How to Save and Load ML Models

Introduction

WHY We need to save and restore/reload later our ML Model, so as to:
a) test our model on/with new data,
b) compare multiple models,
c) or anything else.

object serialization This process / procedure of saving a ML Model is also known as object serialization - representing an object with a stream of bytes, in order to store it on disk, send it over a network or save to a database.

deserialization While the restoring/reloading of ML Model procedure is known as deserialization.

We will be covering following 2 approaches of Saving and Reloading a ML Model -
1) Pickle Approach
2) Joblib Approach

Approach 1 : Pickle approach

Pickle is a generic object serialization module that can be used for serializing and deserializing objects. While it's most commonly associated with saving and reloading trained machine learning models, it can actually be used on any kind of object. Here's how you can use Pickle to save a trained model to a file and reload it to obtain predictions.

how to Save the model After training it.

To save the model all we need to do is pass the model object into the dump() function of Pickle. This will serialize the object and convert it into a "byte stream" that we can save as a file called model.pkl. You can then store, or commit to Git, this model and run it on unseen test data without the need to re-train the model again from scratch.

```
# Import pickle Package

import pickle

Pkl_Filename = "model.pkl "

with open (Pkl_Filename, 'wb') as file:

pickle.dump(LR_Model, file)
```

or

```
pickle.dump(model, open('model.pkl', 'wb'))
```

Load the model

To load a saved model, all you need to do is pass the "pickled" model into the Pickle load() function and it will be deserialized. By assigning this back to a model object, you can then run your original model's predict() function, pass in some test data and get back an array of predictions.

```
with open (Pkl_Filename, 'rb') as file:

Pickled_LR_Model = pickle.load(file)
```

or

```
pickled_model = pickle.load(open('model.pkl', 'rb'))

pickled_model.predict(X_test)
```

Let's Reflect back on Pickle approach :

PROs of Pickle :
1) save and restore our learning models is quick - we can do it in two lines of code.
2) It is useful if you have optimized the model's parameters on the training data, so you don't need to repeat this step again.
CONs of Pickle :
1) it doesn't save the test results or any data.

Approach 2 - Joblib :

The Joblib Module is available from Scikit Learn package and is intended to be a replacement for Pickle, for objects containing large data.

This approach will save our ML Model in the pickle format only but we dont need to load additional libraries as the 'Pickling' facility is available within Scikit Learn package itself which we will use invariably for developing our ML models.

In following Python scripts , we will show how to Save and reload ML Models using Joblib

```
# Import Joblib Module from Scikit Learn

from sklearn.externals import joblib
```

Save the Model using Joblib

```
joblib_file="joblib_RL_Model.pkl"

joblib.dump(LR_Model,joblib_file)
```

Reload the saved Model using Joblib

```
joblib_LR_model = joblib.load(joblib_file)

Reload the Saved Model using Joblib
```

Let's Reflect back on Joblib approach :

PROs of Joblib :
1) the Joblib library offers a bit simpler workflow compared to Pickle.
2) While Pickle requires a file object to be passed as an argument, Joblib works with both file objects and string filenames.
3) In case our model contains large arrays of data, each array will be stored in a separate file, but the save and restore procedure will remain the same.
4) Joblib also allows different compression methods, such as 'zlib', 'gzip', 'bz2', and different levels of compression.

References:

1- https://www.geeksforgeeks.org/saving-a-machine-learning-model/
2- https://medium.com./how-to-save-your-machine-learning-model-using-pickle-and-joblib-c403f98b5d
3- https://www.kaggle.com/prmohanty/python-how-to-save-and-load-ml-models