Contents

Domain Background

Student briefly details background information of the domain from which the project is proposed. Historical information relevant to the project should be included. It should be clear how or why a problem in the domain can or should be solved. Related academic research should be appropriately cited. A discussion of the student's personal motivation for investigating a particular problem in the domain is encouraged but not required.

Music is one of the oldest ways of expression known to man. We create musical compositions by arranging notes from various tools such as vocals, physical instruments, and synthetic tools. We can identify most instruments used in a musical piece by just listening to it. We can also create music of our own by listening to compositions by other artists; we can copy their cadence or mirror their style.

The idea of using machine learning tools to classify and generate music is not entirely new in research [1][2][3]. Exploring this domain can prove useful for purposes of classification such as a streaming service looking to label sound pieces. For business models built on recommendation system, knowing the genre of a piece of music is essential. Machine learning in music can also be useful to test the limits of artificial intelligence in the auditory domain.

I've always been fascinated by the musical process, such as the broad themes that create genres and the subtle variations and progressions that make a song stand out. Through this project I hope to present pieces of music theory and provide a digestible intersection with machine learning.

Problem Statement

Student clearly describes the problem that is to be solved. The problem is well defined and has at least one relevant potential solution. Additionally, the problem is quantifiable, measurable, and replicable.

The objective of this project is to classify the instrument used in a sound recording. Other features to classify can also be the note qualities, and whether a sound is acoustic, synthetic, or electronic.

The project will also explore music generation, by comparing a piece generated by long short term memory (LSTM) versus the actual notes of a composition. This is akin to making predictions on text or stock performance, given that we already know how the sentence ends and how the stock fluctuates.

Datasets and Inputs

The dataset(s) and/or input(s) to be used in the project are thoroughly described. Information such as how the dataset or input is (was) obtained, and the characteristics of the dataset or input, should be included. It should be clear how the dataset(s) or input(s) will be used in the project and whether their use is appropriate given the context of the problem.

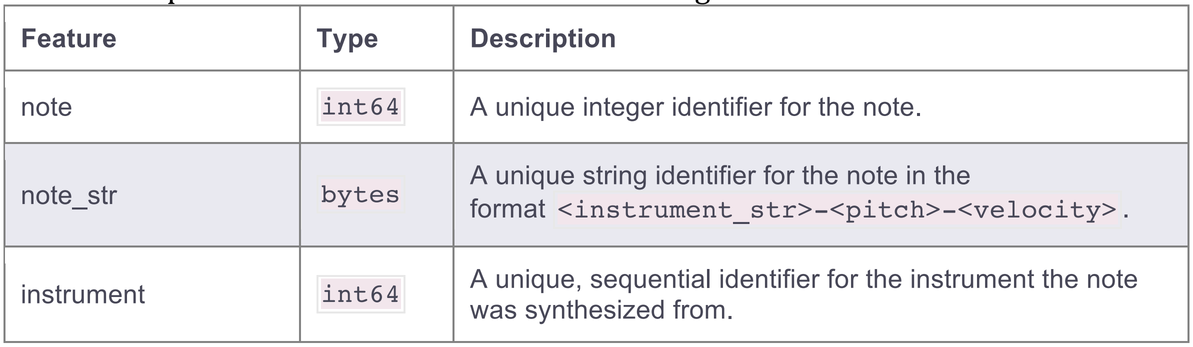
NSynth

NSynth is a large scale and high quality dataset of annotated musical notes. The description below is taken from the host website:

