Graphical Features

In addition to text and natural language processing, bag of words has successfully been applied to images by categorizing a collection of regions and describing only their appearance, but ignoring any spatial structure. However this is not the typical approach which is used, and requires you come up with methods of categorizing image regions. More frequently used methods include:

1. Split the image into a grid of smaller areas, and attempt feature extraction at each locality. Then return a combined array of all discovered features
2. Use variable-length gradients and other transformations as the features, such as regions of high / low luminosity, histogram counts for horizontal and vertical black pixels, stroke and edge detection, etc.
3. Resize the picture to a fixed size, convert it to grayscale, then encode every pixel as an element in a unidimensional feature array

Let's explore how you might go about using the third method with code:

- 3. Exploring Data
- 4. Transforming Data
- 5. Data Modeling

```
# Uses the Image module (PIL)
from scipy import misc

# Load the image up
img = misc.imread('image.png')

# Is the image too big? Shrink it by an order of magnitude
img = img[::2, ::2]

# Scale colors from (0-255) to (0-1), then reshape to 1D Array
X = (img / 255.0).reshape(-1, 3)

# To-Do: Machine Learning with X!
#
```

Audio Features

Audio can be encoded with similar methods as graphical features, with the caveat that your 'audio-image' is already a one-dimensional waveform data type instead of a two-dimensional array of pixels. Rather than looking for graphical gradients, you would look for auditory ones, such as the length of sounds, power and noise ratios, and histogram counts after applying filters.

```
import scipy.io.wavfile as wavfile

sample_rate, audio_data = wavfile.read('sound.wav')
print audio_data

# To-Do: Machine Learning with audio_data!
#
```