 <- as.data.frame(estimate2)
testdf <- test2 %>% select(-label)
estimatedf2 <- cbind(estimatedf2,testdf)
```

```
g<- ggplot(data = train, aes(x = IQR, y = meanfun, colour = label))+
  labs(x = "I.Q.R.", y = "MeanFun", colour = "Label") +
  ggtitle("SVM on IQR and MeanFun W.R.T. Label")+
  geom_point()+
  geom_point(data = estimatedf2, aes(x=meanfun,y=IQR,colour=estimate2,fill=misclass1),s
g
```



The diamond points in the graph are observations from the test set that are colored by their predicted label and filled based on whether that prediction was correct. It is plainly evident that the majority of the predicted labels were correct and that most of the female voices fall within a narrower IQR at a higher meanfun than do the male voices, although there is a strange pocket there of female voices that are both lower and more variable than the male voices. It is plain to see that the SVM is classifying observations based on a circle around the main body of male voices, with any observations falling outside being guessed as female.

Chapter 3: ScrapingVoxForge

Before SVM could be performed, we needed data that was suitable for Voice Recognition. This was the process by which I compiled my own data set after the fashion of Becker's.

We downloaded data from links off of the VoxForge Home site to local files. The code here makes spectrograms for each of the waves in the files and then pulls our desired spectrogram stats from those sepctrograms.

```
path <- 'C:/Users/Gabriel Young/Documents/StatComps/
VoxForge/anonymous-20110310-itc/wav/b0195.wav'
spectro1 <- readWave(path)
```

```
#Code for pulling stats from a single wav file

#isolate the signal
signal <- spectro1@left

#obtain signal duration
dur = length(signal)/spectro1@samp.rate

#get the sampling rate of the signal
fs = spectro1@samp.rate

# center to remove DC offset
signal = signal - mean(signal)

#obtain the spec object
s <- spec(spectro1,fs)
```

```
#get frequency properties we want using specprop
props <- as.data.frame(specprop(s,fs))

#get fundamental frequencies, eliminate NaNs
f <- as.data.frame(fund(spectro1,fs))
f2 <- f[complete.cases(f),]

#get summary stats of fundamental frequency measures
meanfun <- mean(f2$y)
minfun <- min(f2$y)
maxfun <- max(f2$y)
props <- cbind(props,meanfun, minfun, maxfun)

#get dominant frequencies, eliminate NaNs
d <- as.data.frame(dfreq(spectro1,fs))
d2 <- d[complete.cases(d),]

#get summary stats of dominant frequencies
meandom <- mean(d2$y)
mindom <- min(d2$y)
maxdom <- max(d2$y)

#add to data
props <- cbind(props,meandom,mindom,maxdom)
```

Having developed code for mining a single observation, we then mined the rest of our data.

```
path <- 'C:/Users/Gabriel Young/
Documents/StatComps/VoxForge/'
folders <- as.data.frame(list.files(path))
folders <- folders[4:16,]
folders <- as.vector(folders)


#initialize storage dataframe
```

```
desiredstats <- data.frame()

#apply spec processes to all files
for(folder in folders)
{
  files <- as.data.frame(list.files(paste(path, folder, "/wav", sep = "")))
  files <- files %>% mutate(names = as.character(
    `list.files(paste(path, folder, "/wav", sep = ""))`)) %>%
    select(names)
  realfiles = files$names
  for(f in realfiles)
  {
    spectro1 <- readWave(filename = paste(path,folder,"/wav/",f,sep=""))
    signal <- spectro1@left

    dur = length(signal)/spectro1@samp.rate

    fs = spectro1@samp.rate

    # demean to remove DC offset
    signal = signal - mean(signal)

    s <- spec(spectro1,fs)
    props <- as.data.frame(specprop(s,fs))
    props <- props %>% mutate(Files = f, Folders = folder)

    f <- as.data.frame(fund(spectro1,fs))
    f2 <- f[complete.cases(f),]
    meanfun <- mean(f2$y)
    minfun <- min(f2$y)
    maxfun <- max(f2$y)

    d <- as.data.frame(dfreq(spectro1,fs))
    d2 <- d[complete.cases(d),]
    meandom <- mean(d2$y)
    mindom <- min(d2$y)
```

```
    maxdom <- max(d2$y)

    props <- cbind(props,meanfun, minfun, maxfun)
    props <- cbind(props,meandom,mindom,maxdom)

    desiredstats <- rbind(desiredstats, props)
  }
}
```

```
write.csv(desiredstats, file = "VoiceRecogData.csv")
```

This furnished us with a 130 observation dataset with labels of the person speaking for each observation, catering to the Voice Recognition experiment.

Chapter 4: Voice Recognition

Having created a suitable data set for ourselves, we again run the same procedure as with the preliminary analysis. In this data, our response variable is Speaker. We start by splitting the data into training and testing sets. Let us, again, get a summary feel for the data we're working with.

```
voice <-
  read.csv("/home/class19/gyoung19/StatComps/GYoung/VoiceRecogData.csv")
voice <- voice %>%
  mutate(Speaker = Folders) %>%
  dplyr::select(-X,-Folders,-Files)
```

```
msummary(voice)
```

```
      mean                sd           median            sem
 Min.   : 931.4   Min.   :1191   Min.   : 289.6   Min.   : 5.822
 1st Qu.:1595.4   1st Qu.:1676   1st Qu.: 795.4   1st Qu.: 8.000
 Median :1878.7   Median :1888   Median :1012.3   Median : 9.306
 Mean   :1856.1   Mean   :1844   Mean   :1117.6   Mean   : 9.304
 3rd Qu.:2102.2   3rd Qu.:2021   3rd Qu.:1426.6   3rd Qu.:10.534
 Max.   :2998.1   Max.   :2402   Max.   :2670.6   Max.   :14.568

      mode             Q25             Q75            IQR
 Min.   :  0.0   Min.   :104.1   Min.   :1021   Min.   : 831.2
 1st Qu.:  0.0   1st Qu.:297.5   1st Qu.:2098   1st Qu.:1706.3
 Median :102.0   Median :451.5   Median :2689   Median :2261.1
 Mean   :174.8   Mean   :440.9   Mean   :2816   Mean   :2374.6
 3rd Qu.:284.5   3rd Qu.:546.2   3rd Qu.:3465   3rd Qu.:2975.4
 Max.   :922.7   Max.   :941.0   Max.   :4993   Max.   :4459.0
```