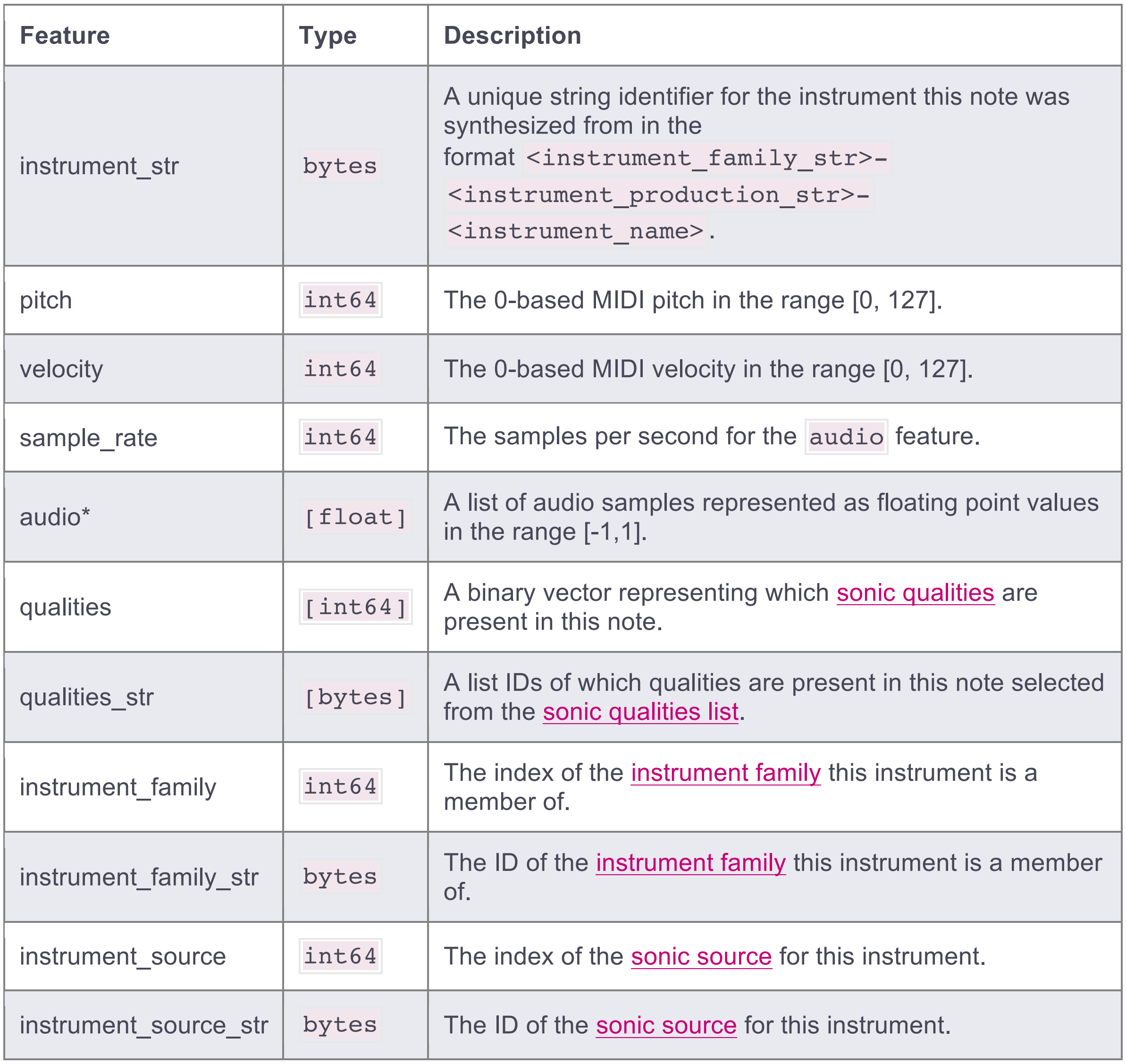
"NSynth is an audio dataset containing 305,979 musical notes, each with a unique pitch, timbre, and envelope. For 1,006 instruments from commercial sample libraries, we generated four second, monophonic 16kHz audio snippets, referred to as notes, by ranging over every pitch of a standard MIDI piano (21-108) as well as five different velocities (25, 50, 75, 100, 127). The note was held for the first three seconds and allowed to decay for the final second.

Some instruments are not capable of producing all 88 pitches in this range, resulting in an average of 65.4 pitches per instrument. Furthermore, the commercial sample packs occasionally contain duplicate sounds across multiple velocities, leaving an average of 4.75 unique velocities per pitch."

Each example in the dataset contains the following features:



Feature	Type	Description
note	`int64`	A unique integer identifier for the note.
note_str	`bytes`	A unique string identifier for the note in the format `<instrument_str>-<pitch>-<velocity>`.
instrument	`int64`	A unique, sequential identifier for the instrument the note was synthesized from.



Feature	Type	Description
instrument_str	`bytes`	A unique string identifier for the instrument this note was synthesized from in the format `<instrument_family_str>-<instrument_production_str>-<instrument_name>`.
pitch	`int64`	The 0-based MIDI pitch in the range [0, 127].
velocity	`int64`	The 0-based MIDI velocity in the range [0, 127].
sample_rate	`int64`	The samples per second for the `audio` feature.
audio*	`[float]`	A list of audio samples represented as floating point values in the range [-1,1].
qualities	`[int64]`	A binary vector representing which sonic qualities are present in this note.
qualities_str	`[bytes]`	A list IDs of which qualities are present in this note selected from the sonic qualities list.
instrument_family	`int64`	The index of the instrument family this instrument is a member of.
instrument_family_str	`bytes`	The ID of the instrument family this instrument is a member of.
instrument_source	`int64`	The index of the sonic source for this instrument.
instrument_source_str	`bytes`	The ID of the sonic source for this instrument.

