visionTestDraft.py
Developed by Amanda Joy Panell, in collaboration with Jon Fenn and Emily Kelly

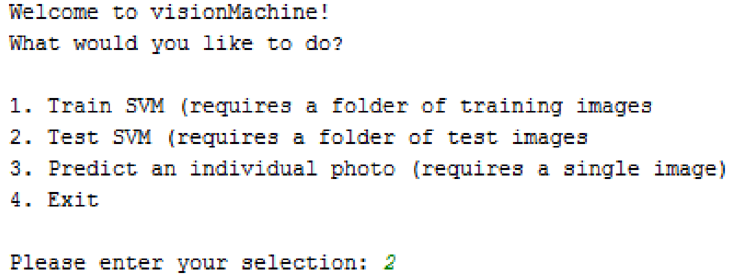
----

HOW TO USE - visionTestDraft.py

----

1. First make sure you are using Python3
2. Next make, make sure you have all the proper libraries installed. OpenCV, Numpy and pip are the only ones you should have to do yourself. Here is how to install OpenCV, the others can be installed the same way. (Assumes that you are using PyCharm)
    a. In PyCharm: go to File->Settings->Project Interpreter->
    b. Click the green plus sign on the left side of the screen
    c. In the search bar type "opencv"
    d. Select "opencv-python" from the menu
    e. Click "install package" button at the bottom of the screen
    f. Note: More details can be found here: https://www.jetbrains.com/help/pycharm/installing-uninstalling-and-upgrading-packages.html
3. Next, you’ll need to have some images to train and test the SVM with. If you are using data from our GitHub repository, you don’t need to do anything here. If you want to use your own images that’s fine. Just make sure you keep the following things in mind.
    a. The images need to be named properly. The naming format is “cameraNickname_temperture_humidity_dewpoint_time_date”
        i. Camera nickname can be pretty much anything.
            1. Example nicknames:
                a. “Medford” for a camera from Medford, Oregon
                b. “Ranch” for a camera located on a ranch
                c. “frontLawnCam” for a camera located on your front lawn
            2. Don’t separate words with underscores! When in doubt, use camel case.
        ii. Temperature, humidity and dewpoint are entered numerically, with a ‘p’ replacing any decimal points.
            1. For example, the temperature 22.5 would be represented as 22p5
        iii. Time is entered in military time, with no colons.
            1. For example, 1:54pm would be entered as 1354
        iv. Date is entered in MMDDYY format.
            1. For example, December 14th, 2017 would be entered as 121417
    b. Your test data folder AND your training data folder MUST both have 3 folders named “clear”, “foggy”, and “smoky”.
        i. Casing is important
    c. Use forward facing slashes (“/”) in your file path

4. When you run the program you will be greeted with a menu that looks like the one below. Select the option that you would like, and follow the prompts given.



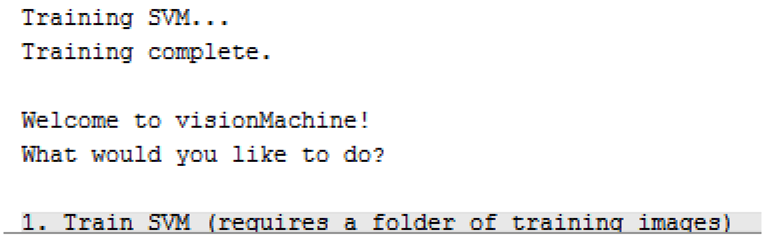
```
Welcome to visionMachine!
What would you like to do?

1. Train SVM (requires a folder of training images
2. Test SVM (requires a folder of test images
3. Predict an individual photo (requires a single image)
4. Exit

Please enter your selection: 2
```

    a. If you enter a “1”, you will be asked if you’d like to use the default training folder (the one from GitHub) or not. If not, you will be asked to enter a path to your own training folder.

        i. Once a folder is selected, all you have to do is sit back and wait. Once all the training images have been processed and the SVM has been trained, you will see the following message, and you will be returned to the menu.