```
      cent           skewness          kurtosis            sfm
 Min.   : 931.4   Min.   : 2.573   Min.   :  11.84   Min.   :0.1574
 1st Qu.:1595.4   1st Qu.: 5.123   1st Qu.:  44.66   1st Qu.:0.3052
 Median :1878.7   Median : 6.157   Median :  70.55   Median :0.3757
 Mean   :1856.1   Mean   : 6.875   Mean   : 138.41   Mean   :0.3740
 3rd Qu.:2102.2   3rd Qu.: 8.019   3rd Qu.: 113.84   3rd Qu.:0.4411
 Max.   :2998.1   Max.   :20.277   Max.   :1418.01   Max.   :0.6863

       sh              prec          meanfun          minfun
 Min.   :0.8477   Min.   :0.1039   Min.   :2.028   Min.   :0.06275
 1st Qu.:0.9057   1st Qu.:0.1667   1st Qu.:3.329   1st Qu.:0.06702
 Median :0.9221   Median :0.2072   Median :3.667   Median :0.09357
 Mean   :0.9184   Mean   :0.2068   Mean   :3.729   Mean   :0.34737
 3rd Qu.:0.9357   3rd Qu.:0.2424   3rd Qu.:4.127   3rd Qu.:0.23708
 Max.   :0.9628   Max.   :0.3810   Max.   :5.207   Max.   :2.00000

    meandom           mindom           maxdom
 Min.   :0.06479   Min.   :0.00000   Min.   :0.625
 1st Qu.:0.22542   1st Qu.:0.00000   1st Qu.:4.258
 Median :0.32281   Median :0.00000   Median :4.906
 Mean   :0.35970   Mean   :0.00601   Mean   :4.859
 3rd Qu.:0.45181   3rd Qu.:0.00000   3rd Qu.:5.906
 Max.   :0.90005   Max.   :0.09375   Max.   :7.719

                   Speaker
 anonymous-20110310-itc:10
 anonymous-20110312-bqz:10
 anonymous-20110312-hkp:10
 anonymous-20110312-wbu:10
 anonymous-20110313-wdp:10
 anonymous-20110314-cjq:10
 (Other)                :70
```

```r
voice2 <- voice %>%
  dplyr::select(-Speaker)
cor(voice2)
```

	mean	sd	median	sem	mode
mean	1.00000000	0.81756264	0.86088918	0.71717159	0.050801175
sd	0.81756264	1.00000000	0.45513110	0.75781035	-0.105785450
median	0.86088918	0.45513110	1.00000000	0.48816143	0.134346600
sem	0.71717159	0.75781035	0.48816143	1.00000000	-0.356292651
mode	0.05080118	-0.10578545	0.13434660	-0.35629265	1.000000000
Q25	0.65017839	0.20932517	0.76482563	0.26243104	0.240008693
Q75	0.93970063	0.85392948	0.70639854	0.73849077	-0.025924573
IQR	0.88140796	0.87425504	0.60953841	0.74042889	-0.073874447
cent	1.00000000	0.81756264	0.86088918	0.71717159	0.050801175
skewness	0.22527806	0.31518553	0.14202066	0.37033082	-0.337001682
kurtosis	0.42047710	0.34932051	0.37775659	0.46740518	-0.325234424
sfm	0.80977417	0.78571570	0.62341643	0.50269344	0.244937808
sh	0.92153645	0.71000059	0.81813454	0.60763925	-0.007537413
prec	0.14283819	0.00174393	0.20654404	0.64426137	-0.432678742
meanfun	-0.09783740	0.05064836	-0.25189423	-0.04516136	0.126407444
minfun	-0.11136107	-0.02010391	-0.11476087	-0.02513370	0.130829882
meandom	0.37217158	0.30527010	0.29846846	-0.03174073	0.542452101
mindom	0.11052749	0.07346632	0.07531672	-0.04836885	0.441630290
maxdom	0.41598734	0.42520891	0.25867709	0.09297370	0.457041540

	Q25	Q75	IQR	cent	skewness
mean	0.65017839	0.93970063	0.881407957	1.00000000	0.22527806
sd	0.20932517	0.85392948	0.874255036	0.81756264	0.31518553
median	0.76482563	0.70639854	0.609538410	0.86088918	0.14202066
sem	0.26243104	0.73849077	0.740428893	0.71717159	0.37033082
mode	0.24000869	-0.02592457	-0.073874447	0.05080118	-0.33700168
Q25	1.00000000	0.44650890	0.286039915	0.65017839	-0.08492437
Q75	0.44650890	1.00000000	0.985112631	0.93970063	0.27544628
IQR	0.28603992	0.98511263	1.000000000	0.88140796	0.31129117
cent	0.65017839	0.93970063	0.881407957	1.00000000	0.22527806
skewness	-0.08492437	0.27544628	0.311291168	0.22527806	1.00000000
kurtosis	0.24358528	0.42041034	0.403417789	0.42047710	0.88730157
sfm	0.45074484	0.74930864	0.715833798	0.80977417	0.06307431
sh	0.73512518	0.82852156	0.746025992	0.92153645	0.07317003
prec	0.14544038	0.13829099	0.120152763	0.14283819	0.17635084
meanfun	-0.24033862	0.05109062	0.100888113	-0.09783740	-0.06089698

minfun	-0.24278150	-0.12075626	-0.082673113	-0.11136107	-0.01155280
meandom	0.24032791	0.33428580	0.311812974	0.37217158	-0.35699032
mindom	0.22451526	0.04927586	0.009634273	0.11052749	-0.04427078
maxdom	0.24081229	0.40933801	0.392093213	0.41598734	-0.17450749
	kurtosis	sfm	sh	prec	meanfun
mean	0.42047710	0.80977417	0.921536451	0.14283819	-0.09783740
sd	0.34932051	0.78571570	0.710000588	0.00174393	0.05064836
median	0.37775659	0.62341643	0.818134537	0.20654404	-0.25189423
sem	0.46740518	0.50269344	0.607639253	0.64426137	-0.04516136
mode	-0.32523442	0.24493781	-0.007537413	-0.43267874	0.12640744
Q25	0.24358528	0.45074484	0.735125179	0.14544038	-0.24033862
Q75	0.42041034	0.74930864	0.828521559	0.13829099	0.05109062
IQR	0.40341779	0.71583380	0.746025992	0.12015276	0.10088811
cent	0.42047710	0.80977417	0.921536451	0.14283819	-0.09783740
skewness	0.88730157	0.06307431	0.073170028	0.17635084	-0.06089698
kurtosis	1.00000000	0.12620774	0.352306514	0.28011299	-0.14957654
sfm	0.12620774	1.00000000	0.759072847	-0.13467559	0.02399626
sh	0.35230651	0.75907285	1.000000000	0.09916473	-0.18514284
prec	0.28011299	-0.13467559	0.099164725	1.00000000	-0.12175703
meanfun	-0.14957654	0.02399626	-0.185142841	-0.12175703	1.00000000
minfun	-0.12721610	-0.09175967	-0.267996628	-0.01289882	-0.12620945
meandom	-0.29223769	0.45888265	0.311635613	-0.38920661	0.07896152
mindom	-0.09612413	0.21809332	0.049831080	-0.15416062	0.35773172
maxdom	-0.08467962	0.47341188	0.360199654	-0.36849879	0.30399639
	minfun	meandom	mindom	maxdom	
mean	-0.11136107	0.37217158	0.110527488	0.41598734	
sd	-0.02010391	0.30527010	0.073466324	0.42520891	
median	-0.11476087	0.29846846	0.075316715	0.25867709	
sem	-0.02513370	-0.03174073	-0.048368855	0.09297370	
mode	0.13082988	0.54245210	0.441630290	0.45704154	
Q25	-0.24278150	0.24032791	0.224515257	0.24081229	
Q75	-0.12075626	0.33428580	0.049275860	0.40933801	
IQR	-0.08267311	0.31181297	0.009634273	0.39209321	
cent	-0.11136107	0.37217158	0.110527488	0.41598734	
skewness	-0.01155280	-0.35699032	-0.044270775	-0.17450749	
kurtosis	-0.12721610	-0.29223769	-0.096124127	-0.08467962	

