ENSEA

Optimization Techniques for Machine Learning

Alassane KONE
(ISE 2019 – Senior Data Scientist)

February 2023

Chapter 7

Model Selection Optimization

Model Selection as Optimization

Model selection involves choosing one from among many candidate machine learning models for a predictive modeling problem.

Really, it involves choosing the machine learning algorithm or machine learning pipeline that produces a model. This is then used to train a final model that can then be used in the desired application to make predictions on new data.

This process of model selection is often a manual process performed by a machine learning practitioner involving tasks such as preparing data, evaluating candidate models, tuning well-performing models, and finally choosing the final model.

This can be framed as an optimization problem that subsumes part of or the entire predictive modeling project.

Model Selection: Function inputs are data transform, machine learning algorithm, and algorithm hyperparameters --> optimization problem that requires an iterative global search algorithm.

Model Selection as Optimization

Increasingly, this is the case with automated machine learning (AutoML) algorithms being used to choose an algorithm, an algorithm and hyperparameters, or data preparation, algorithm and hyperparameters, with very little user intervention.

There are many open-source AutoML libraries, although, the best-of-breed libraries that can be used in conjunction with the popular scikit-learn Python machine learning library.

- **Hyperopt-Sklearn**
- **Auto-Sklearn**
- **TPOT.**

Although fast and efficient, such approaches are still unable to outperform hand-crafted models prepared by highly skilled experts, such as those participating in machine learning competitions.

Tree-based Pipeline Optimization Tool (TPOT)

```
pip install tpot

# check tpot version
import tpot
print('tpot: %s' % tpot.__version__)

# example of tpot for a classification dataset
from sklearn.datasets import make_classification
from sklearn.model_selection import RepeatedStratifiedKFold
from tpot import TPOTClassifier
# define dataset
X, y = make_classification(n_samples=100, n_features=10, n_informative=5, n_redundant=5, random_state=1)
# define model evaluation
cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
# define search
model = TPOTClassifier(generations=5, population_size=50, cv=cv, scoring='accuracy', verbosity=2,
random_state=1, n_jobs=-1)
# perform the search
model.fit(X, y)
# export the best model
model.export('tpot_best_model.py')
```