Instrument Classification Using Fourier Transforms of Sustained Sounds

Gabriel Chen

1: Introduction

Many different musical instruments exist, and each have their own unique sound qualities that make it identifiable to the human ear. The act of classifying the sound of an instrument is to identify which instrument a sound originated from. The act of classification can become difficult to do reliably and quickly as a human being, as humans are limited in their ability to process the physical phenomenon of sound.

When identifying musical instruments, it's easy to identify the sound in the context of the entirety of a note being played. For example, a violin's sound typically does not abruptly start at it's highest volume. Or a piano note naturally cannot increase in volume during the sustaining of a note. If very short samples are played of a set of instruments that exclude these contextual identifiers, it becomes less trivial to identify the origin of the sample. Subjectivity and generality becomes a disadvantage of the human instrument classification system, as what comes in is interpreted consciously in much less detail than what is actually present in the sound. Without temporal context, using only general feelings of tone becomes much less effective.

In addition, human interpretation of musical instruments is limited by the temporally bound act of listening to a sample. Listening to a sample of length one second to identify what instrument the sound belongs to takes one second to listen to. Then the act of discerning which instrument is being played, from a set of instruments, increases with the amount of instruments within the set by an exponential amount. The more instruments are present, the more comparisons need to be made, but the number of comparisons per instrument would also increase with the increase of the set–assuming a consistent amount of accuracy. A person would not be able to clearly maintain the information heard in a sample for the duration of processing all of the samples for efficient comparison.

Even in ideal conditions, classifying a sample set of thousands of sample would take at least 1000 times the length of each sample. Instead of a human classification system, utilizing machine learning to build a model based on information taken from sound samples could be far more efficient. The temporal disadvantage of human interpretation of physical events would be remedied by the ability of a computer to process physical events as data at a much quicker rate. The goal of this project is to build such a classifier that is able to classify instruments such as a grand piano, a clavinet, a rhodes electric piano, a bass guitar, a clarinet, and a violin.

2: Background Information

a. Machine Learning

The definition of machine learning can be marginally different between individuals in the industry, but for the purposes of this project, machine learning pertains to the design and usage of classifiers in the context of supervised learning–not to be confused with data mining which pertains to the usage of classifiers to cluster input patterns into groupings or patterns (or unsupervised learning). Supervised learning is defined as learning that involves training a model with information that has already processed with the intention of processing other data in the same way. Unsupervised learning is the opposite.

Figure 2.a.1: The above image describes machine learning (the usage of classifiers, and the designing of classifiers in the context of supervised learning.

The usage of a classifier is to take an input (feature vector), and classify it by associating the feature vector with an output label *[Figure 1: Classifier Usage (Upper)]*. A feature vector, as referenced in the image, is a vector that contains data about the features of the entity that is being potentially classified. Each feature is some value that represents a single characteristic that is going to be measured in and compared between each object that is used to train the classifier. For example, if the classifier determines gender based on the height of an individual, the feature vector would have a single entry that corresponds to the height of the individual. The class label in such a classifier would be the gender of the individual represented by the feature vector.

The design of a classifier can be conveniently engineered with the understanding of how it is intended to be used *[Figure 1: Classifier Design (Lower)]*. Since a classifier needs to be able to take feature vectors of specified length where each position in the vector represents a specific feature, the classifier needs to be built using vectors of the same characteristics. These vectors collectively are the training set. The training set is fed into a learning algorithm, which builds a model that can be queried with a feature vector with the intention of receiving a prediction for the vector's class label.

b. Relevant Learning Algorithm Information

There are many different types of learning algorithms, but for the purposes of this project, a naive bayesian classifier using Gaussian distributions was selected for the the classification of the instruments. To understand the learning algorithm, a few simple example models are useful.

A simple example of using probabilities for classification is a histogram model. The scenario in *Figure 2.b.1* represents imaginary data collected from people walking through airport security. If heights were taken for each individual, as well as their gender, the data graphed as histograms would look like such.

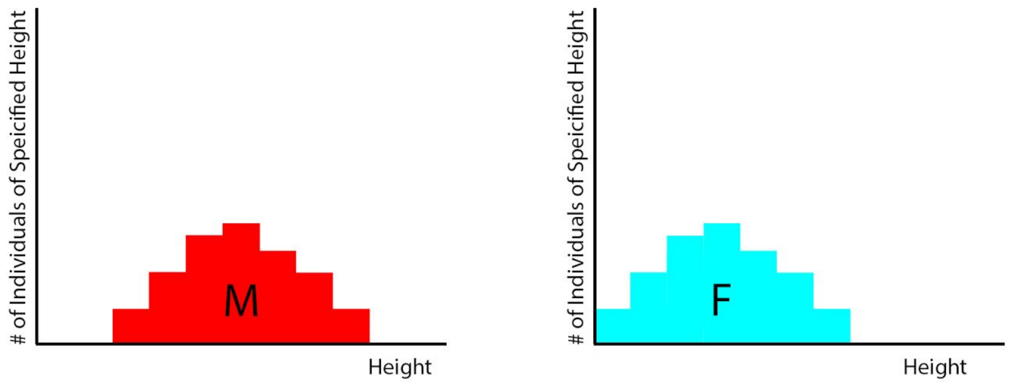


Figure 2.b.1: The above image shows two histograms. One (left) shows the number of male individuals counted at specific heights and the other (right) shows the number of female individuals counted at specific heights.

With the data represented in *[Figure 2.b.1]*, a prediction, based on height, of the gender of the individual can be made. To make a prediction on gender, the classifier would be queried with an individual represented by a feature vector that contains the feature, height. The resulting probability that the query is male would be the number of male individuals of the query height divided by the total number of recorded individuals of that height *[Figure 2.b.2]*.

H_m = male histogram
H_f = female histogram
Height = Query Vector

$P(\text{Male} \mid \text{Height}) = H_m(\text{Height}) / H_f(\text{Height}) + H_m(\text{Height})$
Figure 2.b.2: Histogram model predictions equation.