Google Inc. makes the dataset readily available under a Creative Commons Attribution 4.0 International (CC BY 4.0) license. The creators encourage the machine learning to use NSynth as a benchmark and entry point into audio machine learning.
NSynth is therefore available for public use and suitable for the task of machine learning on audio files. The recordings are of high quality and the features are quantifiable.

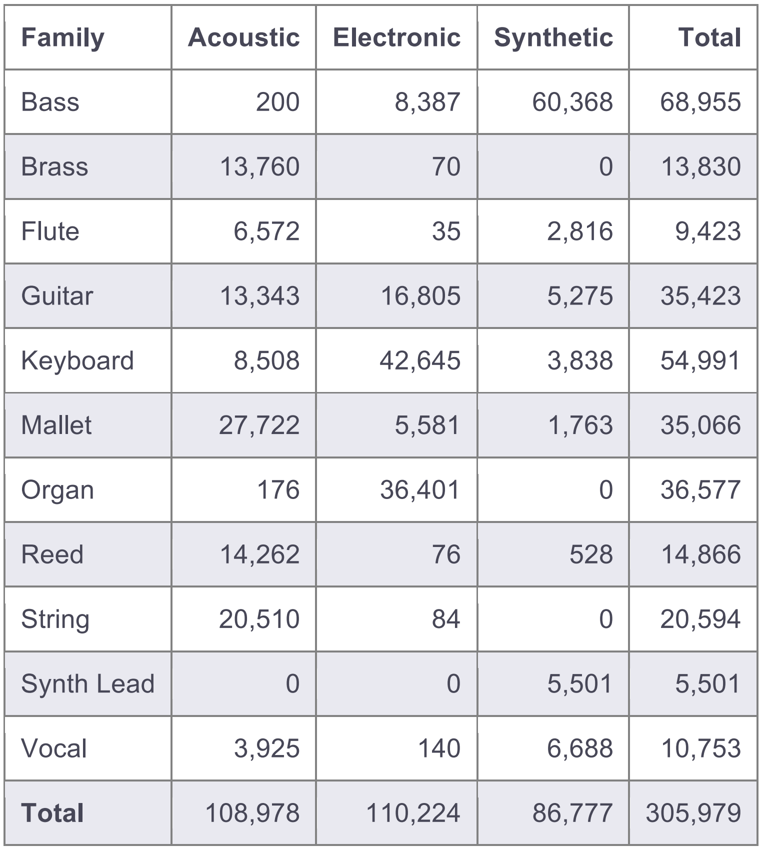
http://piano-midi.de

The second dataset is music files of classical music. The pieces are developed at a digital piano by means of a sequencer on MIDI and then converted to audio format. It is concurrent with the previous dataset since it also uses MIDI format audio. The dataset is licensed by a cc-by-sa Germany License: we can use and adapt the files as long as we attribute to the copyright holder. In addition, the composers no longer exercise copyright due to the fact that they have been all dead from more than 70

years. The owner of the page allows the use of the data as long as attribution is given to the copyright holder.
This dataset is appropriate for music generation because we are using a single instrument: the piano. In addition the entire musical composition is available, meaning we are dealing with a closed environment.

Solution Statement

Student clearly describes a solution to the problem. The solution is applicable to the project domain and appropriate for the dataset(s) or input(s) given. Additionally, the solution is quantifiable, measurable, and replicable.
The classification of the NSynth audio files can be easily evaluated since the data is already provided. For example the creators of NSynth in the table below give the instrument family and type.



Family	Acoustic	Electronic	Synthetic	Total
Bass	200	8,387	60,368	68,955
Brass	13,760	70	0	13,830
Flute	6,572	35	2,816	9,423
Guitar	13,343	16,805	5,275	35,423
Keyboard	8,508	42,645	3,838	54,991
Mallet	27,722	5,581	1,763	35,066
Organ	176	36,401	0	36,577
Reed	14,262	76	528	14,866
String	20,510	84	0	20,594
Synth Lead	0	0	5,501	5,501
Vocal	3,925	140	6,688	10,753
Total	108,978	110,224	86,777	305,979

For the classical music pieces, a solution is defined as a new composition that is similar, yet not identical, to how the song actually progresses. For example we know the entire song and our model has seen 90% of it, a viable solution is where those generated notes match the unseen 10%. We thus have a quantifiable measurement of accuracy for this task.

