Train In Data

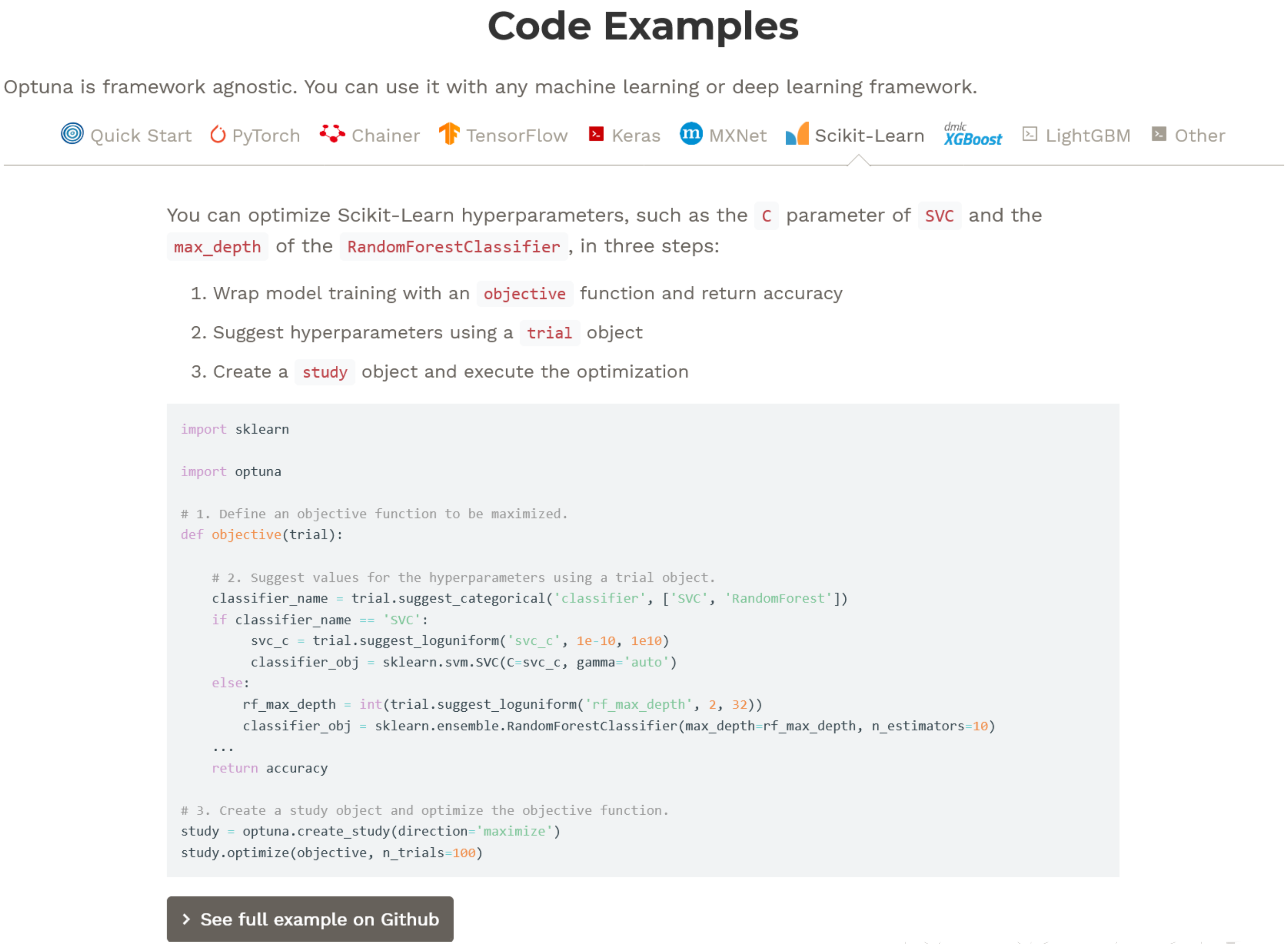
OPTUNA

Optuna

- Website: https://optuna.org/
- Docs: https://optuna.readthedocs.io/en/stable/index.html
- Github: https://github.com/optuna/optuna
- Article: https://arxiv.org/abs/1907.10902

Optuna

https://optuna.org/



Code Examples

Optuna is framework agnostic. You can use it with any machine learning or deep learning framework.