Using a histogram model obviously presents a few problems. If the data collected is not dense enough, (not enough heights are represented) then there could be queries where at certain heights, the number of individuals for that gender may be underrepresented for that height (for example, having no people of that height). It's important to note that a good representation is defined by a distribution that is ideal. In many scenarios, normal distributions can be used. Under representation in a sample set would result in an unrealistic prediction, since the surrounding heights may be more telling of the gender of the individual than the specified height in the query. In a sense these sorts of situations can be interpreted as noise in the data. To achieve better predictions, the data is not used directly in the model for predictions. Instead, the values of the features are put into bins that represent clusters of neighboring heights. These clusters will tend to create cleaner distributions with less noise. Cleaner distributions will allow for better predictions in heights that maybe have been underrepresented in the data collection process.

A histogram model is easily scalable. To add more labels, another histogram can be added and data for that histogram can be recorded while data for the other two were recorded. Queries would be made in the same manner *[Figure 2.b.3]*.

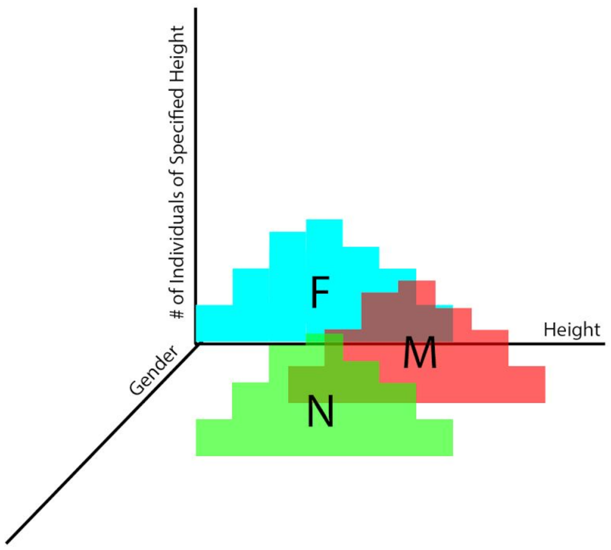


Figure 2.b.3: Three histograms, each visualizing the same type of data. Each histogram represents the recorded information about individuals, their heights and their genders.

To scale the histogram model to include more features, the histograms would just enter more dimensions. For example, if handspan was included as a query feature, rather than 2 dimensional histograms, instead three dimensional histograms with value associated with 2 dimensional bins would be used.

One of the pitfalls of a histogram type model, is that there are issues with noise and empty bins that can't be solved by adjusting feature size. If feature vectors with features that are not recorded are used for queries, the result would be that the model would not be able to discern the class label for the query. As an extreme example, if the model was queried for height 90 feet, the result would be probabilities 0 / 0 + 0 in the 1 feature models.

Instead of using a histogram model, a Gaussian model can be employed instead. Instead of histograms, the data associated recorded is instead fitted with a Gaussian distribution *[Figure 2.b.4]*.

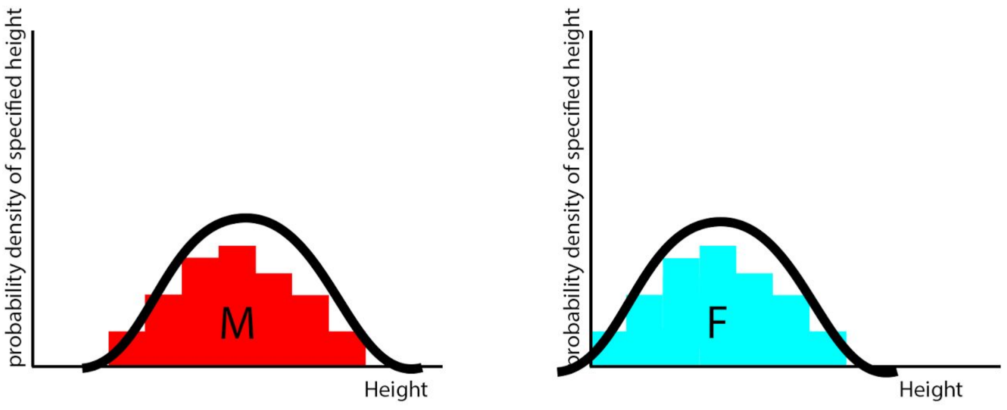


Figure 2.b.4: The top are the gaussian distributions fitted to the data in the previous 1 feature 2 class example. Notice how the Y axis becomes the probability density associated with the specified height as opposed to the number of people.

Fitting the gaussian distribution to the data is just a matter of solving for the mean and the standard deviation of the data set and using the formula for the probability density function of a normal distribution *[Figure 2.b.5]*.

$$y = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$\begin{aligned} \mu &= \text{ Mean} \\ \sigma &= \text{ Standard Deviation} \\ \pi &\approx 3.14159\cdots \\ e &\approx 2.71828\cdots \end{aligned}$$

Figure 2.b.5: The probability density function of a normal (Gaussian) distribution.

Querying the classifier built using a bayesian model is similar to querying the histogram classifier. The probability density of the specified feature vector in the context of one class is compared to the probability density of the specified feature in another class. The extra caveat is that since these are probability densities, they lose the information of their relevance to each other. In other words, it loses the information of the size of each data set compared to the other. To account for the loss of the size of each data set, the probability densities gotten from the probability density functions are multiplied by the size of the training set associated with its class label. For classifiers with more query features, the probability density function is generalized for more dimensions *[Figure 2.b.6]*.

$$\frac{1}{(2\pi)^{\frac{d}{2}}\sqrt{|\Sigma|}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})\,\Sigma^{-1}(\boldsymbol{x}-\boldsymbol{\mu})^T}$$

Figure 2.b.6: The probability function of a normal Gaussian distribution in n dimensions.
At more dimensions, the only thing that changes it the probability density function; the probability density function is generalized so as to account for more dimensions.

c. Relevant Information on Instrument Harmonics

Qualitatively, it is simple to know that different instruments are different and separated from each other by differences in tone. But as sound is a physical phenomenon, these differences can be quantitatively demonstrated. The discrete fourier transform is useful for quantifying the physical phenomenon of sound, as it shows the partials present and their amplitudes in a sound. These are quantitative references of the harmonic content that makes the sounds an instrument produces unique [2.c.r1]. These differences in quantities can be seen in the peaks of in the fourier transforms–heights and locations.

Even though the grand piano and the clavinet look really similar in the fourier transforms from 0 to 1000 Hz *[Figure 2.c.1]*, the harmonic content in the higher frequency range is very different with the clavinet having more present higher frequencies while the grand has relatively less *[Figure 2.c.2]*.

