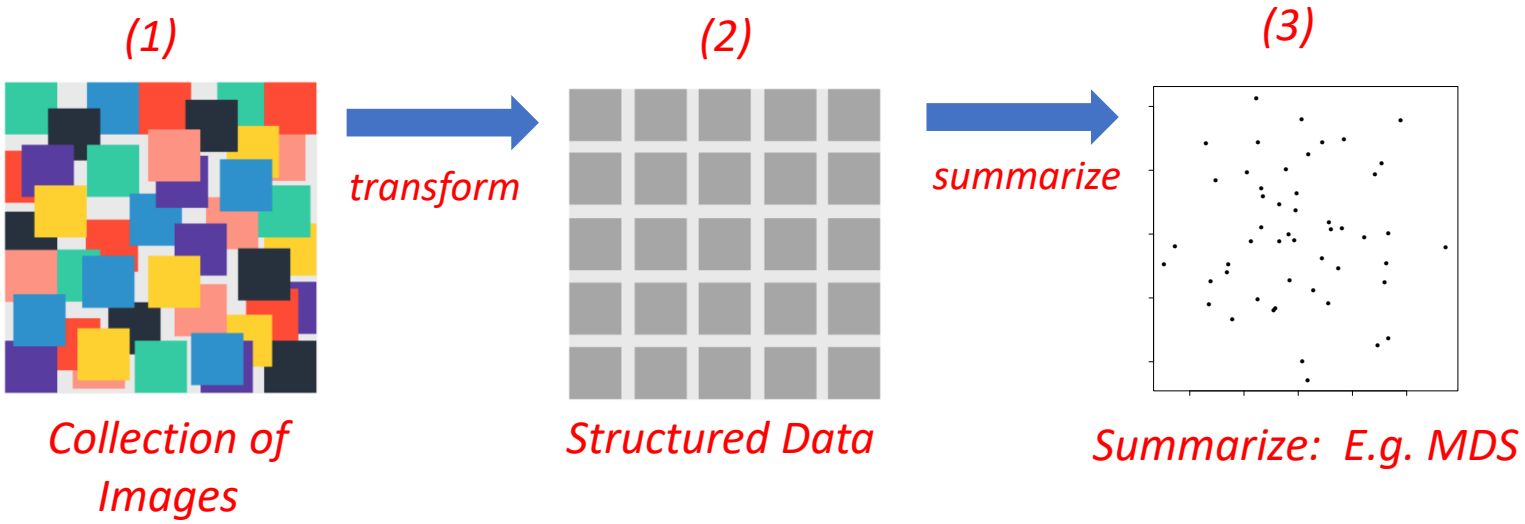
Review: Image Analytics

- Images are unstructured observations in the collection
- Rely on critical thinking to transform and summarize



Transforming Images to Structured Datasets

- Feature approach:
  - Consider a list of all features that *could* be in images
    - E.g., Tree, Beach, Animal, Human, House, Flower, etc.
  - Humans go through images, 1 by 1, to look for features
  - Image analyses are based on feature lists
  - Analogy:
    - Subjective lexicon is to Text Analytics *as* Feature list is to Image Analytics
- Pixel approach:
  - Automate processing of images to structured data via information of each pixel
  - TODAY

Transform to structured data via pixel approach

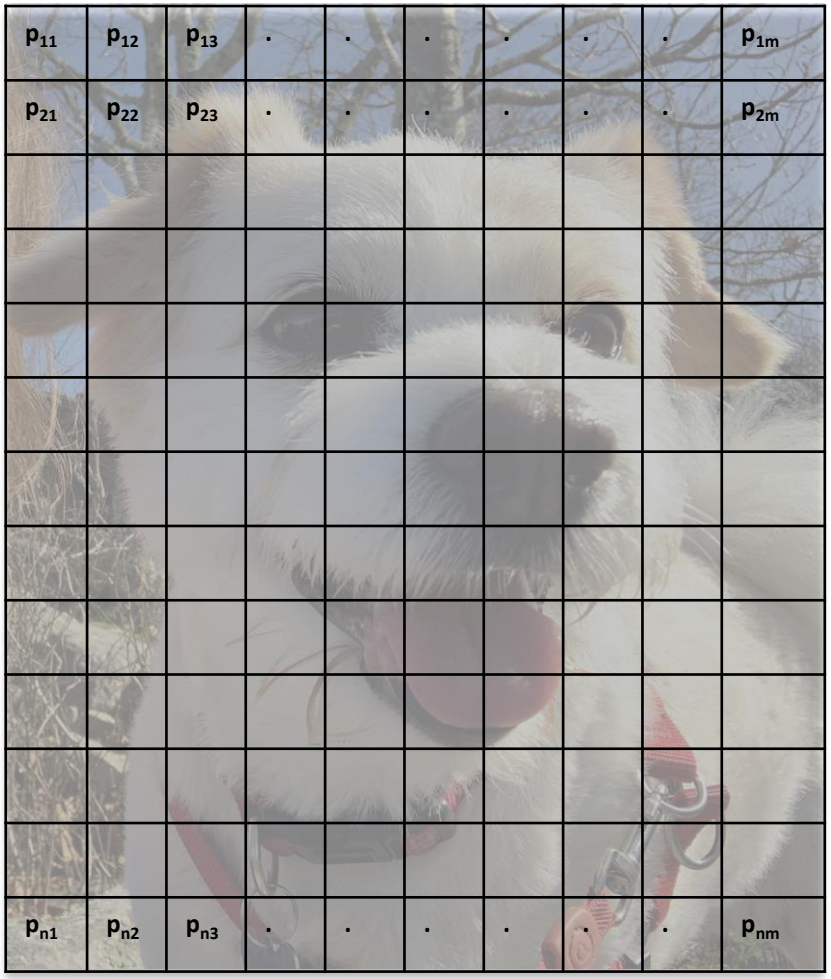
- 1 image is n x m pixels
- Define the amount of a *specific* color in each pixel:
- Pixel Matrix:

