



**Overview:** In completing this lab, you will build on your previous tutorial experience to implement a functioning  $k$ -NN classifier for the provided `digits.csv` data.

### Getting Started:

- Create a GitHub repo named similar to `dcs211_f2025_lab5_student1_student2`, and add me as a collaborator.
- You will need to install (via pip) the `scikit-learn` library to complete this work:

```
python3 -m pip install scikit-learn
```

or use anaconda (for which `scikit-learn` should already be installed).

**Assignment:** You have been provided with a starter `dcs211_lab5.py` on Lyceum. This code provides two commented functions:

- `fetchDigit`: Given a pandas data frame corresponding to data read from `digits.csv`, returns a tuple with digit and pixels information.
- `drawDigitHeatmap`: Given a numpy array of pixels, draws a heatmap of the corresponding digit.

Make sure to inspect the code, understand what each function does, and then run the code to make sure it operates correctly.

Then, using your `iris_knn.ipynb` notebook from last time as a guide, implement a working  $k$ -NN classifier for the provided digits data. Specifications are below:

1. Write a function named `cleanTheData` that accepts a pandas data frame and returns a numpy array of the cleaned data. You can choose to return both a pandas data frame and a numpy array version of your cleaned results. (Do you need to do all the cleaning that was necessary in the iris case?) Make sure to call appropriately from inside `main`.
2. Similar to that from the tutorial, write a function named `predictiveModel` that accepts a numpy array corresponding to a training set (see next step) and a numpy array corresponding to the features (pixel values) for a given digit. Implement by hand a 1-NN implementation, and return the predicted digit. Notes:
  - In implementing 1-NN, you are just finding the one digit from the training set that is closest (in Euclidean distance, using `np.linalg.norm`) to each test digit.
  - In step 3 below, you will split your overall data set into two parts: one training set and one test set. The training set will correspond to 80% of the rows from your data set; the test set will correspond to the remaining 20% of the rows. The training set part of that data is what will be passed in for the first argument to this current function.

- The features array also being passed in will correspond to exactly one of the rows from the test set — i.e., it will be the features (pixels) only (no label) of one particular digit's image. You are treating the rows from the test set as “unknown” (i.e., yet unseen) data, even though you do know the proper label for it and therefore can determine whether it was labeled correctly.
3. Now inside `main`, split your overall numpy array data into two parts (does not need to be done at random at this point): the training set should be the first 80% of the data, and the test set should be the last 20% of the data. Using a progress bar to show progress, use your 1-NN `predictiveModel` function to predict each digit from the test set. Just iterate through each row in your test set, passing in the entire training set and the features of the row at this iteration. In a separate Google Doc, report the accuracy to 3 decimal places.
  4. Repeat the test above, but swap so that the test set is the first 20% of the data and the training set is the last 80% of the data. In your Google Doc, report the accuracy to 3 decimal places. Is the accuracy better? Worse? A good accuracy overall? Comment briefly in your Doc.
  5. Using your heatmap function, visualize the first five incorrectly predicted digits. Comment on whether, based on looking at the visualization of the pixels of each of those misclassified digits, it makes sense to you as to why the digit might have been misclassified.
  6. Consider what kind of issues there may be with the data itself — how the data was collected, from whom the data was collected, and what the data itself represents. Provide some meaningful discussion of potential issues with each of the three.
  7. Now move into the scikit-learn-assisted portion, similar to the tutorial. Write a function named `splitData` that will accept your numpy array of all data, and will return a list of (in order) `X_test`, `y_test`, `X_train`, and `y_train`. Make sure to call this appropriately from within `main`.
  8. As in the tutorial, run the  $k$ -NN classifier using a value of  $k$  that you just guess. Comment on (a) why you chose that value of  $k$ , and (b) the resulting accuracy. Use the tutorial's `compareLabels` function (with variable names changed appropriately) to print meaningful results.
  9. Now write a function named `findBestK`, with `X_train` and `y_train` as parameters, to determine the best value of  $k$ . Use `random.seed` to set up your splitting, determining the best  $k$  for the following three seeds: 8675309, 5551212, a number of your choosing. Comment on: (a) Is  $k$  the same for all? (b) What  $k$  did you choose as the best value, and why?
  10. Now write a function named `trainAndTest` — with `X_train`, `y_train`, `X_test`, and `best_k` as parameters — that will both train and test your model using your best determined value of  $k$ . **In testing, your function will use `X_test` to produce a set of predicted labels. Return those predicted labels so you can eventually (back in main) compare the returned predicted labels to the actual labels, giving you the ability to compute prediction accuracy.** Report the accuracy, and use `compareLabels` to print meaningful output. Include the output of your `compareLabels` in your Google Doc.

---

**Individual Reflection:** In the Lyceum online text, reflect individually on your work, with answers to the following questions:

- What did you find most challenging about this lab, and why?
- What did you find most rewarded about this lab, and why?
- Did you do (a) more than your fair share, (b) your fair share, or (c) less than your fair share for this lab? Justify.
- For each teammate, did they do (a) more than their fair share, (b) their fair share, or (c) less than their fair share for this lab? Justify.

---

**Submitting:** Each group must submit one `dcs211_lab5.py` (via GitHub) and accompanying Google Doc — it doesn't matter which of you submits. **Each person must submit an individual reflection on Lyceum.**