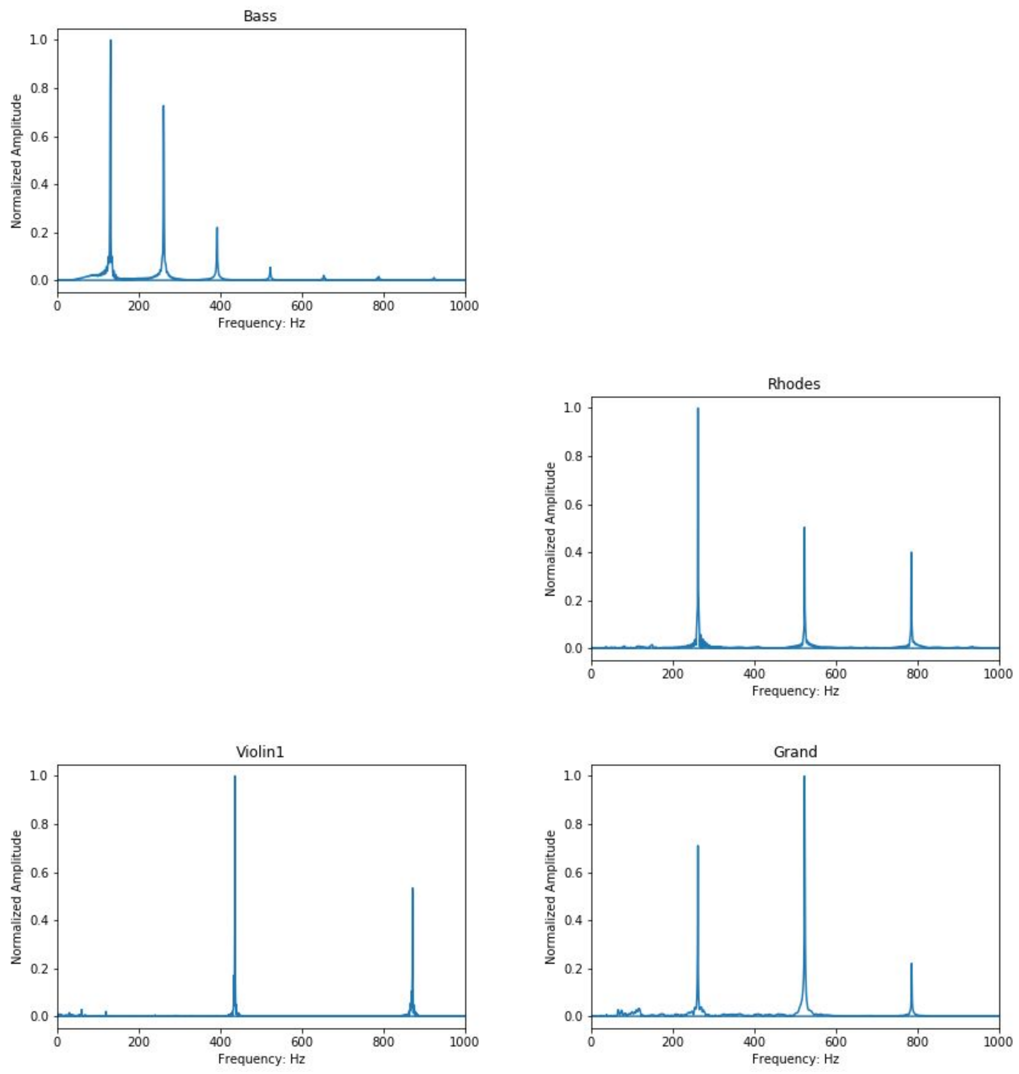


Figure 2.c.1: The fourier transforms show clear characteristics unique to each instrument.

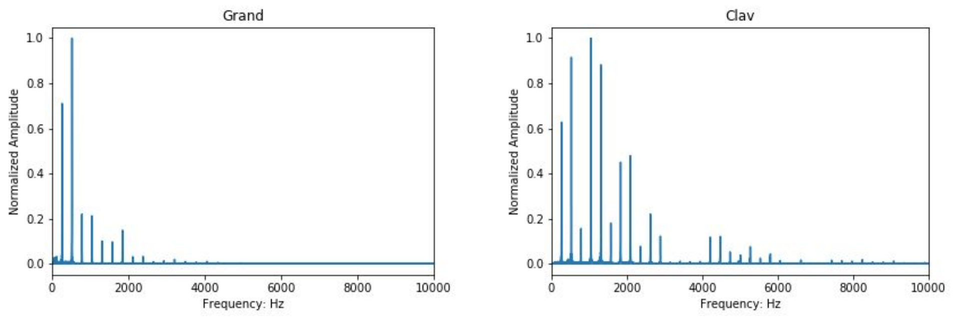


Figure 2.c.2: Larger frequency range fourier transform of grand piano and clavinet.

Figure 2.c.3: Comparison between Violin and Clarinet Frequency transform.

3. Methods

a. Overview

The training set and test set were acquired using virtual instrument software (violin and clarinet samples are exceptions), with sample specific sound engines in order to generate sounds as authentic as was available.

The samples were then processed into numerical, quantified data vectors with the instruments as class labels associated with each vector. Specific binning methods were used in order to achieve desired results.

A portion of the samples were then used as a training set, while the others were designated as the test set. The different combinations of samples were used as training sets to train classifiers, applying the naive bayesian model as the learning algorithm. Predictions of classes were made on the test set, and compared against the real labels of the test set to determine the accuracy of each classifier.

b. Data Set

Samples for grand piano, clavinet, electric piano, and bass guitar were collected using complex sample library software commercially available from Spectrasonics [3.b.r1], and Native

Instruments [3.b.r2]: Keyscape, and Kontakt 5 are the software used, associated with each company respectively *[Figure 3.b.1, Figure 3.b.2]*. The three keyboards, grand piano, clavinet, and rhodes electric piano were selected as a case where the three instruments are instruments of the same category. The purpose of selecting these three instruments was to present a classification example of three instruments that are similar in type, with similar types of sounds in their strong attack and slow decay of the sustained sound.

Figure 3.b.1: Keyscape software interface. Default settings were used for each instrument with the exception of extra effects such as tremolo which were removed.