$p_{11}, p_{12}, \ldots, p_{1m}$

$p_{21}, p_{22}, \ldots, p_{2m}$

..

$p_{n1}, p_{n2}, \ldots, p_{nm}$



- Vectorize: $[p_{11}, p_{12}, \ldots p_{1m}, p_{21}, p_{22}, \ldots, p_{2m} \ldots, p_{n1}, p_{n2}, \ldots . p_{nm}]$

Transform to structured data via pixel approach (cont)

- Stack the vectorized matrices together to make one dataset.
- Each row represents one image (i.e., observation); let N represent the number of images being analyzed.
- Each column represents one color for one pixel; i.e., one variable = one color for one pixel
- If considering ONE color, the dimension of the data is N x (nm)
  - N images
  - n rows of pixels in an image
  - m columns of pixels in an image

Image	**P_{11}**	**P_{12}**	**…**	**P_{1m}**	**P_{21}**	**P_{22}**	**…**	**P_{2m}**	**…**	**…**	**P_{n1}**	**P_{n2}**	**…**	**P_{nm}**
1														
2														
3														
…														
N														

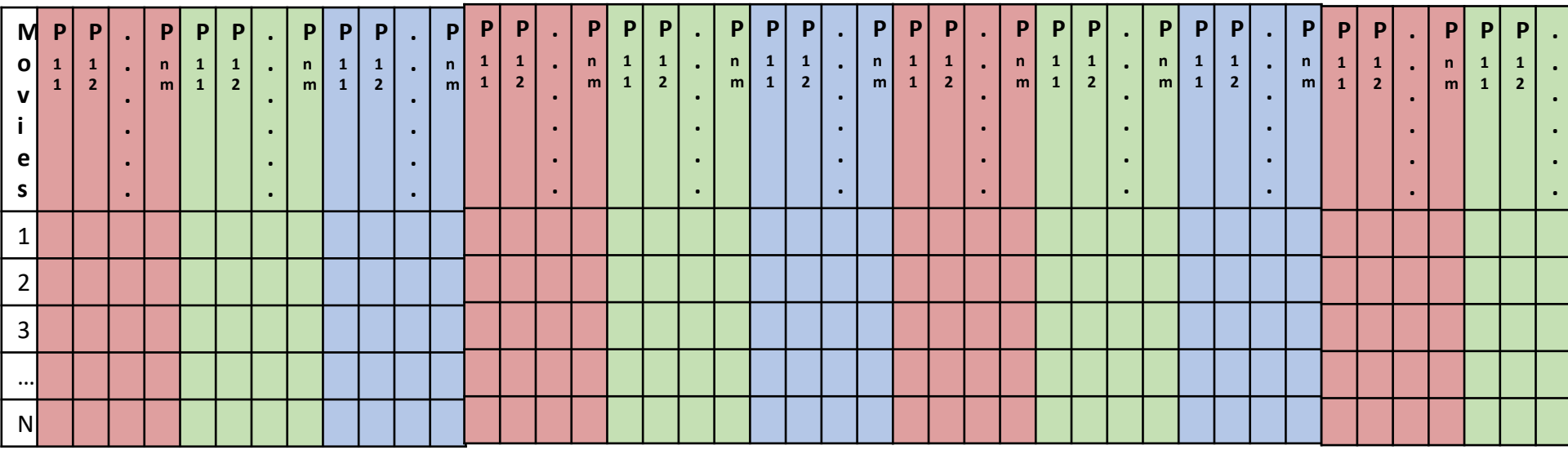
Transform via pixel approach (cont)