You can optimize Scikit-Learn hyperparameters, such as the `C` parameter of `SVC` and the `max_depth` of the `RandomForestClassifier`, in three steps:

1. Wrap model training with an `objective` function and return accuracy
2. Suggest hyperparameters using a `trial` object
3. Create a `study` object and execute the optimization

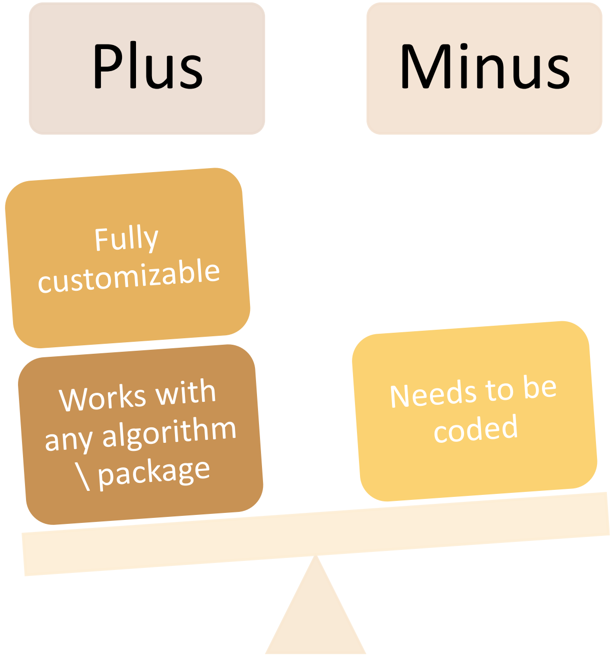
```
import sklearn

import optuna

# 1. Define an objective function to be maximized.
def objective(trial):

    # 2. Suggest values for the hyperparameters using a trial object.
    classifier_name = trial.suggest_categorical('classifier', ['SVC', 'RandomForest'])
    if classifier_name == 'SVC':
         svc_c = trial.suggest_loguniform('svc_c', 1e-10, 1e10)
         classifier_obj = sklearn.svm.SVC(C=svc_c, gamma='auto')
    else:
        rf_max_depth = int(trial.suggest_loguniform('rf_max_depth', 2, 32))
        classifier_obj = sklearn.ensemble.RandomForestClassifier(max_depth=rf_max_depth, n_estimators=10)
    ...
    return accuracy

# 3. Create a study object and optimize the objective function.
study = optuna.create_study(direction='maximize')
study.optimize(objective, n_trials=100)
```

Optuna – Search algorithms

- Grid Search
- Random Search
- Tree-structured Parzen Estimators (TPE)
- CMA-ES
- Multi-objective sampler using the NSGA-II algorithm
- Multi-objective sampler using the MOTPE algorithm

Optuna – objective function

Objective function created by user



Optuna – Distributions

- Samples Reals, Integers and Categories
- Uniform and Log-Uniform distributions
- Define-by-run API ➔ Nested spaces

Optuna – Acquisition Function

- Uses function described in original work that introduced each algorithm.

- We can’t choose which function to use

- Expected Improvement (EI)

- Expected hypervolume improvement (EHVI)

Optuna – Search analysis

- Can store the search in a SQL like DB
- Study object returns a dataframe with search data
- Built-in functions for plotting

Parallelization - SQLite

- Allows search in parallel storing in SQL like database
  - ✓ Reduced efficiency per (single) evaluation
  - ✓ Increased overall efficiency by saving time

Train In Data

THANK YOU

www.trainindata.com