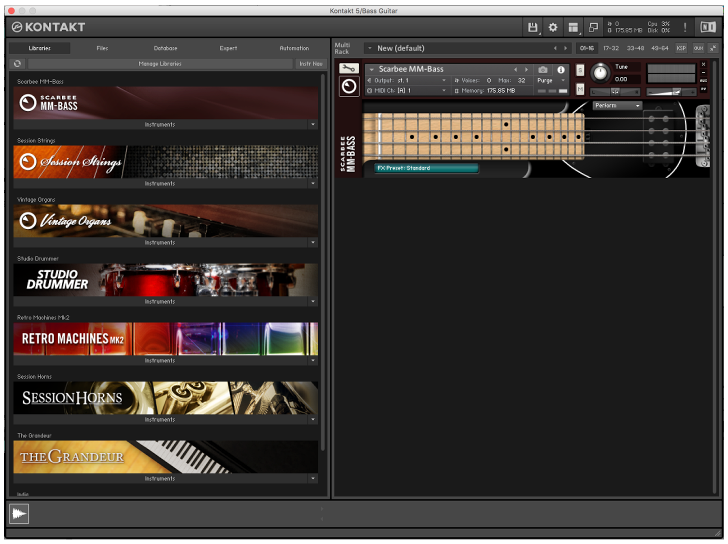


Figure 3.b.2: Kontakt 5 software interface. Default settings were used for the Bass guitar.

The keyboard sounds were sourced from Keyscape by triggering individual notes in digital audio workstation software, Ableton [3.b.r3]. Each sample is played in the octave between middle C and the next highest C (C5-C6). The same was done in Kontakt 5 to source the bass guitar samples. Note triggers were spaced evenly apart and each is the same length and later split apart into individual samples using Python Scipy.Io Wavfile package [3.b.r4]. These triggered sounds were then exported in the WAV [3.b.r5] format.

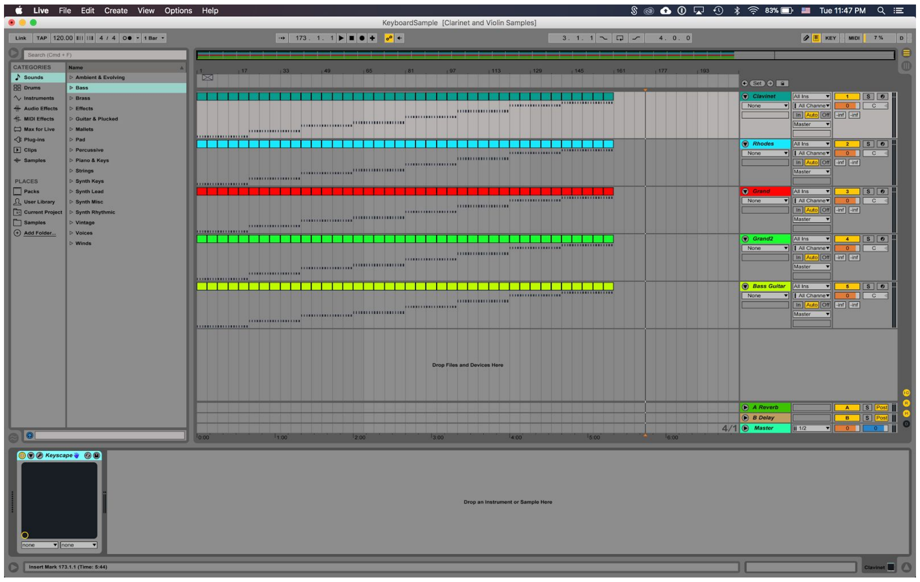

Live File Edit Create View Options Help
KeyboardSample [Clarinet and Violin Samples]
Clavinet
Rhodes
Grand
Grand2
Bass Guitar
A Reverb
B Delay
Master
Drop Files and Devices Here
Keyscape
Drop an Instrument or Sample Here

Figure 3.b.3: Ableton software used for triggering samples and exporting as WAV.

The samples for clarinet and violin were sourced from amateur musicians–recorded and processed in Ableton as well. The resulting sample set was much smaller, with only 35 samples per instrument compared to the 140 samples for each of the other instruments.

c. Preprocessing

i. Sample Digestion

A computer does not inherently know how to interpret sound. Software, computational instruction, has to be written in order allow the computer to make sense of sound. Python 3 [3.c.r1] was used as the primary programming language for this project, link to repository for the Jupyter notebook [3.c.r2, github.r]. Recordings of sounds are digital representations of the sounds in the form of numerical values. Hence, each sound file can be treated as a digital collection, or array, whose values describe the sound sample in a specific manner. In order to decode the sound sample into an array of numerical values, the Python Scipy library is utilized *[Figure 3.c.1]*.

Figure 3.c.1: code snippet for importing the sound file decoding methods.

Using the methods available to read Wav files, inputting the location of a sound file of compatible type .wav results in an array that is the numerical representation of the sound, and an integer that is the sampling rate of the sound file. The sampling rate is measured in Hz and is the number of numbers in the array per second of sound. It's important to understand that the array no longer has temporal relevance without extra interpretations of the locations within the array–the array's indices no longer represent time units. Most modern digital audio is sampled at 44100 Hz. In order to cut the file into appropriate lengths, 1 trigger per sample since each file contains multiple triggers of the same instrument, the array needs to be split at the desired sound locations corresponding to each trigger. A simple calculation can be done to find these splitting points *[Figure 3.c.2]*.

*44100 samples/second * desired number of seconds/split = number of samples/split*
Figure 3.c.2: Simple calculation for converting array indices back to time.

Each trigger lasts for 2 beats, and all triggers are is separated by 2 beat. The beats per minute of the recording is 120, therefore the starting point of every trigger is separated by 2 seconds to the next trigger. Therefore the desired number of seconds/split was 2 seconds. The described sample splitting method was run iteratively over the length of the collective sample in order to split it to the desired lengths.

The desired portion on sound to use excludes the attack and the decay *[Figure 3.c.3]*, so in a sample, anywhere in the middle of the sample can be taken to use in the training set and test set. Specifically, small duration portions of the middle region can be taken in order to not give away characteristics of the temporal development of an instrument's sustained notes.

Figure 3.c.3: The region within the blue lines is the desired region to take a sustained sample from. In the data used for training and testing, these ends were of length a specific and consistent length between samples.