```
sfm       -0.09175967  0.45888265  0.218093318  0.47341188
sh        -0.26799663  0.31163561  0.049831080  0.36019965
prec      -0.01289882 -0.38920661 -0.154160624 -0.36849879
meanfun   -0.12620945  0.07896152  0.357731719  0.30399639
minfun     1.00000000  0.13283577 -0.069112870 -0.10321665
meandom    0.13283577  1.00000000  0.164639261  0.66109542
mindom    -0.06911287  0.16463926  1.000000000  0.23625549
maxdom    -0.10321665  0.66109542  0.236255487  1.00000000
```

The variables for this new set appear to not be standardized in the same way that the Becker set was but this ought not to pose an issue according to the way in which Random Forest and SVM learn. Their ranges about the mean are slightly wider than before but not drastically so and once again the high correlation between variables might call for some sort of Principal Components action, but let's run our tests first.

We start by splitting the data into training and testing sets.

```
#from Professor Wagaman's data splitting examples
set.seed(36) #for reproducability
train <- voice %>%
  sample_frac(0.80) %>%
  mutate(Speaker = as.factor(Speaker))

test <- voice %>%
  setdiff(train) %>%
  mutate(Speaker = as.factor(Speaker))
```

We create a random forest on the training set, using default parameters of 500 trees considering 4 variables at each step with no maximum tree length for the forest and the default options for the SVM.

```
set.seed(2250) #reproducability
rf <- randomForest(Speaker ~ . , data = train, importance = TRUE)
svm <- svm(Speaker ~ ., data = train)
```

```
estimate1 <- predict(rf, newdata = test)
misclass1 <- ifelse(estimate1 != test$Speaker,1,0)
mean(misclass1)
```

```
[1] 0.4615385
```

We can see here that we are doing much worse than with the previous data set aimed at biological sex, our model is only classifying 54% of the data correctly. Still, that is an improvement over random guesswork, which would have a classification rate of around 8%. Let's check again and see which of these many variables were most informative to our model.

```
importance(rf) #list variables and importance
```

```
         anonymous-20110310-itc anonymous-20110312-bqz
mean                  4.4070598              2.2526228
sd                    2.6251283              5.8763140
median                9.7210261              9.5190882
sem                   9.5366518              4.8988455
mode                 13.7298508             12.1679210
Q25                  14.4454219             12.7814131
Q75                   1.2758799              2.7641133
IQR                   1.9250830              2.3408893
cent                  4.2586564              1.9629492
skewness              6.7958306             14.5758375
kurtosis              6.0207849             13.2127298
sfm                   0.7922494              1.9992044
sh                    4.5845651              2.8302549
prec                  8.7819335              0.1142604
meanfun               9.2268615              9.1494387
minfun                7.5529203              6.0963392
meandom              -1.2324420              3.2394816
mindom                1.0010015              2.3097161
maxdom               -0.2296153              3.3414304
         anonymous-20110312-hkp anonymous-20110312-wbu
mean                  0.7400967            -0.59528814
sd                   -1.8496307            -0.01115319
median                2.5021351            -0.78447478
sem                   1.2596965             3.98992413
mode                  9.7567903            10.35385695
Q25                  14.5732128            11.20391713
```

Q75	0.2269190	1.28112484
IQR	-0.5764009	-1.52295344
cent	2.1772254	-1.65971309
skewness	5.9536700	4.63368392
kurtosis	6.4284520	1.19474150
sfm	-1.9863895	2.71137914
sh	1.4247543	-0.55586388
prec	2.2919882	3.55586571
meanfun	7.7697782	-1.23947790
minfun	5.5447717	10.35224884
meandom	4.5960777	1.41252530
mindom	0.0000000	1.60420113
maxdom	2.4500485	2.15643031
	anonymous-20110313-wdp	anonymous-20110314-cjq
mean	0.8021021	-1.2334450
sd	0.6343172	0.5513004
median	2.4545831	4.8690618
sem	-0.3580929	-0.7959346
mode	4.9269534	12.2097589
Q25	9.8706361	9.3046343
Q75	0.4893794	0.9736564
IQR	1.1041657	1.0327538
cent	-0.2411605	1.4089622
skewness	3.2124382	3.3894160
kurtosis	6.4122365	0.8566870
sfm	5.5109804	-0.2257152
sh	3.1110670	3.6578564
prec	2.3752478	0.5082695
meanfun	11.4479332	17.6371134
minfun	3.8880561	13.5431478
meandom	3.6194260	4.0316371
mindom	1.9071415	1.8167779
maxdom	-1.8809679	-2.4522633
	anonymous-20110406-jet	anonymous-20110406-nbk
mean	1.975887e+00	2.68799549
sd	3.594008e-01	1.10245769

median	1.282938e+00	-0.85686073
sem	2.967257e+00	3.15963587
mode	5.801811e+00	8.66551884
Q25	2.009891e+00	4.58501411
Q75	9.438300e-01	-0.06668086
IQR	1.194887e+00	-0.38255463
cent	-8.881784e-17	2.77803774
skewness	1.751293e+00	5.79750516
kurtosis	9.912105e-01	1.89195679
sfm	-1.295892e-01	0.82317186
sh	2.382028e+00	1.99119893
prec	7.832738e-01	-2.11626416
meanfun	1.621570e+00	3.75397030
minfun	1.286650e+00	-1.17527726
meandom	-4.030714e+00	2.63497477
mindom	1.001002e+00	0.00000000
maxdom	-5.003140e-02	1.80008175
	anonymous-20110406-opo	anonymous-20110406-php
mean	3.6018324	0.8648926
sd	3.7497896	1.8318236
median	2.1963040	1.7207555
sem	3.1859924	0.1147198
mode	12.7594749	12.2472430
Q25	10.5483258	5.3589099
Q75	1.8467135	-1.5913141
IQR	0.7374003	-0.4973572
cent	2.8483410	2.8604960
skewness	4.4063324	8.4026811
kurtosis	4.2924790	5.3519865
sfm	4.0188087	2.8910430
sh	1.7291728	4.0877512
prec	3.5016686	-1.0572143
meanfun	7.9471153	2.9637405
minfun	2.9819068	3.8716040
meandom	2.1453750	1.1655518
mindom	9.6611162	0.0000000

maxdom	6.7670252	-1.8380069	
	anonymous-20110406-tmh	anonymous-20110406-uwi	
mean	1.03221935	-0.17189057	
sd	-1.83571238	0.88756199	
median	-2.16805654	0.23720270	
sem	0.46600965	6.19368554	
mode	7.16024092	11.05670480	
Q25	1.19571030	8.58262829	
Q75	1.63736531	-0.17417068	
IQR	1.21474540	-1.41942771	
cent	-1.07661965	0.01994028	
skewness	1.40858361	10.81228235	
kurtosis	2.10534383	12.11898997	
sfm	1.31134493	1.88833049	
sh	-0.03215787	3.15799880	
prec	0.35110317	4.86410390	
meanfun	-0.08272619	2.60051860	
minfun	-1.54022767	1.75346730	
meandom	6.38200544	3.39515272	
mindom	1.00100150	2.05518556	
maxdom	3.04318993	4.97164375	
	anonymous-20111009-eyc	MeanDecreaseAccuracy	MeanDecreaseGini
mean	-0.6469549	5.275592	2.347654
sd	1.7084209	5.621677	2.641594
median	5.3487326	11.198334	4.677975
sem	0.2464368	10.630903	5.288778
mode	7.0538535	23.523196	9.294262
Q25	6.5549755	23.035826	9.776514
Q75	1.9154228	3.403797	1.809911
IQR	1.3278747	1.141061	1.988113
cent	1.1203865	4.480609	2.480985
skewness	8.8749683	19.639549	8.296891
kurtosis	11.5412980	19.863515	8.802468
sfm	2.0176435	6.622198	3.440724
sh	1.1010277	7.262946	2.759295
prec	-0.4909416	8.200927	4.095014

meanfun	11.2498788	21.026062	10.702232
minfun	-4.0591347	15.546762	6.824623
meandom	-0.0425244	8.896386	4.300112
mindom	1.0160256	9.866286	1.788645
maxdom	3.8261001	6.155441	3.614786

With so many response categories, it has become much more difficult to discern the importance of of any single variable.

Let's see how the SVM performed.

