Saving Data

Learn how to save and load NumPy data.

Chapter Goals:

- Learn how to save and load data in NumPy
- Write code to save NumPy data to a file

A. Saving

After performing data manipulation with NumPy, it's a good idea to save the data in a file for future use. To do this, we use the `np.save` (https://docs.scipy.org/doc/numpy/reference/generated/numpy.save.html) function.

The first argument for the function is the name/path of the file we want to save our data to. The file name/path should have a ".npy" extension. If it does not, then `np.save` will append the ".npy" extension to it.

The second argument for `np.save` is the NumPy data we want to save. The function has no return value. Also, the format of the ".npy" files when viewed with a text editor is largely gibberish when viewed with a text editor.

If `np.save` is called with the name of a file that already exists, it will overwrite the previous file.

The code below shows examples of saving NumPy data.

```
arr = np.array([1, 2, 3])
# Saves to 'arr.npy'
np.save('arr.npy', arr)
# Also saves to 'arr.npy'
np.save('arr', arr)
```

Succeeded

B. Loading

After saving our data, we can load it again using `np.load` (`https://docs.scipy.org/doc/numpy/reference/generated/numpy.load.html`). The function's required argument is the file name/path that contains the saved data. It returns the NumPy data exactly as it was saved.

Note that `np.load` will not append the ".npy" extension to the file name/path if it is not there.

The code below shows how to use `np.load` to load NumPy data.

```
arr = np.array([1, 2, 3])
np.save('arr.npy', arr)
load_arr = np.load('arr.npy')
print(repr(load_arr))

# Will result in FileNotFoundError
load_arr = np.load('arr')
```

Output 0.333s

```
array([1, 2, 3])
```

```
Traceback (most recent call last):
  File "main.py", line 10, in <module>
    load_arr = np.load('arr')
  File "/usr/local/lib/python3.5/dist-packages/numpy/lib/npyio.py", line 372, in load
    fid = open(file, "rb")
FileNotFoundError: [Errno 2] No such file or directory: 'arr'
```

Time to Code!

The coding exercise in this chapter will require you to complete the `save_points` function, which will save some randomly generated 2-D points in a file.

You'll generate 100 (*x*, *y*) points from a uniform distribution in the range [-2.5, 2.5), then save the points to `save_file`.

Set `points` equal to `np.random.uniform`, with the `low` and `high` keyword arguments representing the lower and upper ends of the range. The `size` keyword argument should be set to `(100, 2)`.

Call `np.save` with `save_file` as the first argument and `points` as the second argument.

```
def save_points(save_file):
  points = np.random.uniform(
    low=-2.5, high=2.5, size=(100, 2))
  np.save(save_file, points)


```

Show Results | Show Console

1 of 1 Tests Passed

Result	Input	Expected Output	Actual Output	Reason
✓		array([[-1.89675382, -0.07320295], …	array([[-1.89675382, -0.07320295], …	Your code is correct, good job!

0.381s

← Back

Aggregation

Next →

Quiz

☑ Mark as Completed

Report an Issue

Ask a Question (https://discuss.educative.io/tag/saving-data__data-manipulation-with-numpy__machine-learning-for-software-engineers)