Using the same method as before for converting array indices to time, samples were split even further into samples of size .1 seconds. Splitting the samples further gives the effect of having a larger sample size, as each portion of the sustained sample is slightly different. In addition, it also, as previously stated, removes the temporal context of the sample.

ii. Feature Vector Extraction

To extract a feature vector relevant to the instrument's characteristics that is within a reasonable length is critical to the viability of the classifier. The discrete fourier transform of the data is useful for distinguishing a instrument sample's characteristics. If the feature vectors are too long, it becomes computationally infeasible or outside of the limitations of human convenience, to compute a model that can make predictions. In addition, the act of making a prediction on large model is also computationally expensive.

If every frequency represented in the DFT was used (discrete fourier transform) the result would be a feature vector of length 22050–computationally infeasible. The reasons it becomes computationally expensive are outside of the scope of this project, but more can be read on computational complexity, as well as the math involved in a query.

In addition to being computationally expensive, it also is not suitable for the purposes of the classifier. The input vectors in a vector made from every frequency would include different pitches, but not all pitches. For example, the sample set does not include the sharp and flat notes in a scale. The partials and the harmonic information for the note would all be shifted by some amount that is relatively unpredictably affected by the temperament of the instrument's tuning [3.c.r3]. Therefore, making a classifier based on feature vectors generated from every or evenly spaced bins results in a classifier that classifies pitch as opposed to instrument. In order to account for the differences in pitch, or rather to discount the differences in pitch, the fourier transforms are instead binned using a different method.

Figure 3.c.4: Table showing the frequencies of pitches at each octave.

The table shows that each octave is separated by approximately a factor of 2. The spacing between octaves turns out to be a decent piece of information to use as an heuristic for the binning of the fourier transformation data. In order to ignore the pitch of the note being played, each bin encompasses an entire octave. Each bin is two times the width of the previous bin, and in each bin, the value corresponds to the value of the highest peak in the bin. The first bin starts at 16 Hz and is of length approximately 16 Hz. The resulting feature vector is of size 11, since the frequencies captured are from 0 Hz to 22050 Hz *[Figure 3.c.5]*.

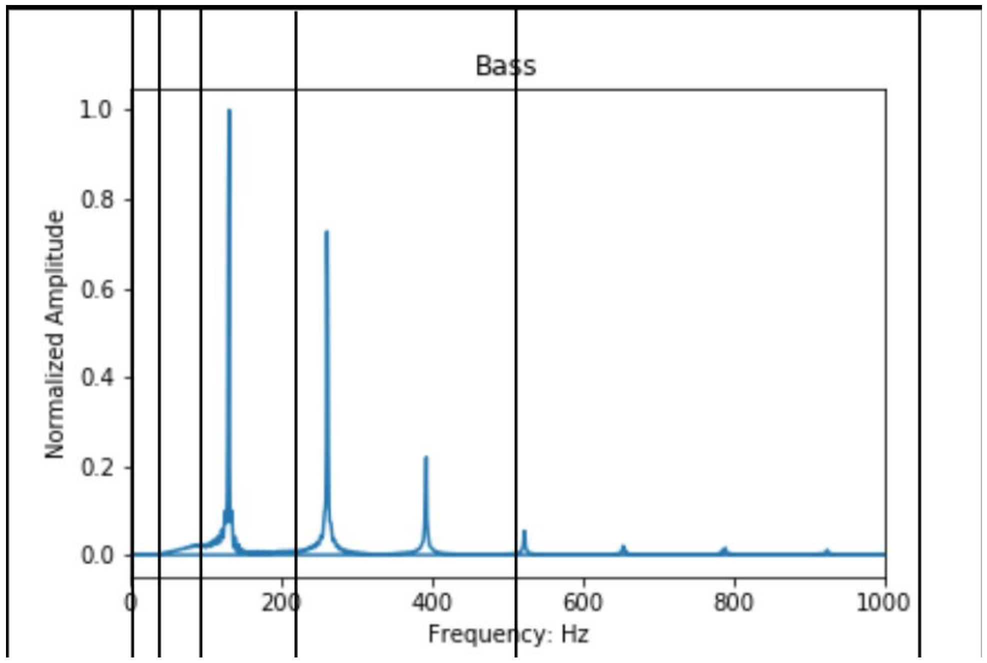


Figure 3.c.5: Each bin in the feature vector is approximately 2 times larger than the previous bin.

From the bins visualized in *[Figure 3.c.5]*, there are clear, visible disadvantages to an octave max heuristic for binning the fourier transform data. In the fourth bin from the left, the range approximately 220 to 520 Hz, there's a clear loss of data in that the bin contains two major peaks. Harmonic information in instruments is not represented by octaves, but instead is more closely related to the pitch of the note being played. The data available in the amplitude of the peaks, though, is enough to offset the disadvantage in most cases as can be seen later in the results.

