index

January 26, 2022

1 Pickle

1.1 Introduction

Pickle is an invaluable tool for saving objects. Think about the importance of being able to save data files to csv, or another format. For example, you start with a raw dataset which you may have downloaded from the web. Then you painstakingly take hours preprocessing the data, cleaning it, constructing features, aggregates, and other views. In order to avoid having to rerun your entire process, you are apt to save the current final cleaned version of the dataset. `pickle` allows you to save any object that is currently loaded into your Python interpreter. Literally anything. You could save data stored in a dictionary, list or set as a pickle file. You can also save functions, or class instances as pickle files. Saving models is one of the important use cases of this.

1.2 Objectives

You will be able to:

- Describe the circumstances in which you would want to use a pickle
- Write a pickle file
- Read a pickle file

1.3 Importing Pickle

[1]:
```
import pickle
```

1.4 Writing Objects to Pickle

[2]:
```
data_object = {
    'a': [1, 2.0, 3, 4+6j],
    'b': ('character string', b'byte string'),
    'c': {None, True, False}
}
```

[3]:
```
with open('data.pickle', 'wb') as f:
    # Pickle the 'data' dictionary using the highest protocol available.
    pickle.dump(data_object, f, pickle.HIGHEST_PROTOCOL)
```

1.5 Importing Objects from Pickle Files

[4]:
```
with open('data.pickle', 'rb') as f:
    # The protocol version used is detected automatically, so we do not
    # have to specify it.
    data_object2 = pickle.load(f)
data_object2
```

[4]:
```
{'a': [1, 2.0, 3, (4+6j)],
 'b': ('character string', b'byte string'),
 'c': {False, None, True}}
```

1.6 Pickle with scikit-learn

The example below is adapted from scikit-learn's documentation on persistence. (See link below.) In the first part of this example, a rudimentary regression model is fit to a simple dataset.

The bottom code snippet is what is pertinent to the current pickle discussion. Here, just like our previous pickle examples, you can see how to save the model instance `reg` to file. In other words, we have saved a trained model to disk. If you are using large production datasets where training can take a substantial amount of time and resources, then saving these model weights is essential. Similarly, you can see how easy it is to reload the saved model from file using `pickle.loads()`.

[5]:
```
import numpy as np
from sklearn.linear_model import LinearRegression
```

[6]:
```
X = np.array([[1, 1], [1, 2], [2, 2], [2, 3]])
# y = 1 * x_0 + 2 * x_1 + 3
y = np.dot(X, np.array([1, 2])) + 3
reg = LinearRegression().fit(X, y)
```

[7]:
```
import pickle
# Save
with open('regression_model.pickle', 'wb') as f:
    pickle.dump(reg, f)

# Load
with open('regression_model.pickle', 'rb') as file:
    reg2 = pickle.load(file)
reg2.predict(X)
```

[7]:
```
array([ 6.,  8.,  9., 11.])
```

1.7 Additional Resources

- Pickle Documentation
- scikit-learn Persistence Documentation (using pickle)

1.8 Summary

In this brief lesson you saw how to both save objects to pickle files and import objects from pickle files. This can be particularly useful for saving models that are non deterministic and would otherwise be difficult or impossible to reproduce exact replicas of.