---

layout: page mathjax: true permalink: /assignments2019/assignment1/

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
- implement and apply a **Two layer neural network** 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

Get the code as a zip file here.

You can follow the setup instructions here.

Download data:

Once you have the starter code (regardless of which method you choose above), 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, with the `jupyter notebook` command. (See the Google Cloud Tutorial for any additional steps you may need to do for setting this up, if you are working remotely)

If you are unfamiliar with IPython, you can also refer to our IPython tutorial.

Some Notes

NOTE 1: There are `# *****START OF YOUR CODE`/`# *****END OF YOUR CODE` tags denoting the start and end of code sections you should fill out. Take care to not delete or modify these tags, or your assignment may not be properly graded.

NOTE 2: The submission process this year has **2 steps**, requiring you to 1. run a submission script and 2. download/upload an auto-generated pdf (details below.) We suggest ***making a test submission early on*** to make sure you are able to successfully submit your assignment on time (a maximum of 10 submissions can be made.)

NOTE 3: This year, the `assignment1` code has been tested to be compatible with python version `3.7` (it may work with other versions of `3.x`, but we won't be officially supporting them). You will need to make sure that during your virtual environment

setup that the correct version of `python` is used. You can confirm your python version by (1) activating your virtualenv and (2) running `which python`.

NOTE 4: If you are working in a virtual environment on OSX, you may *potentially* encounter errors with matplotlib due to the issues described here. In our testing, it seems that this issue is no longer present with the most recent version of matplotlib, but if you do end up running into this issue you may have to use the `start_ipython_osx.sh` script from the `assignment1` directory (instead of `jupyter notebook` above) to launch your IPython notebook server. Note that you may have to modify some variables within the script to match your version of python/installation directory. The script assumes that your virtual environment is named `.env`.

Q1: k-Nearest Neighbor classifier (20 points)

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

Q2: Training a Support Vector Machine (25 points)

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

Q3: Implement a Softmax classifier (20 points)

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

Q4: Two-Layer Neural Network (25 points)

The IPython Notebook **two_layer_net.ipynb** will walk you through the implementation of a two-layer neural network classifier.

Q5: 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.

Submitting your work

Important: *Please make sure that the submitted notebooks have been run and the cell outputs are visible.*

There are ***two*** steps to submitting your assignment:

1. Run the provided `collectSubmission.sh` script in the `assignment1` directory.

You will be prompted for your SunetID (e.g. `jdoe`) and will need to provide your Stanford password. This script will generate a zip file of your code, submit your source code to Stanford AFS, and generate a pdf `a1.pdf` in a `cs231n-2019-assignment1/` folder in your AFS home directory.

If your submission for this step was successful, you should see a display message

```
### Code submitted at [TIME], [N] submission attempts remaining. ###
```

2. Download the generated `a1.pdf` from AFS, then submit the pdf to Gradescope. If you are enrolled in the course, you should have already been automatically added to the course on Gradescope.