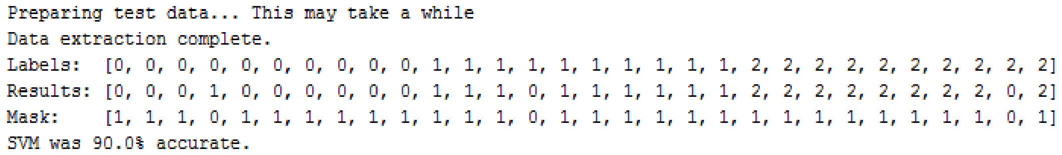
```
Training SVM...
Training complete.

Welcome to visionMachine!
What would you like to do?

1. Train SVM (requires a folder of training images)
```

    b. If you enter a “2”, you will be asked if you’d like to use the default training folder (the one from GitHub) or not. If not, you will be asked to enter a path to your own training folder.

        i. Once a folder is selected, you will have a bit of a wait before the results are ready. When the testing images have been processed and classified, you will see results similar to the ones below, before you are returned to the menu.



```
Preparing test data... This may take a while
Data extraction complete.
Labels:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
Results: [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 0, 2]
Mask:    [1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1]
SVM was 90.0% accurate.
```

        ii. Learn how to interpret results in the “HOW TO INTERPRET RESULTS” section.

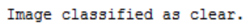
    c. If you enter a “3”, you will be asked to enter the file path of a photo you would like to have classified.

        i. This image MUST follow the proper naming format. See step 3 for more details.

        ii. Use forward facing slashes (“/”) in your file path.

        iii. Only use .jpg or .png files

        iv. Once your image has been processed and classified, you should get a result that looks like this…



```
Image classified as clear.
```

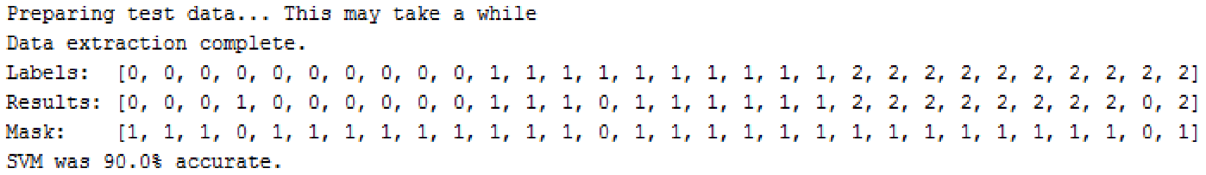
We’ll go over this result in the “HOW TO INTERPRET RESULTS” section.

d. If you entered a “4”, you will be insulted, and the program will exit.
e. If you entered any character other than 1,2,3 or 4, you will be brought back to the main menu to try again.

HOW TO INTERPRET RESULTS

In this section, we will go over the results from two cases. The first case is that you tested the SVM on a whole folder full of images, and the second case is that you used the SVM to classify a single image.

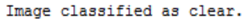
Let’s start with the first case, testing the SVM on a whole folder full of images. Your results should look something like this:



```
Preparing test data... This may take a while
Data extraction complete.
Labels:  [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2]
Results: [0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 0, 2]
Mask:    [1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1]
SVM was 90.0% accurate.
```

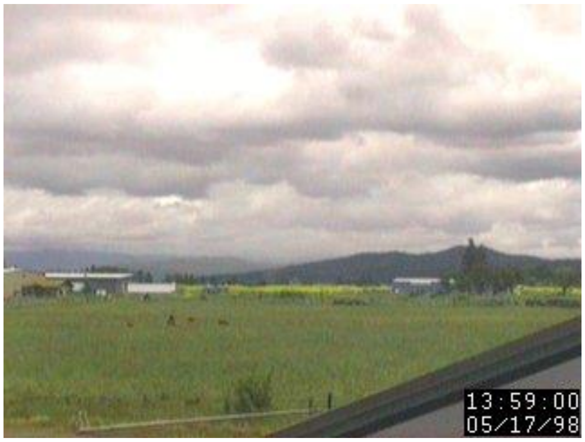
These are the results after testing the 30 test images (10 for each class) found in our GitHub repository, in the testData folder. The three lists you see are parallel. The “labels” list is what the human who sorted the images labeled each image as, by placing it into the “clear”, “foggy”, or “smoky” sub-folder. The “results” list is what the SVM classified the images as, and the “mask” list indicates which images were classified correctly. A 1 means that the SVM was right, and a 0 means that it was wrong. Finally, you will see a score representing the percentage of photos the SVM successfully classified.

The second case is simpler. Your results will look something like this:



```
Image classified as clear.
```

The meaning is simple. The image was classified as clear. The image used in this particular test was this one:



It may look bleak and possibly foggy, but this photo was labeled as clear by both the human who labeled it, and the SVM. The human labeled it this way because although it is overcast, and it looks like it may be raining in the distance, you can still clearly see all the expected features. This image was removed from the training data folder for fear that it would be too complicated for the SVM to classify, but it turns out we need not have worried. The SVM just earned itself a shiny gold star!