```
estimate2 <- predict(svm,newdata = test)
misclass2 <- ifelse(estimate2 != test$Speaker,1,0)
mean(misclass2)
```

```
[1] 0.5
```

Contrary to our original test, it actually performed worse here than the Random Forests by a little, misclassifying a half of the testing observations. When we look at the weights table

```
#code from Source2
W <- t(svm$coefs) %*% svm$SV
W
```

```
           mean          sd      median        sem       mode        Q25
 [1,] -3.7273657 -0.03504357 -5.58251486 -2.7585724  8.0089323  2.1029127
 [2,]  4.9149467  5.97236428  1.57934857 -1.7984637 16.2977456  9.0568985
 [3,]  2.4430041  3.15959784  2.18914618  3.9075201  9.3424773  6.0155046
 [4,] -4.3363175  0.70669410 -7.27112686  0.4687597  4.1212862 -1.2130570
 [5,]  2.9657714  4.54441037 -0.10707175  7.4846005 -0.8047437  3.6522111
 [6,] -4.6442970  0.06849099 -8.00870090 -4.7513667  0.3669068  0.7318693
 [7,]  1.5837223  3.93134435  0.09710231 -0.2548240  2.3967546  4.3386253
 [8,] -1.0258777 -1.71589904  0.03869577  2.0004297  3.1859179  1.5586233
 [9,] -0.8900622 -6.79781252  3.03238813 -3.9605366  1.3089928  3.8510332
[10,]  2.9620493 -6.32363338  7.07651874  3.2701599 -4.9970677  9.8997442
[11,] 10.0425287  3.92044436  9.15200402  7.4967070 -4.7372618 12.5528749
[12,] -1.8759387 -3.31372558 -2.77781483  1.6148697 -8.5517959  1.2833512
           Q75        IQR       cent   skewness   kurtosis       sfm
```

```
 [1,] -5.6690218 -6.4968414 -3.7273657  1.2677866 -1.1073605  1.601158
 [2,]  1.8989262  0.3017076  4.9149467 -0.1524015 -0.1428662 10.098620
 [3,] -1.9373725 -3.2374552  2.4430041  2.0406163  0.3960797  9.133933
 [4,] -5.7790335 -5.9782995 -4.3363175  2.8122456 -2.2144499  2.820106
 [5,]  2.8846962  2.3990916  2.9657714  0.8386053 -3.3524253  8.806495
 [6,] -5.6218079 -6.1828084 -4.6442970 10.3796360  5.3147227  1.460055
 [7,] -1.7271329 -2.6894726  1.5837223  6.8664843  3.5752615  3.458104
 [8,] -1.5472363 -1.9622643 -1.0258777  6.8632694  2.0727582 -1.670629
 [9,] -0.8455694 -1.6483421 -0.8900622  0.9900556 -0.3946107 -7.983400
[10,]  1.9933572  0.2413454  2.9620493 -0.3296966  0.7555857 -7.651431
[11,]  7.9197072  6.1014349 10.0425287 -4.5676899 -1.5692786  0.340268
[12,] -0.5544463 -0.8423616 -1.8759387 -3.0333163 -1.9858746 -8.692907
             sh         prec     meanfun     minfun    meandom     mindom
 [1,] -3.4254504 -5.589857209   3.2526870  -1.171805  0.1486528 13.3012140
 [2,]  4.3300590 -9.598518839   9.9010218   4.132047 10.5008830 17.4033954
 [3,]  1.9281122  1.456739908   4.7199488  -4.061788  0.4078934 14.9565552
 [4,] -6.1158154  0.001394542   6.5301834  -2.431810 -2.8417556 12.0046692
 [5,]  2.7445453  6.730958610  -0.4557016 -12.617306 -4.3497170  8.1417027
 [6,] -4.1950520 -7.159249282   4.8806369 -13.325389 -5.2246208  8.2967867
 [7,]  0.4207524 -4.807737997 -11.5276607  -5.673415  0.7048216  6.4339940
 [8,] -5.6058899  4.690492208  -9.0788104  -2.196705 -4.6599762  2.1922283
 [9,] -4.5281344  1.964586231  -7.7697213  -9.807704  0.8704886  0.3802329
[10,] -0.7237214 12.558625913  -6.5326919  -5.319211 -2.4049458 -1.5195758
[11,]  8.4233051  7.222046510  -6.6567882 -10.805694  6.2766950 -1.9826413
[12,] -3.1345126  7.112813949  -2.5208326  -8.678459 -5.1222461 -4.7456389
         maxdom
 [1,]  2.3137295
 [2,] 10.6591428
 [3,] -3.0028746
 [4,] -3.3971310
 [5,] -4.9415422
 [6,] -2.9269086
 [7,] -0.6771455
 [8,] -6.1297493
 [9,] -2.3974406
[10,] -7.6074678
```

```
[11,]  4.3375244
[12,] -4.4410566
```

we can see that it doesn't tell us a whole lot more than the importance table did for random forests.

Is there a possibility that this data is overfit or that we have too much confusing information? Let's reduce the data set to the components that were most informative in our previous build.

```
train2 <- train %>% dplyr::select(IQR,Speaker,meanfun)
test2 <- test %>% dplyr::select(IQR,Speaker,meanfun)
#Might as well standardize these
mIQR <- mean(voice$IQR)
sdmIQR <- sd(voice$IQR)
mmeanfun <- mean(voice$meanfun)
sdmeanfun <- sd(voice$meanfun)
train2<-train2%>%
  mutate(IQR = (IQR-mIQR)/sdmIQR,
         meanfun = (meanfun-mmeanfun)/sdmeanfun)
test2 <- test2%>%
  mutate(IQR = (IQR-mIQR)/sdmIQR,
         meanfun = (meanfun-mmeanfun)/sdmeanfun)
```

```
set.seed(2250) #reproducability
rf <- randomForest(Speaker ~ . , data = train2, importance = TRUE)
svm <- svm(Speaker ~ ., data = train2)
```

```
estimate1 <- predict(rf, newdata = test2)
misclass1 <- ifelse(estimate1 != test2$Speaker,1,0)
mean(misclass1)
```

```
[1] 0.6923077
```

This seems to be a step in the wrong direction for the Random Forests model, which is now misclassifying 69% of the testing observations. How about the SVM?

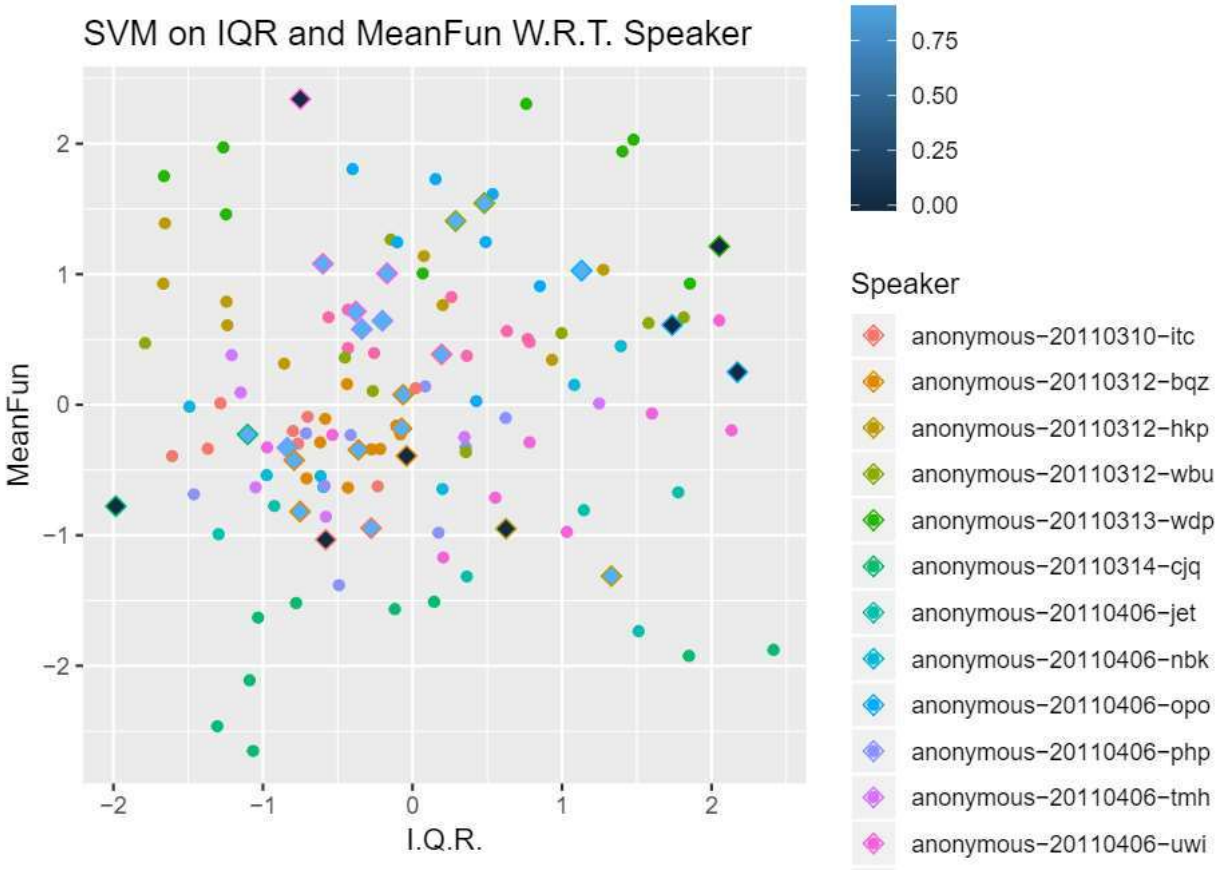
```
estimate2 <- predict(svm,newdata = test2)
misclass2 <- ifelse(estimate2 != test2$Speaker,1,0)
mean(misclass2)
```

```
[1] 0.6538462
```

While the SVM is now performing better than the Random Forest, neither can be said to be performing particularly well, certainly not at the level of the previous experiment. It is misclassifying 65% of the samples. Let's visualize this.

```
estimatedf2 <- as.data.frame(estimate2)
testdf <- test2 %>% dplyr::select(-Speaker)
estimatedf2 <- cbind(estimatedf2,testdf)
```

```
g<- ggplot(data = train2, aes(x = IQR, y = meanfun, colour = Speaker))+
  labs(x = "I.Q.R.", y = "MeanFun", colour = "Speaker") +
  ggtitle("SVM on IQR and MeanFun W.R.T. Speaker")+
  geom_point()+
  geom_point(
    data = estimatedf2,
    aes(x=meanfun,y=IQR,colour=estimate2,fill=misclass1),
    shape = 23,size = 2.5)
g
```



This is obviously a very chaotic graph, it is no surprise that the SVM is having a difficult time constructing its line. There are many possible issues contributing to this, but a likely one is that there are too many classes. Let's see how things go in a "one vs rest" scenario, since SVM is supposed to operate best under the condition of a binary response variable anyway.

```
train3 <- train %>% mutate(Speaker = ifelse(Speaker == "anonymous-20110312-wbu",
                                            "anonymous-20110312-wbu",
                                            "Not anonymous-20110312-wbu"))
test3 <- test %>% mutate(Speaker = ifelse(Speaker == "anonymous-20110312-wbu",
                                          "anonymous-20110312-wbu",
                                          "Not anonymous-20110312-wbu"))
```

```
set.seed(2250) #reproducability
rf <- randomForest(as.factor(Speaker) ~ . ,
                   data = train3, importance = TRUE)
svm <- svm(as.factor(Speaker) ~ ., data = train3)
```

```
estimate1 <- predict(rf, newdata = test3)
misclass1 <- ifelse(estimate1 != test3$Speaker,1,0)
mean(misclass1)
```

```
[1] 0.03846154
```

The misclass rate has dropped dramatically for the Random Forest. How does the SVM fare?

```
estimate2 <- predict(svm,newdata = test3)
misclass2 <- ifelse(estimate2 != test3$Speaker,1,0)
mean(misclass2)
```

```
[1] 0.07692308
```

The misclass rate is twice as high as the one for Random Forest. However, the fact that it is exactly twice as high clues us into a very simple error that has occurred.

