In this assignment you will practice putting together a simple image classification pipeline, based on the k-Nearest Neighbor or the SVM/Softmax classifier. The goals of this assignment are as follows:

- understand the basic **Image Classification pipeline** and the data-driven approach (train/predict stages)
- understand the train/val/test **splits** and the use of validation data for **hyperparameter tuning**.
- develop proficiency in writing efficient **vectorized** code with numpy
- implement and apply a k-Nearest Neighbor (**kNN**) classifier
- implement and apply a Multiclass Support Vector Machine (**SVM**) classifier
- implement and apply a **Softmax** classifier
- understand the differences and tradeoffs between these classifiers
- get a basic understanding of performance improvements from using **higher-level representations** than raw pixels (e.g. color histograms, Histogram of Gradient (HOG) features)

# Setup

You can work on the assignment in one of two ways: locally on your own machine, or on a virtual machine through Terminal.

## Working locally

**Get the code:** Download the starter code here.

**[Optional] virtual environment:** Once you have unzipped the starter code, you might want to create a virtual environment for the project. If you choose not to use a virtual environment, it is up to you to make sure that all dependencies for the code are installed on your machine. To set up a virtual environment, run the following:

```
cd assignment1
sudo pip install virtualenv      # This may already be installed
virtualenv .env                  # Create a virtual environment
source .env/bin/activate         # Activate the virtual environment
pip install -r requirements.txt  # Install dependencies
# Work on the assignment for a while ...
deactivate                       # Exit the virtual environment
```

**Download data:** Once you have the starter code, you will need to download the CIFAR-10 dataset. Run the following from the `assignment1` directory:

```
cd cs231n/datasets
./get_datasets.sh
```

**Start IPython:** After you have the CIFAR-10 data, you should start the IPython notebook server from the `assignment1` directory. If you are unfamiliar with IPython, you should read our IPython tutorial.

## Working on Terminal

We have created a Terminal snapshot that is preconfigured for this assignment; you can find it here. Terminal allows you to work on the assignment from your browser. You can find a tutorial on how to use it here.

## Submitting your work:

Whether you work on the assignment locally or using Terminal, once you are done working run the `collectSubmission.sh` script; this will produce a file called `assignment1.zip`. Upload this file to your dropbox on the coursework page for the course.

## Q1: k-Nearest Neighbor classifier (30 points)

The IPython Notebook **knn.ipynb** will walk you through implementing the kNN classifier.

## Q2: Training a Support Vector Machine (30 points)

The IPython Notebook **svm.ipynb** will walk you through implementing the SVM classifier.

## Q3: Implement a Softmax classifier (30 points)

The IPython Notebook **softmax.ipynb** will walk you through implementing the Softmax classifier.

## Q4: Higher Level Representations: Image Features (10 points)

The IPython Notebook **features.ipynb** will walk you through this exercise, in which you will examine the improvements gained by using higher-level representations as opposed to using raw pixel values.

## Q5: Bonus: Design your own features! (+10 points)

In this assignment we provide you with Color Histograms and HOG features. To claim these bonus points, implement your own additional features from scratch, and using only numpy or scipy (no external dependencies). You will have to research different feature types to get ideas for what you might want to implement. Your new feature should help you improve the performance beyond what you got in Q4 if you wish to get these bonus points. If you come up with nice features we'll feature them in the lecture.

## Q6: Cool Bonus: Do something extra! (+10 points)

Implement, investigate or analyze something extra surrounding the topics in this assignment, and using the code you developed. For example, is there some other interesting question we

could have asked? Is there any insightful visualization you can plot? Or maybe you can experiment with a spin on the loss function? If you try out something cool we'll give you points and might feature your results in the lecture.