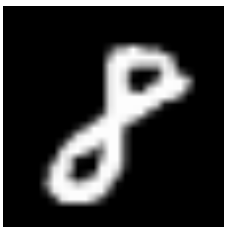
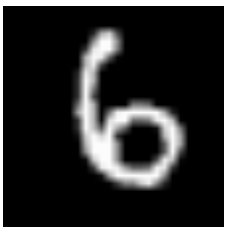
1. Introduction

Alice, Bob, and Daniel are friends learning machine learning together. After watching a very proud of having learned many useful tools, including linear and logistic regression, regularization, and kernel tricks. To see how these methods can be used to solve a real decide to get their hands dirty with the famous digit recognition problem using the MN Institute of Standards and Technology) database.

Hearing that you are an excellent student in the MITx machine learning class with solid material and great coding ability in Python, they decide to invite you to their team and h implementing these different algorithms.

The MNIST database contains binary images of handwritten digits commonly used to t systems. The digits were collected from among Census Bureau employees and high sc database contains 60,000 training digits and 10,000 testing digits, all of which have be centered in a fixed-size image of 28 × 28 pixels. Many methods have been tested with project, you will get a chance to experiment with the task of classifying these images in using some of the methods you have learned so far.


Setup:

As with the last project, please use Python's **NumPy** numerical library for handling array use **matplotlib** for producing figures and plots.

This project will be split in two parts. Project 2 (this project) consists in the first part an the second part.

1. *Note on software:* For all the projects, we will use python 3.6 augmented with the **Nu** the **matplotlib** plotting toolbox. In this project, we will also use the **scikit-learn** packa install in the same way you installed other packages, as described in project 0, e.g. b `scikit-learn` or `pip install sklearn`

2. Download mnist.tar.gz and untar it into a working directory. The archive contains the Dataset directory, along with the following python files:

   - `part1/linear_regression.py` where you will implement linear regression
   - `part1/svm.py` where you will implement support vector machine

between 0 and 1 (0 stands for black, 1 for white, and various shades of gray in-betwe

2. `train_y` : The labels for each training datapoint, also known as the digit shown in th (a number between 0-9).
3. `test_x` : A matrix of the test data, formatted like `train_x`.
4. `test_y` : The labels for the test data, which should only be used to evaluate the accu classifiers in your report.

Next, we call the function `plot_images` to display the first 20 images of the training se and get a feel for the data (don't include these in your write-up).

Tip: Throughout the whole online grading system, you can assume the NumPy python l imported as `np`. In some problems you will also have access to python's `random` library you've already implemented. Look out for the "Available Functions" Tip before the code last project.

‹ Previous Next ›

This project will unfold both on MITx and on your local machine. However, we encourag the functions locally and run the test scripts to validate basic funcitonality. Think of the