In [1]:
```
import time
from sklearn.model_selection import cross_val_score
from scipy.stats import uniform
from mango import Tuner, scheduler
from xgboost import XGBClassifier
from sklearn.datasets import load_wine
```

In [2]:
```
X, y = load_wine(return_X_y=True)
```

In [3]:
```
param_dict = {
    "learning_rate": uniform(0, 1),
    "n_estimators": range(1,300),
    "booster":['gbtree','gblinear','dart']
}
```

Serial

In [4]:
```
@scheduler.serial
def objective_serial(**params):
    clf = XGBClassifier(**params)
    result = cross_val_score(clf, X, y, scoring='accuracy').mean()
    return result
```

In [5]:
```
start_time = time.time()
tuner = Tuner(param_dict, objective_serial)
results = tuner.maximize()
total_time = time.time() - start_time
```

```
  0%|          | 0/20 [00:00<?, ?it/s]
```

In [6]:
```
total_time
```

Out[6]:
```
49.87681579589844
```

Parallel

In [7]:
```
@scheduler.parallel(n_jobs=-1)
def objective_parallel(**params):
    clf = XGBClassifier(**params)
    result = cross_val_score(clf, X, y, scoring='accuracy').mean()
    return result
```

In [8]:
```
import multiprocessing
multiprocessing.cpu_count()
```

Out[8]:
```
16
```

In [ ]:
```
start_time = time.time()
tuner = Tuner(param_dict, objective_parallel)
results = tuner.maximize()
total_time = time.time() - start_time
```

```
  0%|          | 0/20 [00:00<?, ?it/s]
```

In [11]:
```
total_time
```

Out[11]:
```
81.95488786697388
```

In [ ]:
```

```