- If considering, say, three colors (e.g., RGB), the dimension of the data is N x (3nm)
  - N images
  - n rows of pixels in an image
  - m columns of pixels in an image Dimension
  - 3 colors per pixel, thus pixel columns repeated 3 times

Image	P_{11}	P_{12}	… …	… …	P_{nm}	P_{11}	P_{12}	… …	… …	P_{nm}	P_{11}	P_{12}	… …	… …	P_{nm}
1															
2															
3															
…															
N															

Transform via pixel approach (cont)

- If considering a movie that is compiled from, say, 10 (static) snapshots the dimension data dimension is N x (30nm)
  - N *movies* with 10 images each
  - n rows of pixels in an image
  - m columns of pixels in an image Dimension
  - 3 colors per pixel, thus pixel columns repeated 3 times
  - 10



Anatomy of Images (pixel approach)

- Pixels: x-axis (m columns) and y-axis (n rows)
- Number of pixels = nm
- One image has nm pixels
- One movie has nm pixels per depth
  - Static images = 1 depth
  - Movie > 1 deep
- Define amount or “intensity” of color(s) in each pixel
- Color Spectrum (selected by you or image-creator):
  - RGB: Red, Blue, Green (primary colors for light)
  - CMYK: Cyan, Magenta, Yellow, Black (primary colors for pigment)
  - Gray Scale
  - “Amount of each color”

Defining color per pixel

- One color (e.g., aqua, tangerine, violet) is defined by combinations of "primary" colors in a color spectrum
- E.g., If the color spectrum is RGB, we define  with specifying amounts red, green and blue.
- How do we specify amounts? A: Balance computer constraints with levels or "intensities" of color
- Primary colors are coded with 8 bits each, thus
  - 2^8=256 different "intensities" of primary color (PC) to specify
  - E.g., 0 = None of the primary color
  - E.g., 255 = Maximum amount/ intensity of the primary color
- Ex. 
  - Red: 196
  - Greed: 60
  - Blue: 90

Example Static image with RGB

- R, however, uses a scale from 0-1.
  - Maps [0,255] -> [0,1]
  - [0/255, 1/255, …., 255/255] -> [0, 0.00392, … 1]
- Ex. 
  - Red: 196/255 = 0.7686
  - Greed: 60/255 = 0.2353
  - Blue: 90 /255 = 0.3529
- With RGB, one image has three color-matrices:
  - Red-matrix, Green-matrix, Blue-matrix
- I.e., one image is described by an *array* of 3 matrices

Elements of images in R

- Library(imager)
- An image has the following dimensions

  # Columns x #rows x Depth x #Shades

- Static images have depth=1

  # Columns x # rows x 1 x #Shades

- RGB images

  # Columns x # rows x 1 x 3

- CMYK images

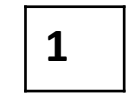
  # Columns x # rows x 1 x 4

Array, matrix, vector, scaler

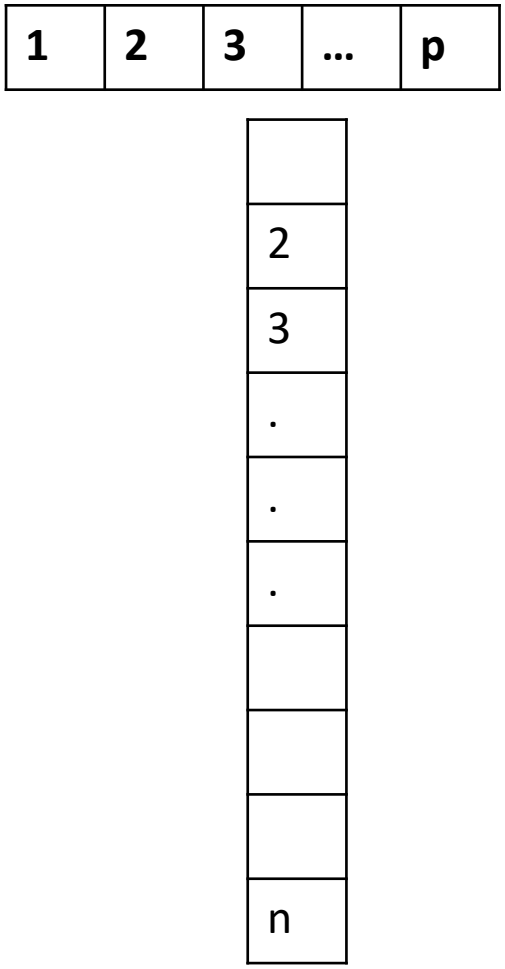
In words

- Scaler: 1 number
- Vector: a vertical or horizontal sequence of scalers
- Matrix: 1 or more vectors
- Array: 1 or matrices