For the NSynth dataset, supervised machine learning methods are the most appropriate since we already know the data labels. An algorithm that can prove useful is Naïve Bayes since some features are mutually exclusive. This is particularly true for note qualities, where "dark" and "light" notes are mutually exclusive.
For generation of music, it can be compared to text generation since musical notes are quantifiable in the same measure that words are. A recurrent neural net to generate text has been done in previous experiments and can be replicated for this task. [4]

Benchmark Model

A benchmark model is provided that relates to the domain, problem statement, and intended solution. Ideally, the student's benchmark model provides context for existing methods or known information in the domain and problem given, which can then be objectively compared to the student's solution. The benchmark model is clearly defined and measurable.
Classification of datasets is done supervised methods such as linear models, Naïve Bayes, Decision Trees, Random Forests, and support vector machines. Unsupervised machine learning models such as k-means and DBSCAN can be used for clustering and identification. However given that we already know the labels, unsupervised learning may be better suited for exploratory data analysis. Thus the benchmark for classification is correctly identifying as many instruments as possible.

In addition, neural networks can be leveraged for both the tasks of classification and generation of music. Methods for creating LSTM models are already established in some examples. In addition, generative adversarial neural networks can be used in the task of music generation if we consider the music as a midi picture. Libraries dealing with MIDI files using python are present on public repositories on github.com and thus accessible for use. Given the above, the benchmark for music generation is generating notes that closely resemble the sequence hidden from the algorithm.

Evaluation Metrics

Student proposes at least one evaluation metric that can be used to quantify the performance of both the benchmark model and the solution model presented. The evaluation metric(s) proposed are appropriate given the context of the data, the problem statement, and the intended solution.
Music classification evaluation is done using accuracy and more generally a confusion matrix. This will provide quantifiable measurement of how true our prediction is and provide insight into common misclassifications. In addition the model will be evaluated using an f1 score because the dataset is not balanced, for example we have about 9,400 flute samples and 35,000 guitar samples.
For music generation, evaluation is based on the mean squared distance between the actual note and the generated one, this will be used to calculate the mean

squared error. Each key on a piano corresponds to a given frequency, thus the information for this task is quantifiable.

Project Design

Student summarizes a theoretical workflow for approaching a solution given the problem. Discussion is made as to what strategies may be employed, what analysis of the data might be required, or which algorithms will be considered. The workflow and discussion provided align with the qualities of the project. Small visualizations, pseudocode, or diagrams are encouraged but not required.

The steps below are inspired by this post on medium.com and seem suitable for the scope of this project.

1) Problem formulation, goal statement, and general background information.
2) Present all libraries and corresponding versions at beginning of project.
3) Present how data is obtained (download, scraping).
4) Exploratory data analysis through visualization plots such as Pearson correlation coefficients. Histograms will be used to describe spread of features, this will be helpful to see if some data follows a distribution (normal, chi squared, etc...).
5) Data manipulation/cleaning into desired formats and types for analysis, we can might use feature scaling to squeeze data into reasonable values. MIDI libraries to be converted into formats that work with tensorflow framekworks. We can view musical compositions as an n-gram bag of words, where each note corresponds to a value inside a matrix and the note order matters.
6) Classification and evaluation using supervised learning: Naïve Bayes, support vector machines.
7) Classification and evaluation using neural networks libraries such as tensorflow.
8) Summarize best model for classification, compare accuracies and f1 scores.
9) Generation of piano music using recurrent neural networks
10) Generation of piano music using GANs
11) Determine mean square error in music generation.
12) Summary and reflections

Sources

The database sources used for this project are so far:

- Jesse Engel, Cinjon Resnick, Adam Roberts, Sander Dieleman, Douglas Eck, Karen Simonyan, and Mohammad Norouzi. "Neural Audio Synthesis of Musical Notes with WaveNet Autoencoders." 2017.
- Bernd Krueger, http://www.piano-midi.de/
- https://arxiv.org/abs/1406.2661
- https://github.com/vishnubob/python-midi

The academic sources:

- [1] Wang, 2013
  http://homepages.cae.wisc.edu/~ece539/fall13/project/WangShu_rpt.pdf
- [2] Huang, Wu, 2016
  https://cs224d.stanford.edu/reports/allenh.pdf
- [3] D Eck, T Bertin-Mahieux, P Lamere - ISMIR, 2007
  http://tbertinmahieux.com/Papers/ismir07_submission.pdf
- [4] Shang, Zhendong, Li, 2015
  arXiv:1503.02364