```
tally(test3$Speaker)
```

```
X
    anonymous-20110312-wbu Not anonymous-20110312-wbu
                         2                         24
```

```
2/26
```

```
[1] 0.07692308
```

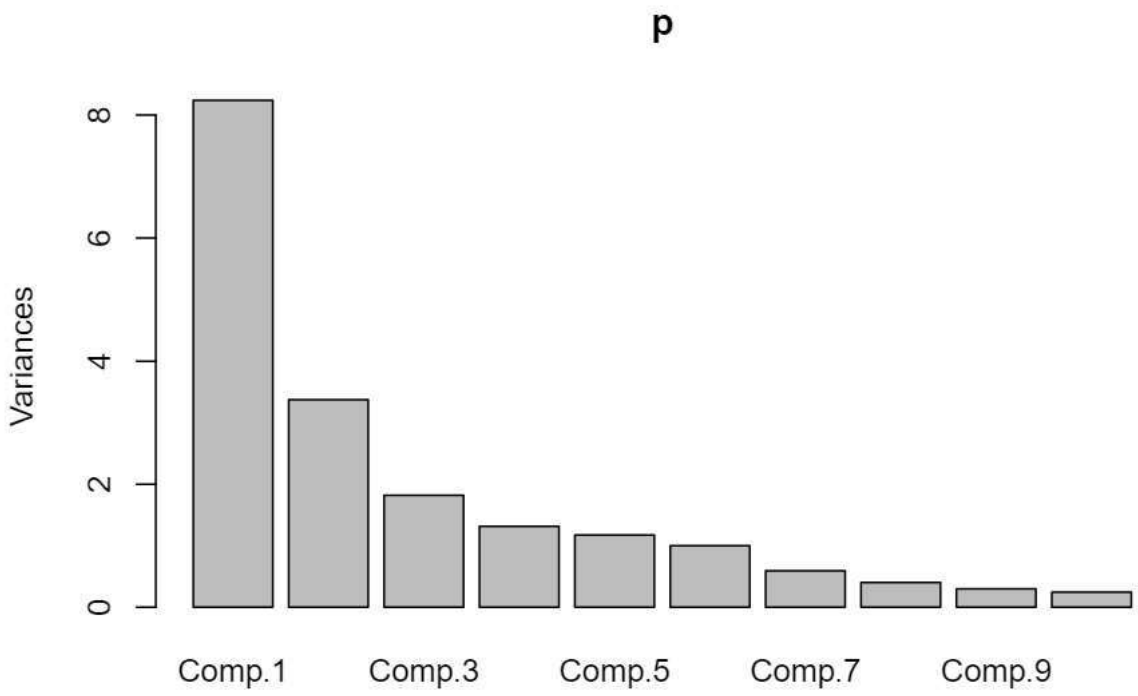
Under these conditions, with so few points in the test and train set belonging to our target group and so many falling into "other", both programs are learning primarily to just guess the "other" group. This illustrates the basic fact that it becomes increasingly difficult to isolate a single voice out of those you know for every new voice that you have to compare it to. It is likely that these models would have better performance if we had either less individuals to differentiate between or more observations from each individual and this would be the most logical next step for this experiment.

Finally, let's run the PCR and see if that improves upon anything.

```r
#from Professor Wagaman's hmk3Stat240
#assignment in Spring 2019
trainforprincomp <- train %>% dplyr::select(-Speaker)
p <- princomp(trainforprincomp, cor = TRUE, scores = TRUE)
msummary(p)
```

```
Importance of components:
                          Comp.1    Comp.2     Comp.3     Comp.4
Standard deviation     2.8700245 1.8367867 1.34820305 1.14770990
Proportion of Variance 0.4335285 0.1775676 0.09566587 0.06932832
Cumulative Proportion  0.4335285 0.6110961 0.70676198 0.77609030
                           Comp.5     Comp.6     Comp.7     Comp.8
Standard deviation     1.08303736 1.00101476 0.76898711 0.63377870
Proportion of Variance 0.06173526 0.05273845 0.03112322 0.02114081
Cumulative Proportion  0.83782555 0.89056401 0.92168723 0.94282804
                          Comp.9    Comp.10   Comp.11     Comp.12
Standard deviation     0.5496674 0.49381201 0.4781813 0.401691211
Proportion of Variance 0.0159018 0.01283423 0.0120346 0.008492412
Cumulative Proportion  0.9587298 0.97156406 0.9835987 0.992091072
                           Comp.13     Comp.14     Comp.15      Comp.16
Standard deviation     0.258870577 0.232164225 0.140535645 0.0787215417
Proportion of Variance 0.003527051 0.002836854 0.001039488 0.0003261622
Cumulative Proportion  0.995618123 0.998454978 0.999494465 0.9998206274
                           Comp.17      Comp.18 Comp.19
Standard deviation     0.0583787503 2.010449e-08       0
Proportion of Variance 0.0001793726 2.127319e-17       0
Cumulative Proportion  1.0000000000 1.000000e+00       1
```

```r
screeplot(p)
```




```
l <- loadings(p)
l[,1]
```

```
      mean         sd     median        sem       mode        Q25
 0.34579936  0.29164361  0.28025879  0.26273602  0.01397803  0.21852440
       Q75        IQR       cent    skewness    kurtosis         sfm
 0.33164140  0.31448064  0.34579936  0.09191439  0.16113812  0.28887634
        sh        prec     meanfun      minfun     meandom      mindom
 0.31989551  0.06445793 -0.02739661 -0.04713311  0.13181014  0.06230263
    maxdom
 0.15100703
```

As is typical for PCR, the interpreting the loadings of the PCs in terms of the original variables is a tricky business, but it seems odd that PC1, which explains most of this variance, does not include meanfun in any great capacity despite it being our most indicative statistic. Let's run our SVM model on the first three PCs and see how they perform.

```
set.seed(2250) #reproducability
svm <- svm(as.factor(Speaker) ~ p$scores[,1] +
             p$scores[,2] + p$scores[,3],
           data = train)
```

```
estimate2 <- predict(svm,newdata = test)
misclass2 <- ifelse(estimate2 != test$Speaker,1,0)
mean(misclass2)
```

