* **joblib is usually significantly faster on large numpy arrays** because it has a special handling for the array buffers of the numpy datastructure. To find about the implementation details you can have a look at the [source code](https://github.com/joblib/joblib/blob/master/joblib/numpy_pickle.py). It can also compress that data on the fly while pickling using zlib or lz4.
* **joblib also makes it possible to memory map** the data buffer of an uncompressed joblib-pickled numpy array when loading it which makes it possible to share memory between processes.
* **if you don't pickle large numpy arrays, then regular pickle can be significantly faster, especially on large collections of small python objects** (e.g. a large dict of str objects) because the pickle module of the standard library is implemented in C while joblib is pure python.
* since PEP 574 (Pickle protocol 5) has been merged in Python 3.8, it is now much more efficient (memory-wise and cpu-wise) to pickle large numpy arrays using the standard library. Large arrays in this context means 4GB or more.
* But **joblib can still be useful with Python 3.8 to load objects that have nested numpy arrays** in memory mapped mode with mmap\_mode="r".

t1 = time()

joblib.load("classi.pickle")

print "time for loading file size joblib", os.path.getsize("classi.pickle"),"KB =>", time()-t1

time for loading file size with pickle 79708 KB => 0.16768

time for loading file size with cpickle 79708 KB => 0.00023

time for loading file size joblib 79708 KB => 0.00068

**Save Model To a File Using Python Pickle**

**import** pickle

**with** open('model\_pickle','wb') **as** file:

pickle**.**dump(model,file)

**Load Saved Model**

**with** open('model\_pickle','rb') **as** file:

mp **=** pickle**.**load(file)

mp**.**coef\_

mp**.**intercept\_

mp**.**predict([[5000]])

Out[13]:

array([859554.79452055])

**Save Trained Model Using joblib**

**from** sklearn.externals **import** joblib

joblib**.**dump(model, 'model\_joblib')

**Load Saved Model**

mj **=** joblib**.**load('model\_joblib')

mj**.**coef\_

mj**.**intercept\_

mj**.**predict([[5000]])

Out[19]:

array([859554.79452055])

READ / WRITE CSV

df = pd.read\_csv("homeprices.csv")

df.to\_csv(‘prediccion.csv’)