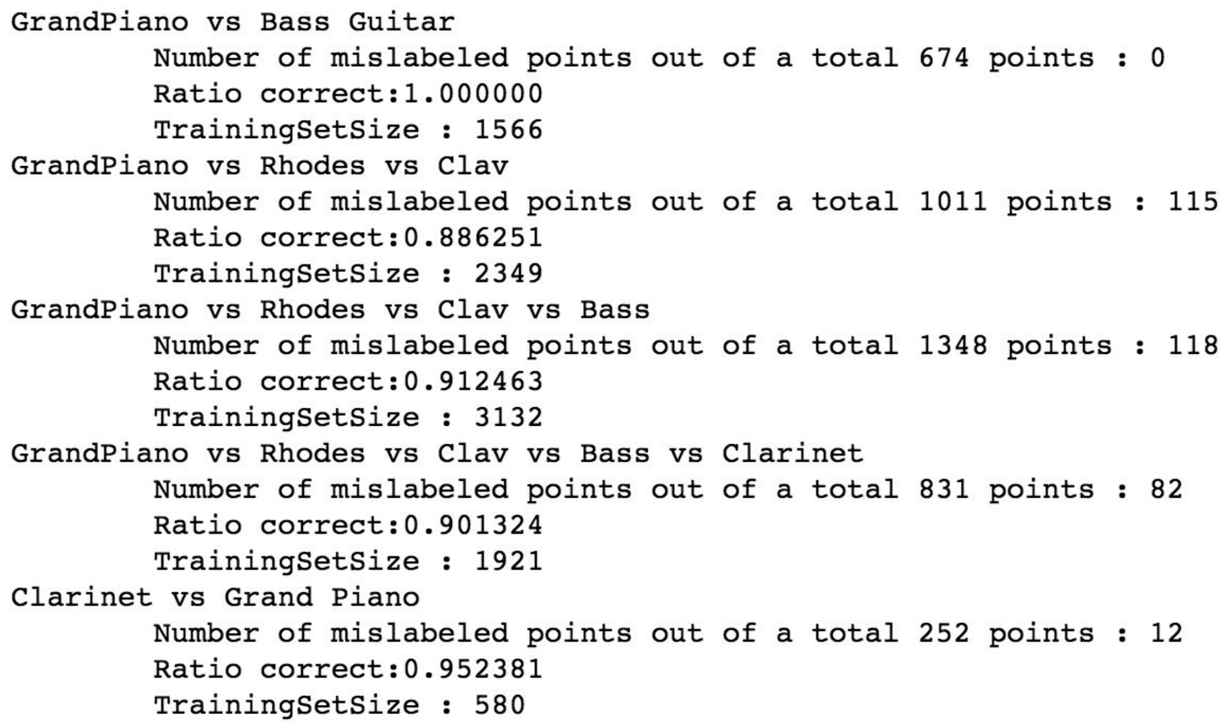
Another feature of a fourier transformation that needs to be accounted for in making a classifier is the amplitude of the peaks. In order for a consistent model that doesn't take the actual amplitude of the peaks into account, but instead the relative amplitude of the peaks, and the locations of the peaks, the heights of the peaks need to be normalized and put in a constant scale. Here, all of the peaks in a fourier transform are based on the height of the highest peak in that fourier transform.

These methods were applied to all of the cut up samples to generate sample sets that consist of feature vectors, each with a class label. These vectors were recorded in .csv files that could be selected to train classifiers, and used to test classifiers.

4. Results

The sample sets were selected to train a variety of different classifiers. In each case, sample sets corresponding to instruments were selected, and 70% of the samples from each instrument's sample set were taken and used as a training set while the other 30% were used as a test set. The classifier made predictions on the feature vectors of the test set, and these results were compared to the actual class labels associated with the test set feature vectors to determine the accuracy of the classifier.

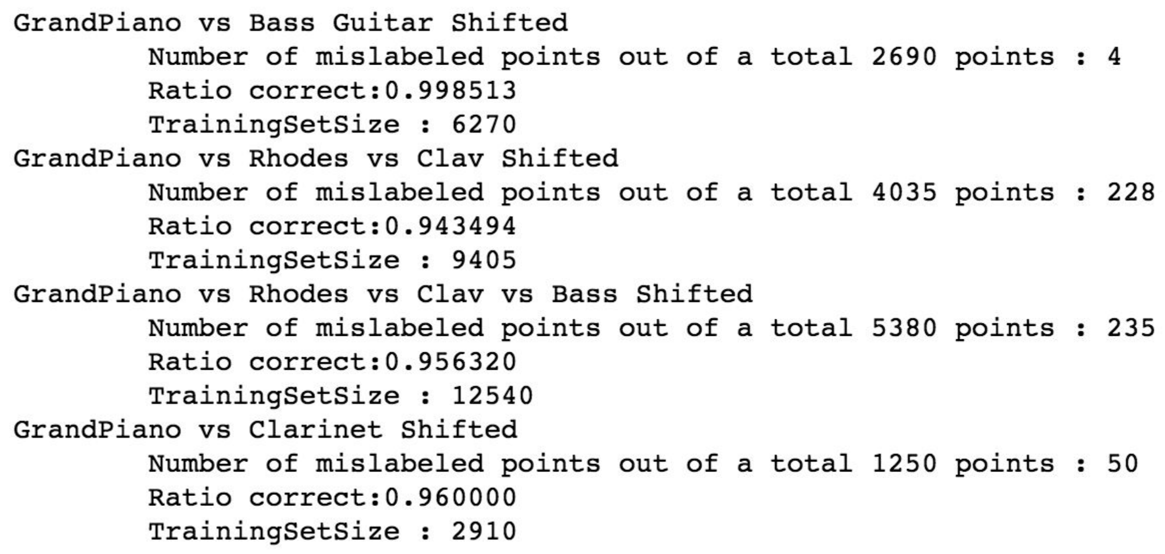
[Figure 4.1] Below shows the classifiers built using the sample sets and their accuracies. Grand piano and bass guitar were selected as a base case, as the two instruments are the most different among the instruments that have were sourced from instrument software, which means they were the most cleanly generated samples. Next, the three instruments that were the most similar in terms of instrument category were compared: grand piano, rhodes electric piano and the clavinet.



```
GrandPiano vs Bass Guitar
        Number of mislabeled points out of a total 674 points : 0
        Ratio correct:1.000000
        TrainingSetSize : 1566
GrandPiano vs Rhodes vs Clav
        Number of mislabeled points out of a total 1011 points : 115
        Ratio correct:0.886251
        TrainingSetSize : 2349
GrandPiano vs Rhodes vs Clav vs Bass
        Number of mislabeled points out of a total 1348 points : 118
        Ratio correct:0.912463
        TrainingSetSize : 3132
GrandPiano vs Rhodes vs Clav vs Bass vs Clarinet
        Number of mislabeled points out of a total 831 points : 82
        Ratio correct:0.901324
        TrainingSetSize : 1921
Clarinet vs Grand Piano
        Number of mislabeled points out of a total 252 points : 12
        Ratio correct:0.952381
        TrainingSetSize : 580
```

Figure 4.1: Accuracy results of classifiers built to classify different sets of instruments.

