Hyperparameter Tuning and GridSearchCV

Justin Hooker
Sy Ahmed

Parameters Vs. Hyperparameters

- Rule of Thumb:
  - If you have to specify a model parameter manually, then it is probably a model hyperparameter
- Parameter Examples
  - The coefficients in a linear regression
  - The support vectors in a support vector machine
- Hyperparameter Examples
  - The K in K-nearest neighbors
  - The C and gamma for support vector machines

Hyperparameter Tuning

- `estimator.get_params()`
- The process involves a search of the hyper-parameter space for the best cross-validation score
- A search consists of:
    - An estimator (regressor or classifier such as `sklearn.svm.SVC()`);
    - A parameter space;
    - A search method;
    - A cross-validation scheme; and
    - A score function

Hyperparameter Tuning

- Visual Reference
    - http://cs.stanford.edu/people/karpathy/svmjs/demo/
    - SVM demo

A Search Method

- GridSearchCV
    - Exhaustive consideration of all parameter combinations for given values
    - Requires a parameter grid
    - Computationally intensive
    - Curse of dimensionality
- RandomizedSearchCV
    - Implements a randomized search over parameters
    - Each setting is sampled over a distribution of possible parameter values
    - A “computation budget”, being the number of sampled iterations, chosen independent of parameter values
    - A distribution over possible values or list of discrete values can be specified

Parameter Grid

GridSearchCV

```
param_grid = [
  {'C': [1, 10, 100, 1000], 'kernel': ['linear']},
  {'C': [1, 10, 100, 1000], 'gamma': [0.001, 0.0001],
'kernel': ['rbf']},
 ]
```

RandomizedSearchCV

```
param_dist={'C': scipy.stats.expon(scale=100),
'gamma': scipy.stats.expon(scale=.1),
  'kernel': ['rbf'], 'class_weight':['balanced', None]}
```

Specifying an Objective Metric

- By default, parameter search uses the score function of the estimator to evaluate a parameter setting.
    - For classification: sklearn.metrics.accuracy_score
    - For regression: sklearn.metrics.r2_score
- An alternative scoring function can be specified via the scoring parameter to GridSearchCV and RandomizedSearchCV

Putting it All Together

Using Pipeline and GridsearchCV for hyperparameter tuning

```
clf = RandomForestClassifier()
steps = [("my_classifier", clf)]
parameters =
dict{my_classifier__min_samples_split=[2, 3, 4, 5]}
### "my_classifier" is the name of the random
forest classifier in the steps list;
min_samples_split is the associated sklearn
parameter that I want to vary
pipe = Pipeline(steps)
cv = GridSearchCV( pipe, param_grid = parameters)
```

Sample Code

Demo of the GridSearch process, tuning the K hyperparameter of the KNN algorithm