Classification is a **data-driven** approach for applying **labels** to **data**. Once a classifier has been **trained** on a training set that includes the true labels, it will **predict** labels for new data it hasn’t seen before.

**Example computer music classification tasks:**
- Which gesture did the performer just make with the accelerometer?
- Which instruments are playing right now?
- Who is singing? What language are they singing? What pitches are they singing? Are they singing vowels or consonants?
- Is this chord major or minor?
- Is this sound dissonant or harmonic?
- Is this dancer moving quickly or slowly?

**Terminology:**

**Data-driven:** In a data-driven approach, you supply the computer with examples of what you mean, instead of sitting down and trying to code a labeling algorithm from scratch. You can often improve accuracy by adding more data, supplying better features, or changing the learning algorithm and its parameters.

**Labels:** The class labels (or “classes”) for an instrument identification task might be “flute,” “trumpet,” and “violin.” Or, for a “who’s talking right now?” task, the labels might be “Rebecca,” “Dan,” and “nobody.”

**Data:** A real-world phenomenon (a sound or a segment of a sound, or the output of a sensor) that is represented as a list of features (or “feature vector,”), and possibly a label for each list of features.

**Features:** Each data point (or “instance”) is associated with a list of features, which in general are numbers (or true/false values treated as numbers) and result from some measurement of the world, perhaps at a point in time. These features should be relevant to the labeling task in some way. For example, an instrument classification task would use features that convey some aspects of the timbre computed for some short time period (e.g. an FFT window), and a gesture classification task might use the sensor output values recorded over some number of seconds.

**Training stage:** In the training stage, the classifier uses the feature vectors and labels (the “ground truth”) to learn to generalize about the relationship between them. Each classifier makes different assumptions about how feature values relate to the labels and uses this information differently during training.

**Classification stage:** After the classifier has been trained, you can **run** it. When you give it a new data point (a feature vector, using the same feature vectors you used in training), it will output its predicted label.
TRAIN: Give labeled features to classifier

<table>
<thead>
<tr>
<th>Reads comics</th>
<th>Loves ruby</th>
<th>Height (cm)</th>
<th>Nerd?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>N</td>
<td>60</td>
<td>No!</td>
</tr>
<tr>
<td>N</td>
<td>Y</td>
<td>71</td>
<td>Yes!</td>
</tr>
<tr>
<td>Y</td>
<td>Y</td>
<td>55</td>
<td>Yes!</td>
</tr>
<tr>
<td>N</td>
<td>Y</td>
<td>80</td>
<td>No!</td>
</tr>
</tbody>
</table>

TEST: Output labels for new feature vectors

<table>
<thead>
<tr>
<th>Reads comics</th>
<th>Loves ruby</th>
<th>Height (cm)</th>
<th>Nerd?</th>
</tr>
</thead>
<tbody>
<tr>
<td>Y</td>
<td>N</td>
<td>81</td>
<td>???</td>
</tr>
</tbody>
</table>

OK, so what do I need to do?
1. Choose your classification problem, e.g., singer identification.
2. Choose some features to extract from the available data sources (adc, sensors, ???), which have something to do with the labeling task (can be implemented using ChucK UAnae and smirK)
3. Construct a training set by extracting features and applying expert labels, and train the classifier (can use smirK)
4. Pass newly extracted features to the trained classifier, possibly in real-time (can use smirK)
5. Do something with the predictions you get back! (Evaluate the classifier, make some noise, etc.)

Resources
ChucK UAnae documentation (how do I extract features in ChucK?)

SMIRK (Small music information retrieval toolkit)
http://smirk.cs.princeton.edu/
Website has all code, examples, and papers
Requires ChucK version 1.2.1.2 or later.