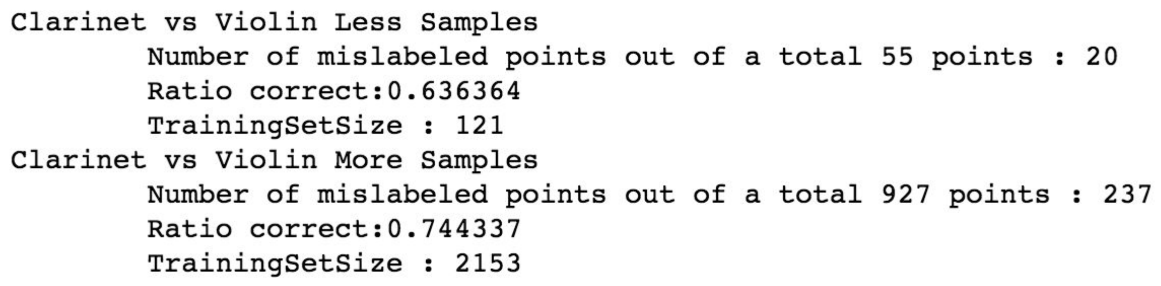
In order to try generating a larger sample set, the starting points of samples were shifted by a number offset from the length of the samples. The sample shift was performed three to four times to effectively triple to quadruple the size of the sample sets.



```
GrandPiano vs Bass Guitar Shifted
        Number of mislabeled points out of a total 2690 points : 4
        Ratio correct:0.998513
        TrainingSetSize : 6270
GrandPiano vs Rhodes vs Clav Shifted
        Number of mislabeled points out of a total 4035 points : 228
        Ratio correct:0.943494
        TrainingSetSize : 9405
GrandPiano vs Rhodes vs Clav vs Bass Shifted
        Number of mislabeled points out of a total 5380 points : 235
        Ratio correct:0.956320
        TrainingSetSize : 12540
GrandPiano vs Clarinet Shifted
        Number of mislabeled points out of a total 1250 points : 50
        Ratio correct:0.960000
        TrainingSetSize : 2910
```

Figure 4.2: Accuracy results of classifiers built to classify different sets of instruments (shifted by time offsets for generating larger training and testing sets.

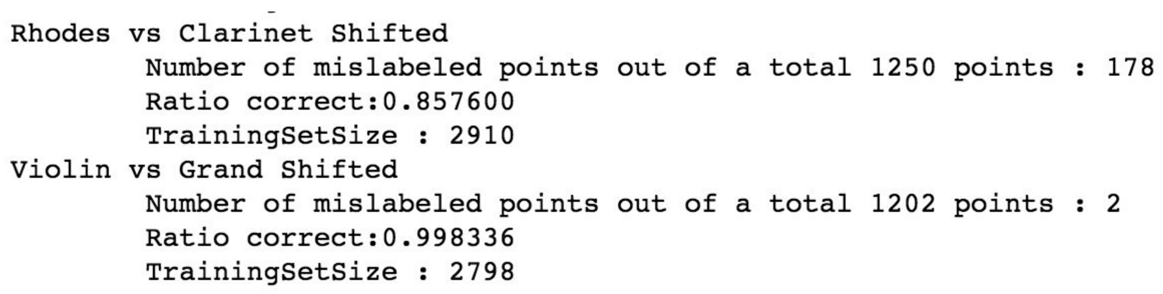
An interesting discrepancy can be seen in the "Clarinet vs Violin Classifiers" in both the unshifted *[Figure 4.3, top]* and the shifted classifiers *[Figure 4.3, bottom]*. Although the results of the other classifiers were really high, the classification between Clarinet and Violin is very inaccurate.



```
Clarinet vs Violin Less Samples
        Number of mislabeled points out of a total 55 points : 20
        Ratio correct:0.636364
        TrainingSetSize : 121
Clarinet vs Violin More Samples
        Number of mislabeled points out of a total 927 points : 237
        Ratio correct:0.744337
        TrainingSetSize : 2153
```

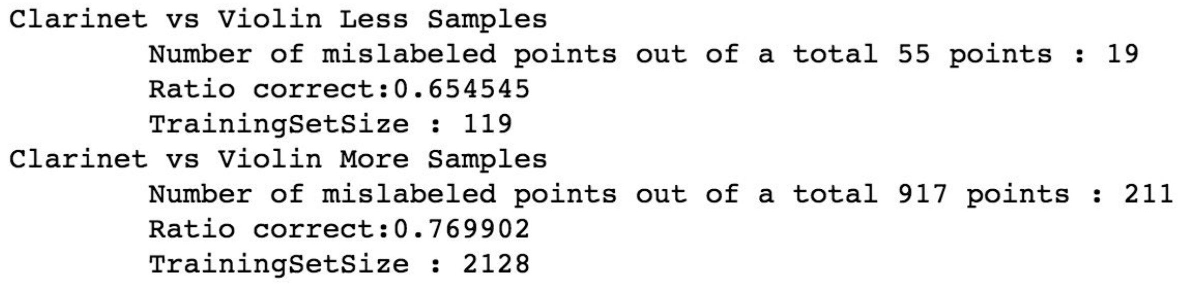
Figure 4.3: Shifted and unshifted sample sets used to build classifiers for Clarinet vs Violin. Note the relatively low accuracy.

Additional classifiers were built for testing these problem instruments against other instruments to see the prediction accuracies *[Figure 4.4]*.



```
Rhodes vs Clarinet Shifted
        Number of mislabeled points out of a total 1250 points : 178
        Ratio correct:0.857600
        TrainingSetSize : 2910
Violin vs Grand Shifted
        Number of mislabeled points out of a total 1202 points : 2
        Ratio correct:0.998336
        TrainingSetSize : 2798
```

Figure 4.4: Clarinet and Violin classification against other instruments. Note the relatively high accuracies.



```
Clarinet vs Violin Less Samples
        Number of mislabeled points out of a total 55 points : 19
        Ratio correct:0.654545
        TrainingSetSize : 119
Clarinet vs Violin More Samples
        Number of mislabeled points out of a total 917 points : 211
        Ratio correct:0.769902
        TrainingSetSize : 2128
```

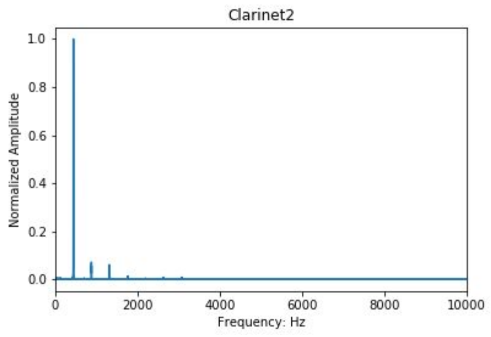
Figure 4.5: Clarinet and violin classification using the sum binning heuristic in an attempt to capture more high frequency information.