```
[1] 0.9903846
```

This model is performing worse than random guesswork, so it seems in this case that, despite the high correlation values between some of the original variables, PCR proves not to be helpful in conjunction with our SVM model.

Conclusion

The project demonstrates that, although certain features of voice (such as the biological Sex of the speaker) are eminently identifiable (models scoring 98% and 99% accuracy) the problem of Voice Recognition appears to be more involved. While we were able to easily replicate the results of the experiment differentiating voice data by biological Sex for both the Random Forest and the SVM classification methods, neither method worked nearly as well when we were attempting to classify the samples by the individual speaker (attaining accuracies on the full data set of between 50% and 55%).

We would be surprised if they did since Voice Recognition is an ongoing area of research, but we were surprised to see that the method of SVM classification, which is a forerunning method in this field, was outperformed by the random forests on this data set. With reference to some additional experimentation, we posit that the 10 voice samples per speaker that were available from VoxForge were an insufficient amount to train either model to that specific person's voice. We also demonstrated, however, that having more speaker's samples in the data set overall correlates with a lesser degree of accuracy for any one voice.

An optimal continuation of this experiment would be experiments with the two methods on a data set having more than 10 observations for each voice. However, that will require additional scraping, possibly from another site than VoxForge.

Such models might also be less reflective of the real-life conditions under which some Voice Recognition problems arise. While, in biometric security, a given Voice Recognition lock would have many samples of its primary user's voice and could be loaded with many other voice samples to prevent its overfitting to that voice, the sample size will likely be much smaller in legal cases. A model could be trained on many preloaded samples but the sample for the voice of interest might consist of only a single piece of recorded evidence.

As for the scope of this project, the results of the preliminary experiment can probably be expanded to human voices in general since, even though Becker's dataset is a mashup of voices from several different sources and we don't know how the voices

were sampled, we do know that our result has been generally shown true across other experiments; the frequency range of the biologically male voice, on average and without intentional falsification, occupies a lower range than that of the biologically female voice. The accuracy of the models in the second experiment leaves something to be desired and seems to confirm the general consensus that Voice Recognition is not a definitively solved problem as of yet. Altogether, the project serves as an introductory investigation into the method of SVM classification, spectrogram measurements, and the field of Voice Recognition.

Bibliography

Introduction

1. Patel, Savan. "Chapter 2 : SVM (Support Vector Machine) - Theory – Machine Learning 101 – Medium." Medium.com, Medium, 3 May 2017, medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72.

2. "Support Vector Machine." Wikipedia, Wikimedia Foundation, 3 Jan. 2019, en.wikipedia.org/wiki/Support_vector_machine.

3. "Support Vector Machines(SVM) - An Overview – Towards Data Science." Towards Data Science, Towards Data Science, 16 June 2018, towardsdatascience.com/https-medium-com-pupalerushikesh-svm-f4b42800e989.

4. "Simple Tutorial on SVM and Parameter Tuning in Python and R." HackerEarth Blog, HackerEarth, 1 May 2017, www.hackerearth.com/blog/machine-learning/simple-tutorial-svm-parameter-tuning-python-r/.

5. "{[{Item.title | Translate}]}." Chrome Music Lab, musiclab.chromeexperiments.com/Spectrogra

6. "Spectrogram." Wikipedia, Wikimedia Foundation, 5 Feb. 2019, en.wikipedia.org/wiki/Spectrogram.

7. Becker, Kory. "Gender Recognition by Voice." RSNA Pneumonia Detection Challenge | Kaggle, 26 Aug. 2016, www.kaggle.com/primaryobjects/voicegender/home.

PreliminaryAnalysis

1. Becker, Kory. "Gender Recognition by Voice." RSNA Pneumonia Detection Challenge | Kaggle, 26 Aug. 2016, www.kaggle.com/primaryobjects/voicegender/home.

2. "Share and Discover Research." ResearchGate, https://www.researchgate.net/post/How_can_I_find_the_w_coefficients_of_SVM.

ScrapingVoxForge

1. Samcarcagno. "Basic Sound Processing with R." Sam Carcagno's Blog, 3 July 2016, samcarcagno.altervista.org/blog/basic-sound-processing-r/?doing_wp_cron=1548891484.3526279926300048828125.

2. Araya-Salas, Marcelo. "Choosing the Right Method for Measuring Acoustic Signal Structure." Bioacoustics in R, marce10.github.io/2017/02/17/Choosing_the_right_method_for_

3. Repository.voxforge1.Org, www.repository.voxforge1.org/.

VoiceRecognition

1. "Share and Discover Research." ResearchGate, https://www.researchgate.net/post/How_can_I_find_the_w_coefficients_of_SVM.

Additional

The coding makes free and liberal reference to R's published notes for its packages, a variety of debugging questions on StackExchange, Professor Wagaman's lab examples from the Stat495 class of Fall 2018, and previous work I have done for Statistics classes at Amherst.

CodeBook

PreLim Code

```
library(mosaic)
library(dplyr)
library(randomForest)
library(glmnet)
library(e1071)
```

```
voice = read.csv("/home/class19/gyoung19/StatComps/voice.csv")
```

```
msummary(voice)
voice2 <- voice %>% dplyr::select(-label)
cor(voice2)
```

```
set.seed(36) #for reproducability
train <- voice %>%
  sample_frac(0.80) %>%
  mutate(label = as.factor(label))

test <- voice %>%
  setdiff(train) %>%
  mutate(label = as.factor(label))
```

```
set.seed(2250) #reproducability
rf <- randomForest(label ~ . , data = train, importance = TRUE) #produce random f
svm <- svm(label ~ ., data = train)
```

```
estimate1 <- predict(rf, newdata = test) #assess accuracy of model on "new" data
misclass1 <- ifelse(estimate1 != test$label,1,0) #create vector indicating where miscl
mean(misclass1) #get percentage of misclassifications on test set
```

```
importance(rf) #list variables and importance
```

```
estimate2 <- predict(svm,newdata = test)
misclass2 <- ifelse(estimate2 != test$label,1,0)
mean(misclass2)
```

```
#code from https://www.researchgate.net/post/How_can_I_find_the_w_coefficients_of_SVM
W <- t(svm$coefs) %*% svm$SV
W
```

```
train2 <- train %>% dplyr::select(meanfun, IQR, label)
test2 <- test %>% dplyr::select(meanfun, IQR, label)
```

```
set.seed(2250) #reproducability
rf <- randomForest(label ~ . , data = train2, importance = TRUE) #produce random forest
svm <- svm(label ~ ., data = train2)
```

```
estimate1 <- predict(rf, newdata = test2) #assess accuracy of model on "new" data
misclass1 <- ifelse(estimate1 != test2$label,1,0) #create vector indicating where misc
mean(misclass1) #get percentage of misclassifications on test set
```

```
estimate2 <- predict(svm,newdata = test2)
misclass2 <- ifelse(estimate2 != test2$label,1,0)
mean(misclass2)
```

```
estimatedf2 <- as.data.frame(estimate2)
testdf <- test2 %>% select(-label)
estimatedf2 <- cbind(estimatedf2,testdf)
```

```
g<- ggplot(data = train, aes(x = IQR, y = meanfun, colour = label))+
  labs(x = "I.Q.R.", y = "MeanFun", colour = "Label") +
  ggtitle("SVM on IQR and MeanFun W.R.T. Label")+
  geom_point()+
  geom_point(data = estimatedf2, aes(x=meanfun,y=IQR,colour=estimate2,fill=miscla
g
```

ScrapingVoxForge Code

```
knitr::opts_chunk$set(echo = TRUE)
#install.packages("e1071")
#install.packages("caret")
library(mosaic)
library(e1071)
library(seewave)
library(tuneR)
```

```
path <- 'C:/Users/Gabriel Young/Documents/StatComps/VoxForge/anonymous-20110310-it
spectro1 <- readWave(path)
```

