In [1]:
```
%pylab inline
```

Populating the interactive namespace from numpy and matplotlib

In [14]:
```
#from IPython.display import Image
from PIL import Image
import os
import pandas as pd
import numpy as np
import pylab as pl
import matplotlib
```

In [20]:
```python
def load_images(img_dir, grayscale=False):
    """Loads images from within a specified directory.

    Args:
      img_dir (str): The directory from which to load (.jpg) images.
      grayscale (bool): Whether to convert the image into grayscale. Defaults to False.

    Returns:
      images: An array of image objects loaded from the specified directory

    """
    images = []

    for file in os.listdir(img_dir):
        if file.endswith(".jpg"):
            im = Image.open(os.path.join(img_dir, file))

            im = im.resize((100, 100))

            if grayscale:
                im = np.array(im, dtype=np.float64) / 255

                # Convert image to grayscale
                r, g, b = im[:,:,0], im[:,:,1], im[:,:,2]
                gray = 0.2989*r + 0.5870*g + 0.1140*b
                im = gray.reshape((1, -1))[0]

            images.append(im)

    return images

def plot_image_space(images, X, title="Projection of the Images into 2 Dimensions"):
    """Generates and shows a plot of images in a feature space.

    A figure with one plot is generated. The plot displays the location of each image in
    relation to the image's feature values in the input feature space (X).

    Args:
      images (Image): An image.
      images (SciPy array): An array of SSQs, one computed for each k.

    """
```

```
    # min-max normalization
    x_min, x_max = np.min(X, axis=0), np.max(X, axis=0)
    X = (X - x_min) / (x_max - x_min)

    # Create a figure
    pl.figure(figsize=(16, 5))
    ax = pl.subplot(111)
    #ax.axis('off')

    # Generate picture thumbnails in the plot
    if hasattr(matplotlib.offsetbox, 'AnnotationBbox'):
        # only print thumbnails with matplotlib > 1.0
        for i in range(len(images)):
            imagebox = matplotlib.offsetbox.OffsetImage(images[i], zoom=.65)
            ab = matplotlib.offsetbox.AnnotationBbox(imagebox, X[i][0:2])
            ax.add_artist(ab)

    # Add figure labels and ticks
    pl.title(title, fontsize=16)
    pl.xticks([]), pl.yticks([])

    # Add figure bounds
    pl.ylim((np.min(X, axis=0)[1])-0.25,(np.max(X, axis=0)[1])+0.25)
    pl.xlim((np.min(X, axis=0)[0])-0.1,(np.max(X, axis=0)[0])+0.1)
```

In [21]:

```
img_dir = os.path.join(os.getcwd(), "images") # directory path
images = load_images(img_dir) # load images in the specified directory
```

In [22]:

```
X = pd.DataFrame([im.histogram() for im in images])
```

In [29]:
```
plot_image_space(images, X, title="Projection of the Image Histograms into 2 Dimensions")
```

Projection of the Image Histograms into 2 Dimensions

Part1 [25pts]: The PCA projection of the image color histograms in 2 dimensions. Using the provided plot image space() function. This should be displayed as thumbnail images distributed within a 2-dimensional plot. You will need to use PCA, which is implemented in scikit-learn. See this link for documentation here (http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html (http://scikit-learn.org/stable/modules/generated/sklearn.decomposition.PCA.html)).

In [47]:
```
from sklearn.decomposition import PCA
# Generate PCA tranformation and plot results
pca = PCA(n_components=2)
#Fit the model with X and apply the dimensionality reduction on X.
X_fit = pca.fit_transform(X)
plot_image_space(images, X_fit, 'PCA Plot Results')
```

PCA Plot Results

Part2 [25pts]: Given this output, What does it mean for two images to be close together in this plot? What does it mean for two images to be far apart?

Answer: Upon intial inspection we can see that the pictures are grouped together by color (seasons), which means that they have similar pixel valuations. Pixel values that are very different are grouped farther apart and we can see this represented further by the seasons being grouped apart from one another. For example, winter is very far from the colorful orange tints of fall.

Part3 [50pts]: Once you completed the first two parts of the assignment, choose one of the following below:

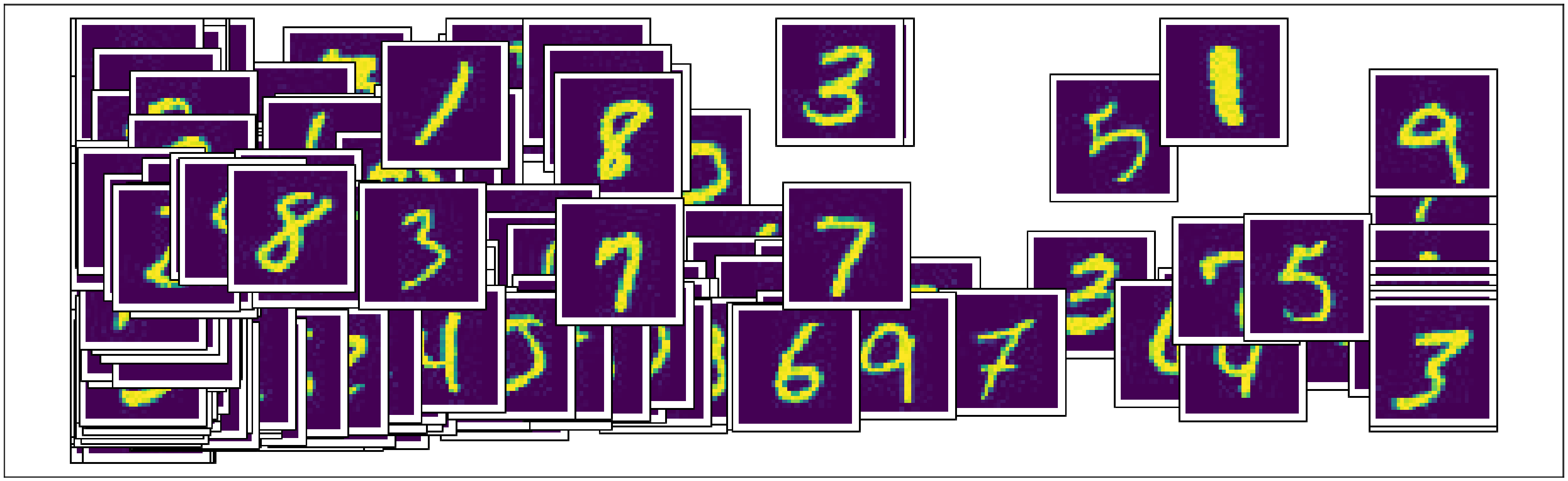
1. Repeat this process while using a different set of images curated by yourself.
2. Repeat this process using a different data reduction method and describe any similarities/differences between that experiment when compared to applying PCA.

In [43]:
```
# Try and see if PCA can group similar numbers together from the popular MNIST dataset:
# https://www.kaggle.com/scolianni/mnistasjpg/download
img_dir1 = os.path.join(os.getcwd(), "mnist") # directory path
mnist = load_images(img_dir1) # load images in the specified directory
```

In [40]:
```
y = pd.DataFrame([im.histogram() for im in mnist])
```

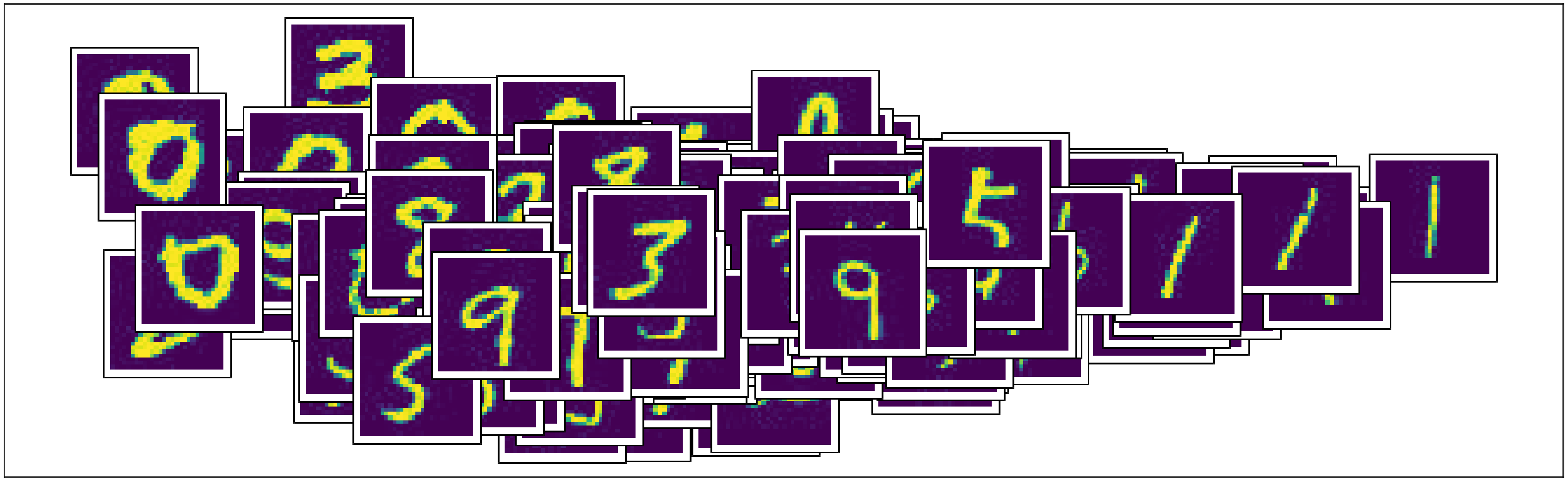
In [41]:
```
plot_image_space(faces, X, title="Projection of the MNIST image Histograms into 2 Dimensions")
```

Projection of the face image Histograms into 2 Dimensions



In [50]:
```
# Generate PCA tranformation and plot results
pca1 = decomposition.PCA(n_components=2)
y_fit = pca1.fit_transform(y)
plot_image_space(mnist, y_fit, 'PCA transformation for MNIST Dataset')
```

PCA transformation for MNIST Dataset



Part2 [25pts]: Given this output, What does it mean for two images to be close together in this plot? What does it mean for two images to be far apart?

Answer: I decided to try using PCA to see how it could apply to the popular MNIST dataset of hand written numbers. Compared to the initial 2-dimensional histogram image, the PCA transform does a slightly better job of grouping similar shaped numbers along with each other. Simple numbers like ones and zeros are clearly separated far from each other, however the more complex shaped numbers are not as differentiated.

In [ ]: