CECS 551 HW 2

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 from BeachBoard.

Working locally

Installing Anaconda: We recommend using the free Anaconda Python distribution, which provides an easy way for you to handle package dependencies. Please be sure to download the Python 3 version, which currently installs Python 3.6.

Anaconda Virtual environment: Once you have Anaconda installed, it makes sense to create a virtual environment for the course. 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 globally on your machine. To set up a virtual environment, run (in a terminal)

```
conda create -n cecs551 python=3.6 anaconda
```

to create an environment called **cecs551**.

Then, to activate and enter the environment, run

```
source activate cecs551
```

To exit, you can simply close the window, or run

```
source deactivate cecs551
```

Note that every time you want to work on the assignment, you should run `source activate cecs551` (change to the name of your virtual env).

You may refer to this page (https://conda.io/projects/conda/en/latest/user-guide/tasks/manage-environments.html) for more detailed instructions on managing virtual environments with Anaconda.

Python virtualenv: Alternatively, you may use python virtualenv for the project. To set up a virtual environment, run the following:

```
sudo pip install virtualenv      # This may already be installed
virtualenv -p python3 .env       # Create a virtual environment (python3)
# Note: you can also use "virtualenv .env" to use your default python (pleas
e note we support 3.6)
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 (regardless of which method you choose above), you will need to download the CIFAR-10 dataset. Run the following from the assignment2 directory:

```
cd cecs551/datasets
./get_datasets.sh
```

Start IPython

After you have the CIFAR-10 data, you should start the IPython notebook server from the assignment2 directory, with the jupyter notebook command.

If you are unfamiliar with IPython, you can also refer to our IPython tutorial (http://cs231n.github.io/ipython-tutorial/).

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.

Some Notes

NOTE 1: `assignment2` code has been tested to be compatible with python version 3.6 (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 2: If you are working in a virtual environment on OSX, you may potentially encounter errors with matplotlib due to the issues described here (https://matplotlib.org/faq/virtualenv_faq.html). 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 `assignment2` 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`.

Submitting your work

Submit **pdf** of the completed iPython notebooks to **Dropbox of BeachBoard**.

To produce a pdf of your work, you can first convert each of the .ipynb files to HTML. To do this, simply run

```
ipython nbconvert --to html FILE.ipynb
```

for each of the notebooks, where `FILE.ipynb` is the notebook you want to convert. Then you can convert the HTML files to PDFs with your favorite web browser.

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