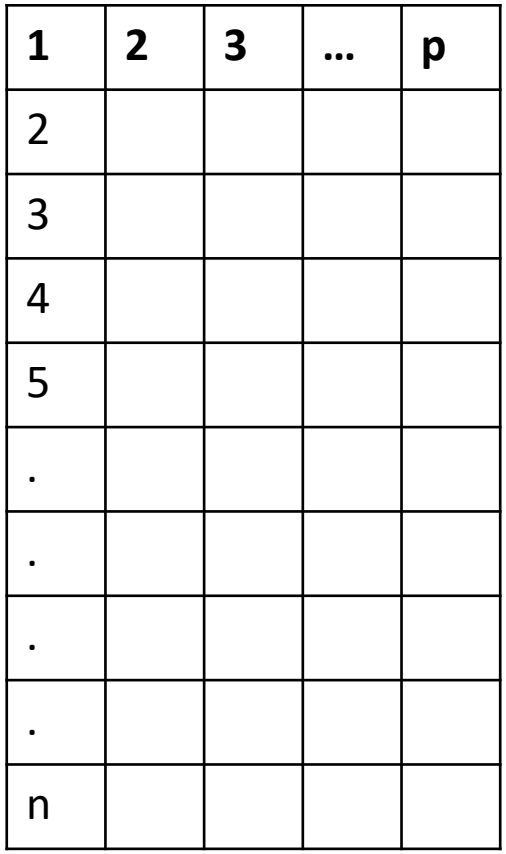
Scaler, vector, matrix, array,



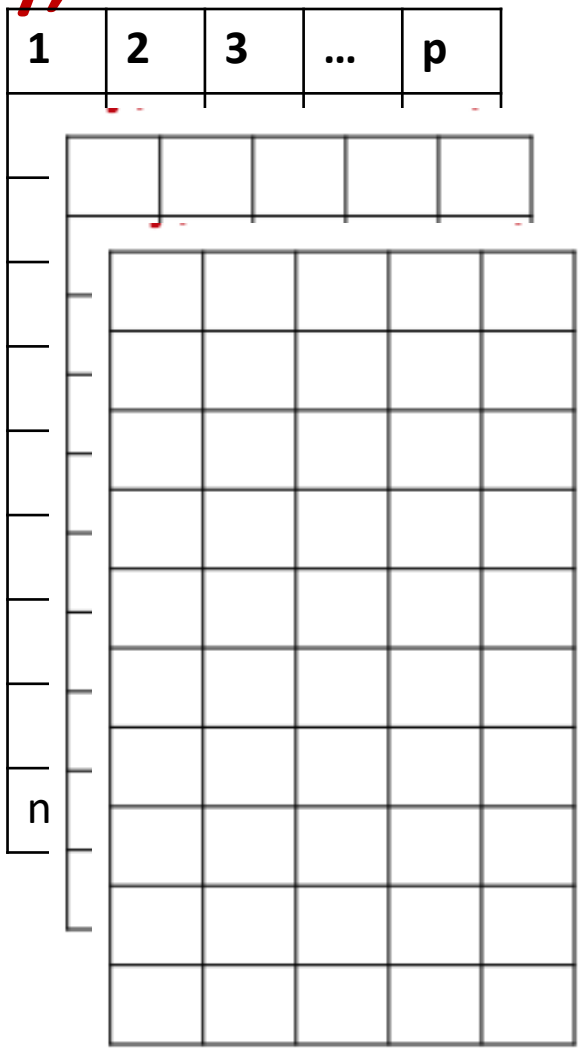
Scaler
1x1



matrix
1xp or n x1



matrix
n x p



array
n x p x **q x …**

Notice relationships

- Matrix is a special case of
  - an array: n x p x 1
- Vector is a special case of
  - a matrix: 1 x p or p x 1
  - an array: 1 x p x 1 or n x 1 x 1
- Scaler is a special case of
  - a vector: 1x1
  - a matrix: 1x1
  - an array: 1x1x1

Math Operations

- Simple operations: $\times$, $\div$, $+$, $-$
- Math
  - $+$, $-$: for objects of SAME dimension vectors, element by element in the object
  - Ex
  - $\times$,$\div$. : for *scalers* with vectors, matrices, arrays.
  - Ex
  - $\times$,$\div$. : for objects of SAME dimension vectors, element by element in the object
  - Technically, we call element-wise $\times$,$\div$ of matries, the *Hadamard* product or quotient

In R, we can create one from others

- vectorize a matrix
- Put a vector back into a matrix
- Under some conditions, we can merge levels in an array into one matrix
- Or we can spit up a matric to add dimensions in an array