```
#Code for pulling stats from a single wav file

#isolate the signal
signal <- spectro1@left

#obtain signal duration
dur = length(signal)/spectro1@samp.rate

#get the sampling rate of the signal
fs = spectro1@samp.rate

# center to remove DC offset
signal = signal - mean(signal)
```

```
#obtain the spec object
s <- spec(spectro1,fs)

#get frequency properties we want using specprop
props <- as.data.frame(specprop(s,fs))

#get fundamental frequencies, eliminate NaNs
f <- as.data.frame(fund(spectro1,fs))
f2 <- f[complete.cases(f),]

#get summary stats of fundamental frequency measures
meanfun <- mean(f2$y)
minfun <- min(f2$y)
maxfun <- max(f2$y)
props <- cbind(props,meanfun, minfun, maxfun)

#get dominant frequencies, eliminate NaNs
d <- as.data.frame(dfreq(spectro1,fs))
d2 <- d[complete.cases(d),]

#get summary stats of dominant frequencies
meandom <- mean(d2$y)
mindom <- min(d2$y)
maxdom <- max(d2$y)

#add to data
props <- cbind(props,meandom,mindom,maxdom)
```

```
path <- 'C:/Users/Gabriel Young/Documents/StatComps/VoxForge/'
folders <- as.data.frame(list.files(path))
folders <- folders[4:16,]
folders <- as.vector(folders)


#initialize storage dataframe
desiredstats <- data.frame()
```

```
#apply spec processes to all files to obtain descriptive statistics that we want
for(folder in folders)
{
  files <- as.data.frame(list.files(paste(path, folder, "/wav", sep = "")))
  files <- files %>% mutate(names = as.character(`list.files(paste(path, folder,
  realfiles = files$names
  for(f in realfiles)
  {
    spectro1 <- readWave(filename = paste(path,folder,"/wav/",f,sep=""))
    signal <- spectro1@left

    dur = length(signal)/spectro1@samp.rate

    fs = spectro1@samp.rate

    # demean to remove DC offset
    signal = signal - mean(signal)

    s <- spec(spectro1,fs)
    props <- as.data.frame(specprop(s,fs))
    props <- props %>% mutate(Files = f, Folders = folder)

    f <- as.data.frame(fund(spectro1,fs))
    f2 <- f[complete.cases(f),]
    meanfun <- mean(f2$y)
    minfun <- min(f2$y)
    maxfun <- max(f2$y)

    d <- as.data.frame(dfreq(spectro1,fs))
    d2 <- d[complete.cases(d),]
    meandom <- mean(d2$y)
    mindom <- min(d2$y)
    maxdom <- max(d2$y)

    props <- cbind(props,meanfun, minfun, maxfun)
```

```
    props <- cbind(props,meandom,mindom,maxdom)

    desiredstats <- rbind(desiredstats, props)
  }
}
```

```
write.csv(desiredstats, file = "VoiceRecogData.csv")
```

Voice Recognition Code

```
library(mosaic)
library(dplyr)
library(randomForest)
library(glmnet)
library(e1071)
```

```
# from https://www.r-bloggers.com/machine-learning-using-support-vector-machines/
mod_svm <- svm(as.factor(Folders)~ ., desiredstats2)
```

```
set.seed(36) #for reproducability
train <- voice %>%
  sample_frac(0.80) %>%
  mutate(Speaker = as.factor(Speaker))

test <- voice %>%
  setdiff(train) %>%
  mutate(Speaker = as.factor(Speaker))
```

```
set.seed(2250) #reproducability
rf <- randomForest(Speaker ~ . , data = train, importance = TRUE) #produce random fores
svm <- svm(Speaker ~ ., data = train)
```

```
estimate1 <- predict(rf, newdata = test) #assess accuracy of model on "new" data
misclass1 <- ifelse(estimate1 != test$Speaker,1,0) #create vector indicating wher
mean(misclass1) #get percentage of misclassifications on test set

importance(rf) #list variables and importance

estimate2 <- predict(svm,newdata = test)
misclass2 <- ifelse(estimate2 != test$Speaker,1,0)
mean(misclass2)

#code from https://www.researchgate.net/post/How_can_I_find_the_w_coefficients_o
W <- t(svm$coefs) %*% svm$SV
W

train2 <- train %>% dplyr::select(IQR,Speaker,meanfun,meandom)
test2 <- test %>% dplyr::select(IQR,Speaker,meanfun,meandom)

set.seed(2250) #reproducability
rf <- randomForest(Speaker ~ . , data = train2, importance = TRUE) #produce rando
svm <- svm(Speaker ~ ., data = train2)

estimate1 <- predict(rf, newdata = test2) #assess accuracy of model on "new" data
misclass1 <- ifelse(estimate1 != test2$Speaker,1,0) #create vector indicating whe
mean(misclass1) #get percentage of misclassifications on test set

estimate2 <- predict(svm,newdata = test2)
misclass2 <- ifelse(estimate2 != test2$Speaker,1,0)
mean(misclass2)

estimatedf2 <- as.data.frame(estimate2)
testdf <- test2 %>% dplyr::select(-Speaker)
estimatedf2 <- cbind(estimatedf2,testdf)
```

```
g<- ggplot(data = train2, aes(x = IQR, y = meanfun, colour = Speaker))+
  labs(x = "I.Q.R.", y = "MeanFun", colour = "Speaker") +
  ggtitle("SVM on IQR and MeanFun W.R.T. Speaker")+
  geom_point()+
  geom_point(data = estimatedf2, aes(x=meanfun,y=IQR,colour=estimate2,fill=misclass1),s
g
```

```
train3 <- train %>% mutate(Speaker = ifelse(Speaker == "anonymous-20110312-wbu","anonym
test3 <- test %>% mutate(Speaker = ifelse(Speaker == "anonymous-20110312-wbu","anonymou
```

```
set.seed(2250) #reproducability
rf <- randomForest(as.factor(Speaker) ~ . , data = train3, importance = TRUE) #produce
svm <- svm(as.factor(Speaker) ~ ., data = train3)
```

```
estimate1 <- predict(rf, newdata = test3) #assess accuracy of model on "new" data
misclass1 <- ifelse(estimate1 != test3$Speaker,1,0) #create vector indicating where mis
mean(misclass1) #get percentage of misclassifications on test set
```

```
estimate2 <- predict(svm,newdata = test3)
misclass2 <- ifelse(estimate2 != test3$Speaker,1,0)
mean(misclass2)
```

```
tally(test3$Speaker)
2/26
```

```
#from Professor Wagaman's hmk3Stat240 assignment in Spring 2019
trainforprincomp <- train %>% dplyr::select(-Speaker)
p <- princomp(trainforprincomp, cor = TRUE, scores = TRUE)
msummary(p)
screeplot(p)
l <- loadings(p)
l[,1]
```

```r
set.seed(2250) #reproducability
svm <- svm(as.factor(Speaker) ~ p$scores[,1] + p$scores[,2] + p$scores[,3], data =
```

```r
estimate2 <- predict(svm,newdata = test)
misclass2 <- ifelse(estimate2 != test$Speaker,1,0)
mean(misclass2)
```