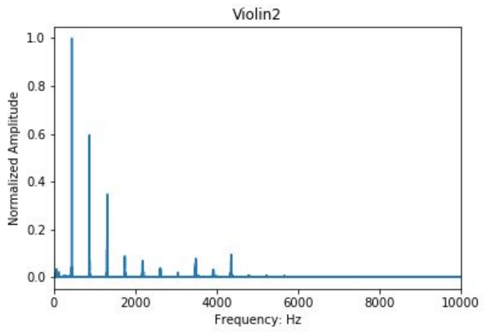
5. Analysis of Classifier Efficacy

In *[Figure 4.1]*, the classification accuracies demonstrates that the naive bayesian gaussian classifier can pretty accurately classify the different sets of instruments. Areas where the classifiers were expected to struggle, the classifiers struggled. For example, the four keyboard instruments that were selected because of their similarity in instrument category are shown to be harder for the classifier to classify–an accuracy of only 88%. The two least similar have the highest accuracy where 100% of the test set was classified correctly. In addition, tests including the clarinet which were the most accurate with the exception of the grand piano vs bass guitar classifier, due probably to the fact that the clarinet does not have a strong partial frequency presence at the frequency of the note being played, instead relying on residue pitches to generate the pitch being played. The lower accuracy is due, at least in part, to the lower consistency of the clarinet sample set.

In *[Figure 4.2]*, the classification accuracies all saw a dramatic increase. Using the shifting method described, the sample sizes were increased by three to four times. The perfect accuracy in the smaller sample size found in the first classifier between the grand piano and the bass guitar saw a small drop in accuracy as a result of the larger test set. More samples being tested results in more chances for the classifier to fail. All three of the other classifiers saw increases in at least 4%, where all three classifiers saw increases to above 94%. The five instrument classifier was not tested for shifted data sets due to the excessive size of the new data set causing issues in the amount of time to generate the data. The clarinet and the piano classifier accuracy actually increased by a much smaller margin than the other classifiers. The relatively small increase in accuracy shows that the issue with the training data is less in the size of the data set, but more in the inconsistency of the clarinet sample set. The approach is somewhat hands off, though, so it becomes difficult to tell exactly why some classes are better classified while others are not without extensive research into the data sets.

[Figure 4.2] shows how the classifier built from the two data sets with more inconsistent recordings were really was really weak at classifying between the two instruments (the violin and the clarinet). In *[Figure 4.3]*, the data sets were artificially increased using the shifting method. Increasing the data set saw an increase by more than ten percent. The increase shows that the fact that the training sets were smaller greatly affected the accuracy of these classifiers. Additional classifiers were constructed of these instruments against more consistent sets of data. The accuracy of these classifiers is much higher at 85% and 99%. The discrepancy shows both that having a larger data set provides better results. In addition, the better consistency of the other data sets is critical to the accuracy of the classifier. Although, it is important to take into account that the test sets being used to test these classifiers are extractions from the same samples as are being used to train the classifiers. The locality of the training set and the test set could be inflating the accuracy results of the classifiers.





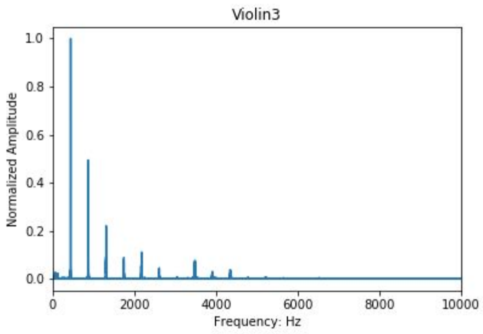


Figure 5.1: Three clarinet sample fourier transforms. Three violin sample fourier transforms.

When listening the the actual samples of the clarinet, the sounds were relatively inconsistent, *[Figure 5.1]* shows that overall, at least these three samples were pretty consistent in terms of their fourier transforms. In addition, the violin's fourier transform as seen in *[Figure 2.c.3]* is visibly different from the fourier transform of the clarinet. A potential cause for the lower accuracy of the classifier between these two instruments may be caused by the binning method. Both of these instruments have a weak presence of the fundamental frequency of the note which can be seen in *[Figure 2.c.3]*. The peaks at just above 400 Hz have the strongest presence, though neither of the instruments is playing 400Hz, but rather playing middle C which is closer to 200 Hz. Both instruments depend heavily on the residue pitches. Higher frequencies are more affected by the binning problems described in Section 3 because higher frequency harmonics will tend to be packed with the same or similar density to those at lower frequencies. The size of the bins increases at higher frequencies, though, meaning more information is lost. Instruments that depend on clusters of higher frequencies for their characteristic tones will in turn, be harder to classify using feature vectors built on this binning heuristic.

[Figure 4.5] Shows an attempt at a method to resolve the issue of the imperfect binning heuristic by instead using a peak sum in the range of the bin as opposed to a max sum as seen in Section 3. The results was a barely marginal increase in accuracy. The marginal increase demonstrates that either using a sum binning heuristic is either not effective in capturing more high frequency information, or that most of the issues are just in the consistency and size of the data set.

6. Conclusion

Overall, the methods used in these classification tests were able to show that classification using a gaussian model and naive bayesian probabilities is effective in a classifying instruments based solely on the sustained portions of the instrument notes, excluding the temporal features. Using the binning heuristic of binning octaves provides enough granularity to classify instruments with more critical high frequency harmonic content in the case of the instruments that were used in this project. If more instruments are to be classified, using a different binning heuristic may be potentially more useful, as even marginal improvements may be more important in sets of more instruments.

References

[2.c.r1] Heller, “The Trumpet”, Why You Hear What You Hear, Section 16.3, 2013

[3.b.r1] Spectrasonics Keyscape, https://www.spectrasonics.net/products/keyscape/index.php
[3.b.r2] Kontakt 5,
https://www.native-instruments.com/en/products/komplete/samplers/kontakt-5/
[3.b.r3] Ableton,
https://www.ableton.com/
[3.b.r4] Scipy, https://www.scipy.org/
[3.b.r5] Wav, https://www.loc.gov/preservation/digital/formats/fdd/fdd000001.shtml

[3.c.r1] Python 3, https://docs.python.org/3/
[3.c.r2] Jupyter Notebook, http://jupyter.org/
[3.c.r3] Heller, “Dissonance and Temperament”, Why You Hear What You Hear, Chapter 26, 2013

[github.r]
https://github.com/gabriellukechen/instrument_classification_physics_project/blob/master/cla
r_vs_violin.csv