# CPSC 585 - Artificial Neural Networks

Project 3, Spring 2021

due March 17

*Last updated Wednesday March 3, 9:15 pm PST*

In this project you will attempt the character recognition problem from Projects 1 and 2 using shallow multiclass networks and compare their performance to the performance of multiple binary perceptrons.

The project may be completed individually, or in a group of no more than three (3) students.

## Platforms

The platform requirements for this project are the same as for [Project 1](https://docs.google.com/document/d/1UING3xL5oPf0zRLnBNyfX4N4mdDO7Aeo6XzuNgRMQ1A/edit?usp=sharing).

## Libraries and Code

Vector and matrix operations for this project must be implemented in [NumPy](https://numpy.org/), without recourse to other libraries.

Code from [*A Whirlwind Tour of Python*](https://jakevdp.github.io/WhirlwindTourOfPython/), the library documentation, and the sample code provided below may be reused. All other code and the results of experiments must be your own original work or the original work of other members of your team.

The Google Colaboratory notebook [Binary Perceptrons in NumPy](https://colab.research.google.com/drive/1ccBIlp0PJD46jqojxExDblSfQHAp0kq6?usp=sharing) demonstrates how to solve the character recognition problem from Project 1 using NumPy.

## Dataset

This project uses the same dataset as Projects 1 and 2. You do not need to have completed either of those projects in order to complete this project.

## Experiments

Run the following experiments in a Jupyter notebook, performing each action in a [code cell](https://jupyter-notebook.readthedocs.io/en/stable/notebook.html#code-cells) and answering each question in a [Markdown cell](https://jupyter-notebook.readthedocs.io/en/stable/notebook.html#markdown-cells).

1. Use from dataset import \* to load the module, then examine TRAINING\_SET, TEST\_SET, and MESSAGE.
2. In order to use the images in TRAINING\_SET, TEST\_SET, and MESSAGE, convert them into two-dimensional [NumPy arrays](https://numpy.org/doc/stable/user/basics.creation.html) of feature vectors.
3. Implementing a Python function show(image) to [reshape](https://numpy.org/doc/stable/reference/generated/numpy.reshape.html) a one-dimensional feature vector into a 7×5 array, then print() the image. Test this function on some of the images in the training set.
4. In order to use the character labels in TRAINING\_SET and TEST\_SET, convert them into an integer class vector using [ord()](https://docs.python.org/3/library/functions.html#ord), then use NumPy to convert them into 26 [one-hot](https://en.wikipedia.org/wiki/One-hot#Machine_learning_and_statistics) encoded vectors.

Note that if you did this for Project 2 using Keras, this conversion must be performed using NumPy directly.

1. Implement a multiclass perceptron as described in Section 2.3.1 of the textbook to recognize the letters A-Z.

Begin with weights initialized to random values and a learning rate of 𝛼 = 0.01. Predict the letter by choosing the class with the largest *W̅r∙X̅i* value.

If the prediction is incorrect, train the multiclass perceptron by applying the update rule of Equation 2.24 in the textbook. Continue training until all items in the training set are correctly classified.

1. Use your trained multiclass perceptron to classify each image in the test set.

Apply the weights to all feature vectors in the test set at once using [np.matmul()](https://numpy.org/doc/stable/reference/generated/numpy.matmul.html), then make predictions using [np.argmax()](https://numpy.org/doc/stable/reference/generated/numpy.argmax.html#numpy.argmax).

What accuracy do you obtain? If the accuracy is less than 100%, which test images are misclassified?

1. Implement and train a multinomial logistic regression classifier as described in Section 2.3.3 of the textbook.

Begin with weights initialized to random values and a learning rate of 𝛼 = 0.01. Predict the letter using the softmax activation function of Equation 2.29 in the textbook.

Train the classifier by applying the update rule of Equation 2.37 in the textbook until all items in the training set are correctly classified.

1. Repeat experiment *(6)* with your softmax classifier. You can find the formula for applying softmax using NumPy in the documentation for SciPy’s [softmax()](https://docs.scipy.org/doc/scipy/reference/generated/scipy.special.softmax.html) function.
2. Use your trained classifiers to identify the letters in MESSAGE

If you completed Project 1, compare the performance of that project with the performance with these classifiers. If you did not complete Project 1, use the sample code provided above under *Libraries and Code*.

If you completed Project 2 compare the performance of that project with the performance of these classifiers.

## Submission

A Markdown cell at the top of the notebook should include project summary information [as described in the Syllabus](https://docs.google.com/document/d/1TG4qngijN2ZNLpvok1_NIeE-iELeN6F2WYiFVZprbBQ/edit#heading=h.5162uorf65x7) for README files.

Since you may be actively editing and making changes to the code cells in your notebook, be certain that each of your code cells still runs correctly before submission. You may wish to do this by selecting *Run All* from the drop-down menu bar.

Submit your Jupyter .ipynb notebook file through Canvas before class on the due date.

If the assignment is completed by a team, only one submission is required. Be certain to identify the names of all students on your team at the top of the notebook. See the following sections of the Canvas documentation for instructions on group submission:

* [How do I join a group as a student?](https://community.canvaslms.com/t5/Student-Guide/How-do-I-join-a-group-as-a-student/ta-p/468)
* [How do I submit an assignment on behalf of a group?](https://community.canvaslms.com/t5/Student-Guide/How-do-I-submit-an-assignment-on-behalf-of-a-group/ta-p/294)